Improving Classification Quality of Support Vector Machines on Hand Eczema Images

Stefan Schnürle

Competence Center Distributed Secure Software Systems (D3S)
Lucerne University of Applied Sciences and Arts
6048 Horw, Switzerland
stefan.schnuerle@hslu.ch

Master Thesis
Master of Science in Engineering (MSE)

24th June 2016

Advisor: Prof. Dr. Ruedi Arnold
Lucerne University of Applied Sciences and Arts, 6048 Horw, Switzerland
ruedi.arnold@hslu.ch

Expert: Dr. Stanislas Nanchen
Deep Impact AG, 8400 Winterthur, Switzerland
stanislas.nanchen@deep-impact.ch

I declare that this thesis has been composed by myself, and describes my own work unless otherwise acknowledged in the text. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Horw, 24th June 2016, Stefan Schnürle

Abstract

Hand Eczema is one of the most frequent dermatoses with grave consequences for patients as well as society as a whole, potentially leading to impairment or disability to work in many professions. *Computer-aided detection* of hand eczema could support patients in their decision whether to visit a dermatologist. Moreover, the amount of time a dermatologist has to spend on manually calculating certain scores for eczematous skin is substantial, a process in which a computer-aided system of adequate quality could be of assistance.

Previous work has shown that computer-aided detection of hand eczema from photographies is feasible in principle, but has failed to reach a satisfying quality. An average F_1 score of 35% with an average accuracy of 80.9% for images of the front side of hands and an average F_1 score of 25.7% with an average accuracy of 68.3% for back sides is reported.

This thesis investigates several promising approaches to improving classification quality by utilising *Support Vector Machines (SVMs)*. The data provided to the SVM consists of several features extracted from the images, including *Texton* frequencies, several *colour moments* and metrics calculated from *grey level co-occurrence matrices*. To obtain the necessary labels, dermatologists have provided markings of eczematous regions on the hands images. These markings are consolidated into a single *consensus diagnosis* for each image.

Classification quality is considerably improved compared to previous work. For the best performing experiments, this thesis reports an F_1 score on front sides of hands of 58.6% with an accuracy of 89.3%, G mean of 74.7%, and an area misclassification error rate of 4.6 percentage points. For back sides of hands, the F_1 score reached is 44.1% with an accuracy of 88.7%, G mean of 75.6%, and an area misclassification error rate of 7.4 percentage points.

Abstract (Deutsch)

Handekzeme gehören zu den am häufigsten auftretenden Hautkrankheiten. Sie ziehen ernsthafte Konsequenzen sowohl für die betroffenen Patienten als auch die Gesellschaft als Ganzes nach sich, da sie von Beeinträchtigungen bei der Arbeit bis hin zu völliger Arbeitsunfähigkeit führen können. Eine *computerassistierte Erkennung* von Handekzemen könnte Patienten in ihrer Entscheidung, eine Dermatologin aufzusuchen, unterstützen. Darüber hinaus muss die Dermatologin bestimmte Merkmale über die Schwere des Ekzembefalls der Haut bestimmen, was mit erheblichem Zeitaufwand verbunden ist. Auch hierbei könnte ein computerassistiertes System unterstützen, wenn es in der Lage ist, Ekzeme mit ausreichender Qualität zu erkennen.

Frühere Arbeiten haben aufgezeigt, dass die computerassistierte Erkennung von Handekzemen auf Fotografien von Händen prinzipiell möglich ist. Allerdings konnte keine zufriedenstellende Qualität erreicht werden: Es werden Durchschnittswerte für das F_1-Mass in Höhe von 35% bei einer einer Genauigkeit (accuracy) von 80.9% auf Bildern von Handvorderseiten und Durchschnittswerte für das F_1-Mass in Höhe von 25.7% bei einer Genauigkeit (accuracy) von 68.3% erzielt.

Diese Masterarbeit untersucht einige ausgewählte, vielversprechende Ansätze um die Klassifizierungsqualität durch *Support Vector Machines (SVMs)* zu verbessern. Als Eingabedaten für die SVM wurden bestimmte Merkmale der Handbilder verwendet. Diese setzen sich zusammen aus Verteilungen von *Textonen*, einer Anzahl von *Farbmomenten* und einiger ausgewählter Werte, die aus *Grauwertematrizen* berechnet werden. Die für das Verfahren notwendige Kennzeichnung der Klassenzugehörigkeit der Daten wird aus Markierungen von Handekzemen berechnet, welche von Dermatologen auf den Handbildern vorgenommen wurden. Diese Markierungen werden zu einer *Konsensdiagnose* für jedes Bild konsolidiert.

Die Klasssifizierungsqualität ist im Vergleich zu früheren Arbeiten wesentlich verbessert. Die durchgeführten Experimente mit den besten Ergebnissen erreichen auf Bildern von Handvorderseiten ein F_1-Mass von 58.6% mit einer Genauigkeit (accuracy) von 89.3%, einem G-Mean-Mass von 74.7% und einer Flächenfehlklassifizierungsrate von 4.6 Prozentpunkten. Auf Bildern von Handrückseiten werden ein F_1-Mass von 44.1% mit einer Genauigkeit (accuracy) von 88.7%, einem G-Mean-Mass von 75.6% und einer Flächenfehlklassifizierungsrate von 7.4 Prozentpunkten erreicht.

Contents

Abstract **i**

Abstract (Deutsch) **iii**

1 Introduction **1**

1.1 Motivation: Hand Eczema 2

1.2 Research Questions and Expected Results 4

1.3 Starting Position 5

1.4 Organisation of this Thesis 13

1.5 Acknowledgements 13

2 Previous and Related Work **15**

2.1 Previous Work 15

2.2 Related Work 17

3 Improving Classification Quality **27**

3.1 Features 27

3.2 Pre-Processing the Data 43

3.3 Support Vector Machine 48

3.4 Evaluation Setting 58

3.5 Implementation 64

4 Experiments **71**

4.1 Features Used In Experiments 71

4.2 Experiments 71
4.3 Metrics 79
4.4 Baseline Classifiers 87

5 Results **89**
5.1 Software Component 89
5.2 Improving Classification Quality: Summary 92
5.3 Improving Classification Quality: Details 96

6 Conclusion **123**

A SkinAppWeb Data Cleaning **133**
A.1 Delete Users 133
A.2 Determine Experts Having Seen Duplicated Images 135
A.3 Delete Illicit Markings 135

B KNIME SVM Bash Scripts **137**
B.1 LIBSVM train 137
B.2 LIBSVM predict 138
B.3 SVMPerf train 138
B.4 SVMperf predict 139

C Experiments Details **141**
C.1 Textons Only 142
C.2 Colour Moments, GLCM 143
C.3 Textons, Colours, GLCM 144
C.4 Many Features 145
C.5 SVM Comparison 146
C.6 Oversampling, Data Cleaning 147

Chapter 1

Introduction

This thesis aims to improve the results of previous work conducted on the computer-aided detection of hand eczema. (Schnürle, 2016; Suter, 2012, 2013; Suter et al., 2014), as well as this thesis, utilise photographies of both front sides and back sides of hands associated with experts' diagnoses. This thesis is a direct continuation of the work conducted in (Schnürle, 2016), see section 2.1.

The fundamental hypothesis is that the two classes of skin (eczematous and healthy) on the photographies labelled by the experts are separable by utilising supervised machine learning methods to elicit meaningful predictions on previously unseen examples of similar photographies. Thus, the approaches pursued in this thesis utilise *Support Vector Machines (SVMs)* as introduced by (Vapnik, 2000) and as investigated in (Schnürle, 2016) with promising, but not yet satisfactory results.

Certain sections in this thesis start with a disclaimer pointing out that the corresponding following text is collaborative work between three master students. See subsection 1.3.1 for details.

1.1 Motivation: Hand Eczema

Disclaimer: This section 1.1 is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

The following medical assessment of the situation regarding hand eczema in Switzerland and around the world is a synthesis of three previous project proposals, which are not published (Arnold et al., 2012; Hofstetter et al., 2016; Pouly et al., 2015).

Hand eczema is one of the most frequent dermatoses, affecting up to 14% of the population (Hald et al., 2008), and more than 20% have a moderate-to-severe disease requiring intensive treatment. It can occur at all ages, commonly follows a chronic course and persists 10-15 years after onset (Meding et al., 2005). Hand eczema becomes manifest in redness of the skin, scaling, fissures, pain, itching, papules, and blisters, as well as clear discharge and sometimes bacterial infection with subsequent pustules. Although the area it usually affects is small, it can have grave consequences for the patients as well as society as a whole, leading to impairment or disability to work in many professions. Hand eczema ranks among the most prevalent occupational diseases (Diepgen, 2003) and can be caused by e.g. exposure to contact allergens and irritants (Hald et al., 2008). Thus, they are classified in allergic and irritative types (Johansen et al., 2011), depending on the putative or proven eliciting compound. Allergic hand eczema is usually associated with positive reactions in patch tests, a procedure in which suspected eczema-causing compounds are glued to the back skin of the patient for 2-3 days. Irritative eczema is much harder to prove, and the diagnosis usually remains circumstantial, taking into account the profession, possibly wet working environment and rather unspecific test such as alkali resistance testing.

Costs caused by occupational contact diseases (OCD) consist of direct medical costs and indirect costs associated with lost workdays and loss of productivity. However, costs of occupational retraining and eventually disability payments as well as costs attributable to the effects of impaired quality of life have to take into account as well (Diepgen, 2003). Data on the economic impact of OCD reveals that in the United States of America alone, annual direct costs for physicians, clinic services, and prescription drugs were as high as USD 1.6 billion, with indirect costs of approximately USD 566 million for lost productivity (Meding et al., 2005). In the Netherlands, direct medical costs alone were estimated at EUR 42 million in 1995. In Europe, OCD constitute up to 30% of all occupational diseases, for which compensation is payable with an average incidence rate of 0.5 to 1.9 cases per 1'000 full-time workers per year (Diepgen, 2003). Since these figures originate from registries, they may even underestimate the actual number of cases as registries are known to be incomplete due to under-diagnosis and under-reporting of diseases (Diepgen, 2003). British studies reported an estimate of approximately 4 million working days to be lost every year resulting from work-related skin diseases. The costs for these lost days were estimated to be approximately GBP 200 million per year (English, 2004).

Management of OCD consists of prevention and therapy. Hand eczema is best treated with prevention. Allergic disease can be controlled by avoiding contact with allergens, once identified. Irritant hand eczema, however, tends to relapse independent of the nature of the offending irritating stimulus. Thus, once obvious irritants such as wet work, extreme temperatures, etc. are avoided, primary prevention (i.e. elimination of cutaneous exposure) is not trivial and even not always possible, e.g. if the patients cannot change occupation and therefore cannot avoid encountering irritating agents or allergens.

If hand eczema cannot be prevented, e.g. by changing profession, early detection of exacerbation is key to an effective treatment (Diepgen, 2003). Moisturisers and gloves are often used as secondary prevention to avoid relapses. However, these have weak adherence rates by affected patients. More severe or recalcitrant cases require the application of topical steroids class III/IV for a few days to reduce disease activity. Photo(chemo)therapy is very effective for hand eczema as well. Systemic therapy has been revolutionised recently by the introduction of alitretinoin. This drug is a retinoid that is metabolised quickly and can greatly reduce inflammation at the affected sites.

Dermatitis patients rely on their own perception whether their skin condition is stable or worsening. Because of gradual changes and habituation, they often miss the time point when early treatment of a relapse would be efficacious to prevent full-fledged eczema. A study that evaluated how different physicians quantify the severity of atopic dermatitis has shown that there is a good agreement in the assessment of the overall extent of the disease between them. However, it was also found that there is a significant variation when assessing the severity of the illness according to different criteria (Sprikkelman et al., 1997). Considering this situation, establishing a way to objectively quantify eczematous skin changes to support diagnostic and therapeutic decisions by physicians as well as patients is of utmost importance.

1.2 Research Questions and Expected Results

The following goals have been specified for this thesis:

- **Software Component**
  Development of a software component which provides a classification service for hand eczema on images of hands, including the specification of a sensible interface and the implementation of the necessary software. For testing and demonstration purposes, a prototype web client shall be developed. This goal, including the integration into existing code, shall be realised in close cooperation with the SkinApp code owner, T. Koller.
- **Improving Classification Quality**
  Improving classification quality of hand eczema utilising SVMs is the main goal of this thesis, following up the results and insights gained and reported by previous work (Schnürle, 2016). This task includes analysing the available material of hands images as well as a systematic search for optimisation opportunities. The results shall be compared to other classification methods. To accomplish this goal, M. Pouly and T. vor der Brück (both members of the SkinApp team) shall be consulted. All data has to be scientifically and reproducibly collected and documented, such that they are publishable. From a medical point of view, the number of eczematous pixels or the ratio of said pixels to the whole hands area are particularly interesting.

During the work on this thesis, it turned out that reaching the goal of improving classification quality is more demanding than initially estimated, and first successes could only be achieved at a late stage of this thesis. Moreover, after conducting a meeting with the business partner, the need for a prototype of the software component seemed questionable, since the business partner already plans on integrating SkinApp into their software.

Thus, in agreement between T. Koller, R. Arnold and the author of this thesis, developing a software component is postponed and will be realised either by the business partner or future work at the University. Nevertheless, an architecture for the component is proposed and presented in section 5.1.

The following research questions have been compiled for which this thesis aims to find satisfying answers:

- **Classification Quality**
  To what extent can an SVM based method for the classification of hand eczema be improved, given the available set of hands images?
- **Image Material**
  How does additional image material or image material of different qualities influence the training and classification stages? For example, is a consensus diagnosis always needed, or is it feasible to conduct certain process steps on data of lesser quality?
- **Hybrid Approaches**
  Can the classification performance be increased by incorporating hybrid approaches (ensemble learning)?
- **Features**
  Which additional features provide for an improvement in classification quality? How can colour information be exploited?
- **Area Measurements**
  How does classification quality correlate with the predicted eczematous area on the hands? For example, when measuring the positive predicted hands area regardless of whether the predictions are true or false, considering the area solely, false positive and false negative pixels might cancel each other out.

1.3 Starting Position

The starting position of this thesis is mostly the state of previous work conducted in (Schnürle, 2016). It was shown that the applicability of an SVM to attack the problem of classifying hand eczema is feasible in principle, although the resulting quality did not yet reach a satisfying level. Nevertheless, the code base established in (Schnürle, 2016) has proven to be reliable and extensible and is this re-used in this thesis.

M. Pouly hinted at the *Konstanz Information Miner*, or *KNIME*[1], an Eclipse[2]-based analytics platform best used for data mining and prototyping machine learning approaches. In several discussions and a workshop with him as well as other researchers at the institute, it was estimated that KNIME would be a very powerful tool for prototyping most approaches to be pursued in this thesis. Thus, more aspects of improving classification quality could be investigated than it would have been possible with a more time-consuming implementation in C++. See section 3.5 for excerpts of workflows implemented in this thesis.

[1] see https://www.knime.org/ (accessed 2016-06-09)
[2] see https://eclipse.org/ (accessed 2016-06-09)

The idea for extracting colour information from the available hands images has been deemed potentially beneficial already while conducting previous work (Schnürle, 2016). The approaches to integrating colour information in this thesis are rooted in the work of (Al Abbadi et al., 2010), located by T. vor der Brück. His searches preliminary to this thesis are therefore an important aspect of the starting position.

1.3.1 Project Settings

As elaborated on in (Schnürle, 2016), "'SkinApp' has become an umbrella term for various research conducted at the D3S[3] Institute, e.g., whether it is possible to classify images taken by a mobile phone, offering the classification in terms of software as a service, how to obtain hands images within controlled conditions in the first place, or how to achieve colour and scaling invariance for images taken under varying circumstances". SkinApp is currently funded by the Swiss CTI[4] and the Lucerne University of Applied Sciences and Arts and is based on previous projects and master studies.

Not only this master thesis is conducted within the scope of the SkinApp CTI project. At the same time, two other master theses are engaged in researching different machine learning approaches. Manuel Brun's thesis explores new research conducted by the authors of the *statistical learning approach* (Varma and Zisserman, 2005) on which previous work (Suter, 2012, 2013; Suter et al., 2014) is based. Patrick Buchter's thesis researches the applicability of *convolutional neural networks (CNNs)* on the problem of hand eczema detection. Depending on the outcome of those two theses, hybrid approaches might be considered between SVMs and CNNs as e.g. briefly introduced in subsection 2.2.7. Manuel Brun's thesis could provide for different features which could prove beneficial for this thesis' SVM approach.

Thus, the project settings have been unique in the sense that it was possible to not only discuss the theses with advisors and experts from the University but between students, too. Weekly meetings have been conducted which proved to be of enormous value for this thesis.

This thesis utilises functionality provided by the *OpenCV* framework version 2.4.11[5], which implements a variety of algorithms used in previous work and this thesis. Contrary to previous work in (Schnürle, 2016), the *LIBSVM* implementation of OpenCV is not used. Instead, the latest native LIBSVM binaries in version 3.21 (Chang and Lin, 2011) are incorporated into the workflows. The already mentioned KNIME software is utilised in version 3.2.1.

[3] https://www.hslu.ch/de-ch/technik-architektur/forschung/kompetenzzentren/distributed-secure-software-systems/ (accessed 2016-06-21)

[4] The Commission for Technology and Innovation CTI, 3003 Bern, Switzerland; https://www.kti.admin.ch/kti/en/home.html (accessed 2016-06-21)

[5] see http://opencv.org/ (accessed 2016-06-23)

Figure 1.1: SkinAppWeb

1.3.2 SkinAppWeb

Disclaimer: This subsection 1.3.2, excluding subsections 1.3.2.1 and 1.3.2.2, is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

SkinAppWeb is a web application developed by the D3S Institute to collect markings from experts of eczema regions on hands images. Ten different hands images are presented to each expert sequentially in a statistically sound, randomised fashion. Those hands images have been acquired by A. Navarini from the Department of Dermatology, University Hospital Zurich, Switzerland. With the help of paintbrush and eraser tools similar to those in common image editing software, eczematous regions can be highlighted. Thus, *markings* from different experts on different hands images are obtained. Figure 1.1 shows a screenshot of SkinAppWeb after some areas have been highlighted.

The images presented to the experts, and thus the markings obtained, are scaled down to a width of 1'000 pixels to be presentable on smaller screens. Therefore, those markings have to be interpolated to the original hands image's size.

Each image's ID referred to throughout this thesis corresponds to its primary key in the SkinAppWeb database, thus providing for traceability without the need for additional mappings.

1.3.2.1 Setup, Data Preparation, and Cleaning

Accidentally, duplicate images were included in the first run of SkinAppWeb. The reasons for how this happened could not be completely reconstructed. Most notably, it was far from obvious that duplicates were present since they existed as separate files with no apparent connection between them. Moreover, for some of the duplicates, one file was stored as JPG and the other as PNG, which hindered the recognition as duplicates even more.

Because of this, certain experts have seen the same image twice, which distorts the results. The following process describes the mitigation of this problem. Thus, the data used in this thesis did not contain duplicated images (which applies to the other two master theses conducted at the same time as well, since they operated on the same data, see subsection 1.3.3).

The duplicated images are the following (in parenthesis the actual file format):

1. image **#229**: X887X69D.jpg (PNG) == image #235: YX8XX5XD.jpg (JPG)
2. image **#239**: X887X69P.jpg (PNG) == image #245: YX8XX5XP.jpg (PNG)
3. image **#246**: YY68Y75D.jpg (PNG) == image #237: YY8X5X8D.jpg (JPG)
4. image **#227**: YY68Y75P.jpg (PNG) == image #247: YY8X5X8P.jpg (JPG)

The bold image numbers indicate those images that are kept, whereas the images on the right-hand side are cleaned with the following process, which was defined together with M. Pouly and J. Hofstetter of the D3S Institute. The detailed description aims to provide for a well-documented and thus reproducible process of the transition from raw to "cleaned" data.

1. Obtain the raw, "unclean" data from the web server and the database.
2. Push to and tag in the Gitlab repository[6].
3. Delete data of administrative or test users from the database (see listing A.1).
4. Manually create empty markings, since in case an expert marked nothing on an image and just continued to the next, neither the (empty) PNG is created in the file system nor the corresponding entry in the column *markings.baseMarking* is populated. Thus, this expert's diagnosis would be ignored if the entry would not be created manually[7]. Marking #629 should exist, since its *date* column corresponds to other markings of the same expert.
5. Delete empty pre-reserved markings which have not yet been marked by the expert.
6. Determine which experts were shown both images of a duplicated image pair (see listing A.2). Handle the duplicated cases (delete users' markings for this image, transfer the remaining markings to the image kept, delete the other image; see listing A.3)
7. Delete superfluous marking and image files on the file system.
8. Push to and tag in the Gitlab repository[8].

[6] https://gitlab.enterpriselab.ch/skinapp/Material/tree/skinappweb_run01_unclean_2016-03-02/skinappweb_run01_unclean_2016-03-02

[7] this is reported as a bug: https://gitlab.enterpriselab.ch/skinapp/skinappweb/issues/65

[8] https://gitlab.enterpriselab.ch/skinapp/Material/tree/skinappweb_run01_cleaned_2016-03-02/skinappweb_run01_cleaned_2016-03-02

Several improvements of SkinAppWeb might be considered for the future, many of them already recorded in the project's bug tracking system by different researchers at the D3S Institute. For example, ambiguities in determining whether a certain user has provided useful markings or which user is created for internal testing purposes only pose a challenge when choosing on which data to conduct experiments. Image comparison techniques could be implemented to prevent duplicate data from being entered into the database. The markings are stored on the file system as well as in the database; this data duplication should be avoided.

1.3.2.2 Setup Second Run

After mitigating the issues of duplicated images in the first run, it has been necessary to set up a second, including new images and experts. Since in the first run, not all the images provided by A. Navarini have been used, the second comprises these unused images. Several experts have been invited to participate in the first run, but not all of them provided markings. These experts have been contacted again to partake in the second run. Its starting position has been pushed to and tagged in the Gitlab repository, too[9].

1.3.3 Available Data

Disclaimer: This subsection 1.3.3 is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

The available image material for this thesis consists of a total of 48 hand photographies, of which 24 show front sides, and the other 24 show back sides of hands. For each photography, there are between 1 and 11 associated expert markings. These markings are combined to generate a consensus diagnosis in the form of an image that contains all pixels which more than 50% of the experts marked eczematous. Moreover, an image mask of the detected and manually reviewed hand area is available for each image. The hand photographies are colour pictures, whereas the consensus diagnosis and the hand area masks contain only black and white pixels. Tables 1.1 and 1.2 provide an overview of the available image material. The available data exposes an average imbalance in the ratio of positive to negative examples of about 1:8 for images of front sides of hands and 1:15 for images of back sides.

However, note that there is an imbalance between the images chosen for the hold-out and evaluation sets, as summarised in table 1.2 since the images with the least number of expert's markings are selected for the hold-out set. The reasoning behind this is that the most "valuable" images are considered to be the ones with the most markings since their consensus diagnosis should provide for a more reliable ground truth. The chosen images for the hold-out set are thus

[9] https://gitlab.enterpriselab.ch/skinapp/Material/tags/skinappweb_run02-prestart_2016-03-17

not representative of the whole population, at least with respect to the ratio of eczematous area to the entire skin area. However, whether this affects the machine learning process as a whole would have to be investigated more thoroughly than by only comparing these ratios.

Front sides				Back sides			
ID	Long ID	(*)	Size	ID	Long ID	(*)	Size
1012	Y00X956P	1	1088 × 957	1003	X906X88D	2	1432 × 1089
1002	X8XP96XP	2	1463 × 1088	1005	X985X76D	2	1540 × 1088
1014	Y0XPY50P	2	1089 × 914	1007	X9YXXX5D	2	1671 × 1089
1016	Y56PPP5P	2	1482 × 1089	1013	Y0XPY50D	2	1089 × 932
1022	Y7XX8PXP	2	1266 × 1090	1015	Y56PPP5D	2	1493 × 1089
230	X9786X6P	3	1858 × 1088	1017	Y5959X7D	2	1655 × 1089
1018	Y5959X7P	3	1647 × 1088	1001	X8XP96XD	3	1446 × 1088
1020	Y6XPX67P	3	1460 × 1089	1009	Y00009PD	3	1627 × 1089
234	YX00058P	3	1524 × 1088	1019	Y6XPX67D	3	1664 × 1089
225	YX8X8XPP	3	1624 × 1089	231	Y567PX9D	4	1880 × 1090
1024	YYPY067P	3	1586 × 1088	1021	Y7XX8PXD	4	1372 × 1089
1008	X9YXXX5P	4	1736 × 1089	1023	YYPY067D	4	1613 × 1090
1010	Y00009PP	4	1900 × 1089	229	X887X69D	6	1423 × 1089
243	Y5X0P99P	4	1524 × 1089	224	YX00058D	6	1543 × 1089
219	X856X80P	5	1431 × 1090	244	YX8X8XPD	6	1611 × 1088
1004	X906X88P	5	1373 × 1090	218	X6PP799D	7	1656 × 1090
232	Y58Y580P	5	1774 × 1089	238	X856X80D	7	1684 × 1089
236	YXP80P5P	5	1089 × 1089	240	Y0667P5D	7	1603 × 1088
228	X6PP799P	7	1744 × 1088	242	Y5PYX9PD	7	1545 × 1090
241	Y567PX9P	7	1797 × 1088	246	YY68Y75D	7	1630 × 1090
223	Y5PYX9PP	7	1592 × 1089	220	X9786X6D	8	1765 × 1090
227	YY68Y75P	7	1613 × 1089	222	Y58Y580D	8	1597 × 1089
221	Y0667P5P	9	1359 × 1089	233	Y5X0P99D	8	1602 × 1089
239	X887X69P	11	1478 × 1089	226	YXP80P5D	10	1290 × 1089

Table 1.1: List of the available images. (*) denotes the number of markings. The dashed line marks the border between the hold-out and the evaluation set (see section 3.4 for details).

Front sides				Back sides			
ID	Skin	Eczema	Ratio	ID	Skin	Eczema	Ratio
1012	643’092	18’903	2.94%	1003	724’434	53’741	7.42%
1002	864’490	14’881	1.72%	1005	618’816	6’547	1.06%
1014	465’614	34’485	7.41%	1007	868’411	60’658	6.98%
1016	905’164	0	0%	1013	497’044	23’524	4.73%
1022	823’455	0	0%	1015	794’253	0	0%
230	847’654	99’523	11.74%	1017	784’311	10’620	1.35%
1018	802’242	25’943	3.23%	1001	871’443	13’458	1.54%
1020	879’231	19’762	2.25%	1009	777’712	38’649	4.97%
234	822’076	85’091	10.35%	1019	927’639	10’631	1.15%
225	899’361	48’390	5.38%	231	788’658	41’872	5.31%
1024	877’829	217’211	24.74%	1021	964’730	15’098	1.56%
1008	1’008’328	15’140	1.5%	1023	913’979	3’091	0.34%
1010	822’969	157’145	19.09%	229	833’940	240’576	28.85%
243	689’171	17’534	2.54%	224	868’772	26’890	3.1%
219	714’010	0	0%	244	946’734	29’098	3.07%
1004	603’716	138’499	22.94%	218	862’108	1’000	0.12%
232	791’761	5’884	0.74%	238	760’415	22’786	3%
236	731’672	507’085	69.3%	240	793’148	91’723	11.56%
228	811’883	17’090	2.1%	242	791’008	249’393	31.53%
241	740’609	220’954	29.83%	246	872’281	20’547	2.36%
223	785’890	24’590	3.13%	220	847’768	44’748	5.28%
227	844’055	175’763	20.82%	222	720’165	3’705	0.51%
221	682’688	24’860	3.64%	233	743’286	25’866	3.48%
239	817’799	123’668	15.12%	226	744’208	183’863	24.71%
average whole set	786’448	83’017	10.86%	average whole set	804’803	50’754	6.31%
average hold-out	758’245	27’965	3.97%	average hold-out	714’545	25’848	3.590%
average evaluation	795’849	101’367	13.15%	average evaluation	834’889	59’055	7.358%

Table 1.2: List of images with their skin pixels and the number of eczema pixels according to experts’ consensus diagnoses.

1.3.4 SkinAppCore

1.3.4.1 Current State

Disclaimer: This subsection 1.3.4.1 is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

The term SkinAppCore denotes a collection of core libraries for the SkinApp CTI project. The files are organised in a Git repository maintained in the Enterprise Lab of the Lucerne University of Applied Sciences and Arts[10]. At the time of writing this thesis, the SkinAppCore library can be described as an accumulation of processing steps that have been gathered over past work (see section 2.1). These processing steps are contained in independent classes in terms of object-oriented programming. Since no complete processing workflow is implemented, the code base of SkinAppCore has more the character of a software library. However, the medium-term objective concerning the SkinApp CTI project is to compose several processing pipelines consisting of these processing steps. The version of SkinAppCore discussed in this thesis is not ready to provide for a complete pipeline. Missing functionality includes, for example, the reading and processing of multiple input files, the writing and reading of partial processing results into and from files or the reduction of learned features. Furthermore, a classification process is entirely missing. The architecture of the SkinAppCore library is illustrated in figure 1.2.

1.3.4.2 Addition of Previous Work

From previous work conducted in (Schnürle, 2016), some code has been transferred to the SkinAppCore code project. This transfer has been accompanied by several refactorings which provide for a better generalisation of the code, for compliance with coding guidelines in the SkinAppCore project and better separation of concerns as well as re-usability by dividing certain process steps into smaller and more coherent units of code. Additionally, several unit tests have been implemented to enable extensibility and to provide for a more robust code base.

[10] see `http://www.enterpriselab.ch/` (accessed 2016-06-21)

Figure 1.2: Class diagram of SkinAppCore at the beginning of this work. The members of the classes inheriting from FeatureStep (see boxes on the left) are not listed on the diagram. Some of these classes have additional member methods and variables to the inherited ones. This diagram aims to illustrate the pipeline character of the architecture rather than to show the processing details of individual classes. cv::Mat refers to OpenCV's matrix datatype, pt::ptree denotes a boost[11] property tree.

[11] see: http://www.boost.org/ (accessed 2016-06-09)

1.4 Organisation of this Thesis

In chapter 2, previous work, as well as related work, is discussed. Chapter 3 contains details about the approaches investigated in this thesis including a brief introduction to Support Vector Machines and the evaluation settings collectively shared with the two other master theses. Chapter 4 defines the experiments conducted and provides for a few intuitions concerning the metrics reported. The results of this thesis are presented in chapter 5, and finally, chapter 6 concludes this thesis and provides an outlook for future research.

1.5 Acknowledgements

First of all, I thank my advisor Prof. Dr. Ruedi Arnold for supporting and accompanying me throughout my master studies. Our discussions of all the aspects of my thesis and beyond along with uncountable thought-provoking impulses encouraged me constantly and enabled me to conduct the research and work culminating in this thesis. The same goes for many other researchers at the D3S Institute: Prof. Dr. Marc Pouly, Prof. Dr. Thomas Koller, and Dr. Tim vor der Brück. Without exception, they always found the time to discuss my current questions and provided me with innumerable inputs on possibly promising directions.

Similarly invaluable have been the countless discussions with other master students. Patrick Buchter and Manuel Brun conducted their master theses at the same time on the same project – SkinApp – as me, and as such, I very much like to remember our lively debates on many aspects of SkinApp, machine learning in general and beyond. Likewise, discussing various aspects of machine learning, the KNIME framework or SkinAppWeb with other master students and research associates at the D3S Institute provided me with new ideas; Daniel Pfäffli, Michael Handschuh and Roland Christen have all provided me with various insights most helpful.

Prof. Dr. Alexander Navarini, a dermatologist from the Department of Dermatology of the University Hospital Zurich in Switzerland, did not only provide for the photographies of hands vital to the whole SkinApp project but also with valuable medical insight on hand eczema.

The Enterprise Lab team was most helpful in providing resources, virtual as well as physical machines. Without this, much fewer experiments would have been possible with far less conclusive results. Bruno Joho, Sandro Bachmann, and Dario Lötscher have all been very supportive.

But most of all, words fail me to express my gratitude to my wife Béatrice, who supported me and encouraged me throughout my seemingly countless years of studies. They have accompanied us for longer than we both care to remember. Even if I might never fathom how exhausting and demanding this must have been for you, I thank you from the bottom of my heart for trusting in me, always. Never in my life will I forget how profoundly you believe in us and our love. It is done – here's to our future!

Chapter 2

Previous and Related Work

2.1 Previous Work

Disclaimer: This section 2.1, including subsections 2.1.1 and 2.1.2, is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

2.1.1 Statistical Learning

In (Suter, 2012, 2013; Suter et al., 2014), a supervised statistical learning approach developed by (Varma and Zisserman, 2005) has been implemented and evaluated for the problem of deciding whether certain skin regions are healthy or eczematous. The labels for the supervised approach have been provided by a dermatologist. First of all, an image *filter bank* is applied to the hands images. A certain number of clusters is obtained from the *filter responses*; the cluster centroids are called *Textons* and stored in a *Texton dictionary*. For each pixel of the hands images, the pixel's nearest neighbour in the Texton dictionary is searched with respect to the Euclidean distance between their corresponding filter response values. This procedure results in a *Texton map* for each hands image. The Texton maps are divided into patches of a certain size, and from each of these patches, the Texton frequency is obtained by calculating the histogram of a patch and normalising it to unity. These histograms are called the *models*.

For the *training* stage, a certain number of these models is calculated from the training hands images. Since the labels of these patches are known from the experts, the models represent either healthy or eczematous skin. In the *classification* stage, the models of the unseen hands images are created in the same manner as described above. Then, the χ^2 similarity between these models and all models created in the training stage is calculated. The label of the trained model most similar to the unseen model is assigned to the unseen model and therefore the pixel corresponding to this model is labelled. Since, for computational reasons, not every unseen pixel's model could be calculated, the result is interpolated to obtain a label for all pixels of the unseen hands images. For a more in-depth description, refer to (Suter, 2012, 2013; Suter et al., 2014) as well as (Schnürle, 2016).

(Suter et al., 2014) report an average accuracy of 78% with precision 35%, for front sides of hands only (since there were no images of back sides of hands available).

2.1.2 Support Vector Machine

In (Schnürle, 2016), a Support Vector Machine with a *Gaussian radial based function* as the kernel and *different error costs* for the two classes has been used to determine a decision boundary between healthy and eczematous skin texture. The features provided to the SVM are based on the same ideas as developed and implemented in (Suter, 2012, 2013; Suter et al., 2014; Varma and Zisserman, 2005). The same filter bank is applied to the hands images, Textons are created and stored in a Textons dictionary, the Texton map and finally the models are calculated. Instead of the similarity search, these models are provided as input to an SVM. From the perspective of the SVM, each histogram bin of the models thus represents a feature describing the model.

Multiple experts' labels could be obtained via a browser application called *SkinAppWeb* (see subsection 1.3.2), in which several experts participated. Thus, a *consensus diagnosis* for each pixel could be calculated as the final label for each model.

(Schnürle, 2016) reports an average F_1 score of 35% with accuracy 80.9% and precision 32.7% for images of the front side of hands and an average F_1 score of 25.7% with average accuracy 68.3% and precision 20.1% for back sides. The SVM optimised for the F_1 score.

2.2 Related Work

The literature on how to improve classification performance with SVMs is very rich. In this section, certain aspects relevant to this thesis and beyond are addressed. These aspects fall into the following categories:

- Features extracted from the available data (see subsection 1.3.3) and provided to the SVM.
- Pre- and post-processing of the data before or after the SVM is applied.
- Choice of the hyperparameters or different optimisation objectives for the SVM.
- Hybrid approaches.

Different metrics are reported in the following related work. See section 4.3 for their definition and a discussion of them.

The selected related work presented in this section was chosen mainly based on its applicability and assumed potential on this thesis' research questions (see section 1.2). Although not all of the following ideas have been investigated in this thesis, they show potentially promising approaches. Thus, they are presented here to provide a more comprehensive introduction nonetheless.

2.2.1 Different or Additional Features

The classification performance in previous work (Schnürle, 2016; Suter, 2012, 2013; Suter et al., 2014) could not be improved beyond a certain boundary (e.g., a precision higher than 35% could rarely be achieved). Thus, one approach to attacking the problem of improving the classification performance focuses on the choice of features different from Texton-based approaches.

Most of the following literature investigates similar questions as in this thesis, namely the detection of certain skin diseases from photographies of healthy and ill skin.

(Nisar et al., 2013) point out that for detecting skin lesions, different colour spaces of the images should be taken into consideration. They investigate the HSI space, CMY, YCbCr, and CIELab spaces. See subsection 3.1.1 for a brief introduction of the HSI and the CIELab spaces used in this thesis. For classification, they choose k-means clustering. They conclude that the H channel of the HSI colour space and the a* channel of the CIELab space provide the best results, with H providing for 71.9% sensitivity and 76.6% accuracy, and a* providing for 97.6% specificity and 94.8% precision.

(Al Abbadi et al., 2010) investigate the problem of Psoriasis detection on images, a problem similar to detecting hand eczema. They rely on the extraction of *colour moments* (see subsection 3.1.4) on the RGB channels of the image and on texture features calculated from a *grey level co-occurrence matrix (GLCM)* (see subsection 3.1.2). They utilise a neural network as a classifier. They report results for very few image instances, which they classify with 100% accuracy.

In the literature, there are many examples to be found which use features calculated from a GLCM in combination with skin diseases. (Arifin et al., 2012) extract colour gradients, colour moments and features from the GLCM among other features together with patient history and provide them to a neural network. For some skin diseases, they report an average accuracy of 96% for diseased skin detection and an average accuracy of 94% for disease identification.

There is other research utilising GLCMs that does not focus specifically on skin diseases. (Gebejes and Huertas, 2013) explore the effect of choosing different values for the displacement parameter of the GLCM. They calculate the GLCM not on the grey channel, but on the L* channel of the CIELab colour space since "several works stated that the majority of texture information is located on this channel".

Aside from GLCMs, there are many other approaches to capture texture on images. (Das et al., 2013) compare the performance of an SVM with features from different sources, namely Local Binary Patterns, GLCM, Discrete Cosine Transform and Discrete Fourier Transform. They report the best accuracy of 89.7% using LBP features, followed closely by features from the GLCM with an accuracy of 88.5%.

(Meenalosini et al., 2012) use SVMs for malignancy detection concerning breast cancer with the help of mammography images. After a certain number of pre-processing steps like artefact removal, contrast enhancement, and edge detection via the *Sobel operator*, certain features from the GLCM of the edge detection results are extracted. Additionally, they use features from Local Binary Patterns. They report a sensitivity of 98.8% and a specificity of 97.4%.

In this thesis, the findings about colour spaces, colour moments and GLCMs on different channels of different colour spaces are applied to the problem of detecting hand eczema.

2.2.2 Imbalanced Learning Problem

As introduced in subsection 1.3.3, the available data is imbalanced with a ratio of 1:8 or 1:15, respectively. Hence, implications and mitigation strategies concerning that fact are investigated in this thesis.

When optimising solely for accuracy in the presence of imbalanced data, the *Accuracy Paradox* (Zhu and Davidson, 2007) might apply such that models reporting less accuracy on training examples would have yielded better performance on previously unseen examples. This observation seems reasonable, since e.g. if there are only a few positive examples, a simple majority-class classifier would reach a very high accuracy (Ben-Hur and Weston, 2010). This issue is illustrated in more detail with the baseline classifiers discussed in section 4.4.

The challenge of learning from imbalanced data can be approached in many different ways. (He and Garcia, 2009) provide an excellent overview of the nature of the problem, of various strategies to overcome the difficulties as well as a discussion of metrics more meaningful in the presence of imbalanced data than accuracy. In their analysis of the nature of the problem, they suggest that "data set complexity is the primary determining factor of classification deterioration, which, in turn, is amplified by the addition of a relative imbalance". *Data complexity* relates to the presence of overlapping classes, noise, etc. They differentiate between different types of imbalances; those potentially relevant to this thesis are:

1. **Between-class imbalance**
   Data exhibiting an unequal distribution between its classes, one class severely outrepresenting others. Disregarding this may lead to accuracies of near 100% for the majority class, while the minority classes show accuracies of only 0-10%.
2. **Intrinsic imbalances**
   In this case, the imbalance is a direct result of the nature of the data space. Between-class imbalance, in turn, is an example of intrinsic imbalance.
3. **Extrinsic imbalances**
   Here, the imbalance is rooted in variable factors, such as time. It is possible that a sample of data is drawn at an unfavourable time, such that the acquired, imbalanced data truly attains from a balanced data space.
4. **Relative imbalance**
   With an increasing number of examples, the imbalance between the classes remains the same. Thus, the minority class is not necessarily rare in its own right but rather relative to the majority class.
5. **Within-class imbalance**
   If subconcepts exist within a class (i.e., the class is heterogeneous), these might be dominated by the largest cluster and not learned well.

A possible strategy to mitigate the *imbalanced learning problem* which is easily implemented is *sampling* the available data. Either the minority class can be *oversampled*, or the majority class *undersampled*. While balancing the ratio of the positive to the negative examples, which is in principal beneficial to many machine learning algorithms, both strategies exhibit potentially serious drawbacks: By undersampling the data, information is lost. By oversampling the data, the original data distribution might be diluted. The simplest approach for sampling is to choose randomly which examples to remove or to duplicate, respectively.

Algorithms exist to perform a more informed undersampling with the goal of reducing information loss by retaining informative examples and removing redundant examples. (Liu et al., 2009) develop an ensemble that trains several classifiers via AdaBoost as discussed in (Schapire, 1999), which "explores the majority class data by using independent random sampling with replacement" (He and Garcia, 2009).

(a) Imbalanced data, ideal and learned hyperplane *(b) Undersampled majority class with hyperplanes*

Figure 2.1: Hyperplanes on imbalanced data: Undersampling[12]

(a) Application of different error costs *(b) Application of DEC and SMOTE*

Figure 2.2: Hyperplanes on imbalanced data: Different error costs (DEC) and SMOTE[13]

For oversampling, synthetic sampling with data generation is a way to provide for more interesting oversampled examples. (Chawla et al., 2002) develop an algorithm called *Synthetic Minority Oversampling Technique* (SMOTE) that creates new data examples between two adjacent examples of the minority class, see subsection 3.2.1. This procedure improves learning (He and Garcia, 2009), but for certain data distributions, SMOTE might amplify existing problems such as outliers or noise, as (Batuwita and Palade, 2013) point out. (Cao and Wang, 2011) argue that SMOTE "ignores data distribution and density information". (Wu and Chang, 2003) recognise the problem, too: "It assumes the neighborhood of a positive instance to be still positive, and the instances between two positive instances positive. Assumptions like these, however, can be data-dependent". Several approaches exist to mitigate these issues (He and Garcia, 2009).

(Akbani et al., 2004) discuss the problem of imbalanced data and the undersampling approach using an SVM. They identify two main challenges: The separating hyperplane might be skewed towards the minority class, and the orientation of the hyperplane might be incorrect. They randomly created test datasets with an ideal boundary. Consider figure 2.1a: Although the positive class is represented by fewer examples, the orientation of the learned hyperplane is close to the ideal hyperplane. However, the hyperplane is too close to the minority class.

[12]source: (Akbani et al., 2004)
[13]source: (Akbani et al., 2004)

After undersampling the majority class, they present figure 2.1b: The distance between the classes is about the same, but the orientation of the learned hyperplane is considerably different to the ideal hyperplane.

As a solution to this problem, they suggest the application of *different error costs (DEC)* (see subsection 3.3.5) as well as oversampling by SMOTE. Figure 2.2a shows that by application of different error costs, the learned hyperplane is pushed towards the ideal hyperplane, although its shape may not correspond well to the ideal hyperplane. Figure 2.2b finally shows the learned hyperplane after oversampling examples of the minority class by SMOTE, which leads to a more well-defined shape of the learned hyperplane.

(Akbani et al., 2004) compare their approach SVM+SMOTE+DEC with the application of each of the strategies alone. Concerning the average of the G mean of different datasets, they report a value of 92.3%. This corresponds to a performance gain of 38.5 percentage points compared to a conventional SVM without DEC (performing on average at 53.8% G mean), and a performance gain of 4.2 percentage points compared to undersampling (which performed on average at 88% G mean).

A complementary approach to sampling that (He and Garcia, 2009) discuss is the application of data cleaning techniques such as removing so-called *Tomek links* as described in (Tomek, 1976). If a data example's nearest neighbour is of another class, this pair is considered a Tomek link (see subsection 3.2.1).

(Sain and Purnami, 2015) compare the application of SMOTE, Tomek links removal and both strategies combined to the unmodified data using an SVM without DEC and an RBF kernel. For their combined approach, they report an F_1 score from 97.7% up to 99.8% depending on the dataset, which corresponds to an improvement in performance from 29.7 to 52 percentage points against the unmodified data (which achieved an F_1 score from 45.7% to 70%). SMOTE without Tomek links removal performed only a few percentage points worse than their combined approach.

SVM-based active learning is another strategy to overcome the imbalanced learning problem. Usually, "active learning methods are used to solve problems related to unlabeled training data", as (He and Garcia, 2009) observe. Nevertheless, research shows that this approach identifies the examples closest to the hyperplane which are the most informative examples and retrains the SVM on them.

In this thesis, some mitigation strategies mentioned above are applied to the problem of detecting hand eczema. SMOTE without any extensions is used for oversampling. DEC are applied to both classes, as it was already the case in (Schnürle, 2016). Removal of Tomek links is applied, too. Informed undersampling and SVM-based active learning are not investigated.

2.2.3 Ensembles of SVMs

Figure 2.3: Ensemble of separately trained SVM component classifiers and final classification[14]

Several publications report that by querying an ensemble of SVMs instead of a single SVM, prediction performance could be improved, see (Batuwita and Palade, 2013; He and Garcia, 2009; Liu et al., 2006). Multiple SVMs are trained on a different subset of the original data. Then, all unseen examples are predicted by every SVM and afterwards a *majority voting* rules the final prediction.

(Wang and Japkowicz, 2010) use *boosting* in combination with SVM component classifiers to elicit a final classification by voting; this is shown in figure 2.3. They compare their ensemble to various other SVM-based approaches and report an average of the G means of different datasets of 77%, corresponding to a performance improvement of between 3.69 (versus a method called *Databoost-IM* focussing on examples hard to classify) and 13.1 percentage points (versus undersampling with SVMs). SVMs with different error costs and SMOTE perform almost as good as Databoost-IM.

Many combinations of the discussed methods are possible. (Liu et al., 2006) show one example of this; their architecture is shown in figure 2.4. They compare their approach with several other methods such as SVM, SVM+SMOTE, or *random forests*. Concerning the F_1 score, they report values between 54.2% and 95.6%, depending on the dataset. This corresponds to an improvement of between 1.4 to 14.5 percentage points compared to the average performance of several different SVMs.

In this thesis, the performance of an ensemble of SVMs (see section 3.5), as well as some hybrid approaches, is investigated.

[14]source: (Wang and Japkowicz, 2010)
[15]source: (Liu et al., 2006)

Figure 2.4: Hybrid approach consisting of data cleaning, SMOTE, bootstrapping and an SVM ensemble[15]

2.2.4 Probabilistic SVM Outputs, Fuzzy SVMs

(Platt, 2000) describes a way how to gain not only class labels from the SVM but also probabilities for the class membership of the predicted data examples. The SVM is trained as usual. Additionally, a sigmoid function is trained to "map the SVM outputs into probabilities". (Platt, 2000) reports "comparable quality to the regularised maximum likelihood Kernel method, while still retaining the sparseness of the SVM". This approach is incorporated into the popular LIBSVM implementation, see (Chang and Lin, 2011).

(Batuwita and Palade, 2013) discuss the benefits of *Fuzzy SVMs (FSVM)* which "handle the problem of class imbalance together with the problem of outliers and noise". (Lin and Wang, 2002) provide further insight into an FSVM. The idea of different error costs for different classes is generalised to assigning a different weight, or error cost, for each data example according to its importance. Thus, "the FSVM algorithm can find a more robust separating hyperplane through maximising the margin by allowing some misclassification for less important examples, like the outliers and noise". In the setting of SkinApp, since multiple experts are labelling the same hands image, the ratio of experts classifying an example as eczematous could be considered for the individual weights. In LIBSVM, this approach is not directly implemented, but available as an extension, see (Chang and Lin, 2011).

In this thesis, these approaches have not been investigated, nevertheless remain promising for future work.

2.2.5 Optimising Directly for F1 Score

(Joachims, 2005) proposes a generalised SVM formulation which can directly optimise for many non-linear performance measures like the F_1 score. However, he suggests to only use the linear kernel, since using non-linear kernels will be "painfully slow"[16]. Interestingly, "when using the error rate as the loss function, the conventional SVM arises as a special case" of his formulation.

(Joachims, 2005) compares the performance of this generalised SVM with a conventional SVM with different error costs as e.g. used by (Morik et al., 1999). He suggests that this provides a "strong baseline to compare against". Concerning the F_1 score and different datasets, (Joachims, 2005) reports between 56.8% and 92.5% depending on the dataset. This shows an improvement of between 1.0 and 7.2 percentage points.

In this thesis, this generalised SVM formulation is used in a few experiments.

2.2.6 Metaheuristics and Feature Selection

When applying *metaheuristics* such as *simulated annealing* or *genetic algorithms*, the literature often combines the search for an SVM's hyperparameters with the task of *feature selection*. The importance of feature selection is discussed by (Chapelle et al., 2002; Chen and Lin, 2006; Lei and Huan, 2003; Weston et al., 2001). Various strategies exist to reduce the number of features, such as investigating *correlation coefficients* or applying *principal component analysis*. They all aim to "Improve generalisation error, Determine the relevant features (for explanatory purposes), Reduce the dimensionality of the input space" according to (Chapelle et al., 2002). As (Frohlich et al., 2003) observes, "the choice of the feature subset has an influence on the appropriate kernel parameters and vice versa". Hence, optimising both aspects simultaneously seems attractive.

(Gaspar et al., 2012) investigate simulated annealing. They state that it "is well established that features have a large influence on how well data can be separated into two classes, depending largely on the correlation of each feature to its class [...] Moreover, not all features play a positive role, and some might even contribute negatively to the classification process". Concerning the F_1 score, they report an average improvement over various datasets of between 4.0 and 23.2 percentage points. In one setting, they report a degradation of 11 percentage points.

(Frohlich et al., 2003) report an error rate with various genetic algorithms similar to when no feature selection is performed, but the reduction in features provides for a higher computational feasibility. (Huang and Wang, 2006) compare their genetic algorithm approach with the conventional grid search for an SVM's hyperparameters. Aside from a substantial decrease in the number of features needed, for different datasets, they report an area under ROC in between 0.77 and 0.976, which shows an increase in performance of between 0.02 and 0.16.

[16] see `https://www.cs.cornell.edu/people/tj/svm_light/svm_perf.html` (accessed 2016-06-09)

In this thesis, those metaheuristics have not been investigated, but may well prove to be beneficial to future work. Nevertheless, the idea of feature selection based on correlation coefficients is investigated, see subsection 3.2.3.

2.2.7 Combination of Convolutional Neural Networks with SVMs

Figure 2.5: Convolutional Neural Network with two convolutional and two fully connected layers[17]

Figure 2.6: Hybrid approach combining CNN with and SVM replacing the output layer[18]

The approach to combine *Convolutional Neural Networks (CNNs)* with SVMs is comparatively new, as (Niu and Suen, 2012) point out. They propose to combine the strengths of both classifiers such that the "CNN works as a trainable feature extractor and SVM performs as a recogniser". Figure 2.5 shows a conventional CNN, whereas figure 2.6 displays their proposed architecture. They compare their approach on handwritten digits with the best-performing algorithms to date for this task and find that they can lower the error rate to as low as 0.19%, instead of a 0.35% by the previously best algorithm based on a 6-layer neural network.

(Tang, 2013) replaces the *softmax layer* of a CNN with an SVM and reports a decrease in the test error for facial expression recognition from 0.99% for CNN+softmax vs. 0.87% for their approach, and 14% for CNN+softmax vs. 11.9% for their approach for other datasets.

In this thesis, combining CNNs with SVMs has not been investigated. If CNNs provide for sufficient qualification results, the addition of an SVM might prove beneficial.

[17] source: (Niu and Suen, 2012)
[18] source: (Niu and Suen, 2012)

Chapter 3

Improving Classification Quality

This chapter aims to provide insight into the different aspects of the approaches investigated in this thesis to improve the classification quality of SVMs. As illustrated in section 2.2, there is a wide variety of possibilities to attack the problem. Not all of them could be explored in this thesis; nevertheless, amongst those implemented, there are several approaches which yield a substantial improvement in quality.

The decision on which of the approaches introduced in section 2.2 should be investigated in this thesis is based on the assumed potential for this thesis' research questions (see section 1.2). However, to be able to examine multiple approaches exploratively, the presumed degree of difficulty to integrate each approach is considered, too. For example, the metaheuristics algorithms discussed in subsection 2.2.6 might promise great benefits in finding optimal hyperparameters as well as an optimal subset of features; however, they are estimated to take a longer time to implement than other, similarly promising approaches, and thus might be investigated in future work.

3.1 Features

As briefly reviewed in section 2.1, most of the research conducted until now relies on the assumption that the methodology developed by (Varma and Zisserman, 2005) applies to the specific problem of classifying hand eczema, as well. The model histograms of the Texton frequencies remain as a set of features in this thesis. However, since previous work could not improve classification quality beyond a certain level, additional features are considered.

(a) Original image *(b) Consensus diagnosis highlighted*

Figure 3.1: Exemplary image #239

(a) Magnified section of the right hand *(b) Consensus diagnosis highlighted*

Figure 3.2: Exemplary image #239

Figure 3.1 shows an example of a hands image with its consensus diagnosis highlighted. To the naked, untrained eye, it seems hard to assess the different skin textures, since the distinction between healthy and eczematous skin is often quite subtle, as shown in figure 3.2. Various strategies are pursued to investigate which features might capture the differences between healthy and eczematous skin better than the hitherto pursued Texton approach.

For a discussion of the Texton approach, refer to section 2.1 or (Schnürle, 2016), where this is discussed in great detail, as well as the foundations of this thesis as developed in (Suter, 2012, 2013; Suter et al., 2014; Varma and Zisserman, 2005). Figure 3.3 provides a brief overview of the process of Textons and models acquisition. Note that the first step, the labels acquisition, refers to the calculation of the consensus diagnosis used throughout this thesis. The consensus diagnosis is calculated by combining the available experts' markings and deciding on each pixel's label by a majority voting of experts for that pixel.

(a) Labels acquisition

(b) Filter bank application

(c) Textons acquisition

(d) Models acquisition

Figure 3.3: Texton approach developed in previous work

As observed in (Schnürle, 2016), from "domain knowledge, eczema should expose a higher level in redness than healthy skin", and thus colour information should increase classification performance. However, the approach pursued by (Suter, 2013), the so-called *VZ-dipoles* proposed by (Burghouts and Geusebroek, 2006), did not culminate in improved results. Thus, alternative approaches are followed in this thesis and introduced in this section.

All of the following approaches are applied on the 41×41 pixel neighbourhood of a pixel in question. However, on the image's borders or on the edges of the skin regions, when a 41×41 pixel patch would contain background regions, only pixels with skin are considered. In such cases, if e.g. normalisation is required, the information is normalised by dividing by the number of actual skin pixels instead of $41 \cdot 41 = 1'681$.

3.1.1 Colour Spaces

Many of the approaches discussed in this section aim to exploit colour information to various extents. Hence, in this subsection, the used colour spaces are briefly introduced.

3.1.1.1 RGB Colour Space

(a) red channel *(b) green channel* *(c) blue channel*

Figure 3.4: RGB channels of image #239

Figure 3.4 shows the three channels of the RGB space separately. Unsurprisingly, the red channel exposes higher values than the other two channels, since the photographies of hands are all taken from Caucasian people with pink-coloured skin.

The available hands images are stored as RGB (red, green, and blue) images, which can be converted to other colour spaces. The RGB space is additive: adding the three base colours red, green, and blue together results in the spectrum of the mixture (Poynton, 2012). However, this model suffers from some shortcomings. (Sangwine and Horne, 1998) list the following:

- highly correlated components since each one incorporates brightness: (Sangwine and Horne, 1998) report a correlation of "0.78 for B-R, 0.98 for R-G and 0.94 for G-B components".
- psychologically non-intuitive: it is "hard to visualize a colour based on R, G, B components".
- non-uniformity: the perceived differences between colours are "impossible to evaluate [...] on the basis of distance in RGB space".

Since features extracted from images for the subsequent machine learning algorithm should be predictive of their example's class, it seems advisable that highly correlated features should be avoided.

Psychological aspects should not be discarded prematurely for this thesis, since the expert's markings are ultimately based on their perception of the images of hands they are presented, and thus a disparity between their perception and the features extracted might prove detrimental.

3.1.1.2 HSI Colour Space

(a) hue channel *(b) saturation channel* *(c) intensity channel*

Figure 3.5: HSI channels of image #239

Figure 3.6: HSI colour model[19]

Figure 3.5 shows the three dimensions of images converted to the HSI colour space. The HSI space consists of the dimensions *hue*, *saturation*, and *intensity*, see figure 3.6 for a visualisation of these terms.

[19] source: http://www.had2know.com/technology/hsi-rgb-color-converter-equations.html (accessed 2016-06-19)

This colour space is "the closest to how human eye perceive a colour" (Nisar et al., 2013). The dominant perceived colour is captured by the hue dimension, saturation is a measure of "relative impurity of a colour" and intensity measures brightness, hence is similar to images converted to greyscale.

Figure 3.7: HSI hue channel rotated by 180°

For this thesis, the hue channel is rotated by 180° when incorporated in certain features, such that the red colours are no longer located near 0° or 360°. The reasoning behind this transformation is that, as stated at the beginning of this section, eczema should show a higher level of red colour. Since the hue channel exposes a circular range rather than a range of both minimal and maximal values, it is not linearly separable. Moreover, the discontinuity at 0° is just at the red colour which is assumed to be of more importance than the other colours. Thus, when converting to features, a red colour could both be represented by the minimum and the maximum of the feature's range. Hence, to mitigate this issue, the hue channel is rotated by 180°.

There are similar colour spaces like HSI, e.g. HSV and HSL. Note that while agreeing on the definition of hue, they differ in the calculation of saturation and the third dimension (intensity in HSI, value in HSV, or lightness in HSL).

3.1.1.3 CIELab Colour Space

(a) L^* channel (b) a^* channel (c) b^* channel

Figure 3.8: CIELab channels of image #239

Figure 3.9: CIELab colour model[20]

Figure 3.8 shows the three dimensions of images converted to the CIELab colour space. The CIELab space consists of the dimensions L^*, a^*, and b^*, see figure 3.9 for a visualisation.

The CIELab colour space "is subdivided into a luminance axis and two chrominance axes", whereas chrominance "is the difference between a colour and a chosen reference colour of same luminance" (Nisar et al., 2013). L* represents lightness, a* the degree of green to red, and b* the degree of blue to yellow. Positive a* correspond to a red colour, while negative values represent green colours. Positive b* correspond to a yellow colour, while negative values represent blue colours. CIELab is a "perceptually uniform" colour space, meaning that "a small perturbation to a component value is approximately equally perceptible across the range of that value", which is not the case for e.g. the RGB space (Vezhnevets et al., 2003). A colour space in unison with the perception of experts highlighting eczematous regions on images should provide for more meaningful features, as already pointed out in subsection 3.1.1.1.

[20]source: http://www.mdpi.com/1424-8220/14/7/11943/htm (accessed 2016-06-19)

3.1.2 Grey Level Co-Occurrence Matrix of Pixels' Neighbourhood

3.1.2.1 Grey Level Co-Occurrence Matrix (GLCM)

(a) RGB converted to greyscale *(b) CIELab L* channel* *(c) Sobel operator on greyscale image* *(d) Equalised histogram with Sobel operator*

Figure 3.10: Conversions and filters

Several transformations have been applied to the hands images, a few of them are shown in figure 3.10. The red square shows an eczematous region, the green squares two healthy regions (smooth skin and skin with hand lines), and their magnification.

The original coloured image has been converted to a greyscale image (see figure 3.10a), or to the CIELab colour space and only the L* channel capturing the lightness has been used (see figure 3.10b). Although both channels are very similar since both capture how bright the images are, in the literature, both channels are used (see subsection 2.2.1), and thus both have been chosen to investigate whether one of them provides for better classification quality. Figure 3.10c shows the results of converting the image to grey scale and afterwards applying a horizontal and a vertical *Sobel operator* and calculating the magnitude of the gradient at each pixel. Finally, figure 3.10d represents the result after performing a histogram equalisation on the original hands image, then converting it to a grey scale image and applying the Sobel operator in the same manner as described before.

In these examples, the responses from the Sobel operator seem to capture the differences between skin textures quite well. Although seemingly promising, it has to be investigated thoroughly whether this observation prevails when applying the same processing to other images and skin regions, and whether this leads to better results in the actual classification problem. See chapter 5 for the results on this.

These observations led to conducting research on how the differences between the skin regions could be captured best, which revealed the exploitation of *grey level co-occurrence matrices (GLCMs)* in the literature, see subsection 2.2.1.

Figure 3.11: A sample image with the corresponding grey level co-occurrence matrix (GLCM) in horizontal direction with displacement = 1[21]

The literature is rich in successfully extracting texture information using a GLCM, among others in the field of recognising and classifying skin diseases. The GLCM was introduced by (Haralick et al., 1973) to provide for "easily computable textural features based on graytone spatial dependancies". As (Parekh and Mittra, 2011) put it, "GLCM defines the probability of one grey tone occurring in the neighborhood of another grey tone at a specified distance and along a specified direction". (Haralick et al., 1973) propose to capture the texture of a single-channel image by a matrix calculated from said image. This matrix describes the magnitude of neighbouring pixel's difference concerning their values.

Figure 3.11 shows an example of this process. On the left-hand side, an image with its corresponding grey level values is displayed. For each pixel, the pixel's neighbour directly to its right is examined; in the example image, this is shown for a pixel with value 8 on the last row. The value of its neighbour to the right is 7.

On the right-hand side, the constructed GLCM is shown. The GLCM is a square matrix of the same dimension as the possible range of grey levels that can occur in the original image – in this example, a range from 0 to 10 is possible. For a conventional 8-bit greyscale image, this would normally range from 0 to 255.

Now, the neighbouring pixels' values have been determined to be 8 and 7, respectively. Thus, the matrix' component at position $(7, 8)$ is incremented by 1. When the process is finished for each pixel, the GLCM contains the number of occurrences of each difference in the original image's pixels' values.

[21] source: http://jtd.amegroups.com/article/view/2121/html (accessed 2016-06-09)

Figure 3.12: Nearest neighbours of a pixel in different directions.[22]

It is evident that for images exposing different textures and therefore different distributions of greyscale values, distinct GLCMs would be constructed. For example, if an image were very smooth and therefore the values of all pixels would be the same value x, the GLCM would be a zero matrix but for the single component at (x, x), which would accumulate all increments. If the gradient of pixels for another example were not zero but very small nonetheless, mainly the components of the matrix' diagonal would be populated, while grey value combinations further away from the diagonal would be zero.

The following three generalisations of this process are common:

- Not only examining the pixel directly to the right of a pixel, but also to the remaining other 7 directions, thus calculating a GLCM for each direction as displayed in figure 3.12. Most of the time, only four GLCMs are constructed for 0, 45, 90, and 135 degrees. Each opposite direction is incorporated by adding the GLCM's transpose matrix to the GLCM itself, resulting in a symmetric matrix representing the occurrences in both directions.
- Applying different displacements. In the example in figure 3.11, the GLCM is computed with a displacement of 1, which is synonymous with the directly adjacent pixel. Larger displacements are possible to capture features which manifest more prominently in larger areas.
- Not only an image converted to greyscale can be used in this process, but also any other single-channel image that captures texture features. For example, (Gebejes and Huertas, 2013) use the L* channel of the CIELab colour space.

The GLCM is normalised to unity by dividing by the sum of the matrix' components. This yields the frequency of each possible occurrence of adjacent grey levels on the greyscale image. From the four GLCMs of different directions, the arithmetic mean is calculated, resulting in a single rotation invariant GLCM, as suggested by (Haralick et al., 1973).

[22] source: (Haralick et al., 1973)

(a) RGB converted to greyscale *(b) CIELab L* channel* *(c) Sobel operator on greyscale* *(d) Equalised histogram with Sobel operator*

Figure 3.13: GLCM eczema patch

(a) RGB converted to greyscale *(b) CIELab L* channel* *(c) Sobel operator on greyscale* *(d) Equalised histogram with Sobel operator*

Figure 3.14: GLCM "lower" healthy patch

(a) RGB converted to greyscale *(b) CIELab L* channel* *(c) Sobel operator on greyscale* *(d) Equalised histogram with Sobel operator*

Figure 3.15: GLCM "upper" healthy patch (with hand lines)

Figures 3.13 to 3.15 show the 256×256 GLCMs of the three patches from the four images shown in figure 3.10. For illustrative purposes, their histograms have been equalised.

From these illustrations, it seems justified to expect that features from the GLCM (see subsection 3.1.2.2) should provide for a separability of the two classes. For the grey and L* channel, the GLCM of the eczematous patch exposes more fringed borders than both healthy regions. Grey and L* channels are very similar, but not identical. The GLCMs on the two columns on the right-hand side show differences, too, but the upper GLCMs seem similar to the lower GLCMs. This suggests that the difference between eczematous skin and skin with hand lines might not be captured well enough by applying the Sobel operator. Obviously, this analysis is not representative of all kind of different eczematous and healthy regions displayed in the available images. Nevertheless, the extraction of features from the GLCMs and providing them to the SVM did manifest in improved classification quality.

3.1.2.2 Features Extracted From the GLCM

From the constructed, normalised GLCM, a rich number of features can be calculated and interpreted as a texture specific feature of an image. (Haralick et al., 1973) propose a set of 14 features which are based on certain consolidations calculated from the GLCM. In most of the literature, as well as in this thesis, the features *entropy, energy, homogeneity, contrast, and dissimilarity* are considered, as defined in equations (3.1) to (3.5) ($p(i,j) \in \mathbb{R}$ denoting the $(i,j)^{\text{th}}$ component in the GLCM):

- **Entropy:**

$$-\sum_{i,j} p(i,j)\log(p(i,j)) \tag{3.1}$$

  Entropy aims to capture spatial disorder. Random distributions would gain high entropy, whereas solid tone images result in zero entropy. (Gebejes and Huertas, 2013).

- **Energy:**

$$\sum_{i,j} p(i,j)^2 \tag{3.2}$$

  Energy is presented as "Angular Second Moment" in (Haralick et al., 1973) as a measure of homogeneity, thus constituting the opposite of entropy. More homogeneous images correspond to high energy, constant images expose an energy of 1 (Gebejes and Huertas, 2013).

- **Homogeneity:**

$$\sum_{i,j} \frac{p(i,j)}{1+|i-j|} \tag{3.3}$$

  A variant of equation (3.3) is presented as "Inverse Difference Moment" in (Haralick et al., 1973): $\sum_{i,j} \frac{p(i,j)}{1+(i-j)^2}$. The literature is ambiguous in which formulation to use; in this thesis, equation (3.3) is calculated. Homogeneity "measures the uniformity of the non-zero entries" and is equal to 1 when the image exposes no variation (Gebejes and Huertas, 2013).

- **Contrast:**

$$\sum_{i,j} (i-j)^2 p(i,j) \tag{3.4}$$

  Contrast captures the variation in the GLCM. If neighbouring pixels are similar in their value, contrast is low. Constant images correspond to a contrast of zero (Gebejes and Huertas, 2013).

- **Dissimilarity:**

$$\sum_{i,j} |i-j| p(i,j) \tag{3.5}$$

  Dissimilarity measures variation of grey level pairs, similarly to contrast, but contrast grows quadratically. Constant images expose a dissimilarity of zero (Gebejes and Huertas, 2013).

	eczema patch	lower healthy patch	upper healthy patch
Entropy	6.603	4.922	6.365
Energy	0.002	0.011	0.002
Homogeneity	0.307	0.46	0.368
Contrast	56.621	6.929	19.855
Dissimilarity	5.149	1.983	3.301

Table 3.1: GLCM values calculated for example regions in figure 3.10a (greyscale)

	eczema patch	lower healthy patch	upper healthy patch
Entropy	6.532	4.809	6.261
Energy	0.002	0.012	0.003
Homogeneity	0.316	0.47	0.376
Contrast	52.064	6.258	17.708
Dissimilarity	4.933	1.883	3.129

Table 3.2: GLCM values calculated for example regions in figure 3.10b (CIELAB L channel)*

	eczema patch	lower healthy patch	upper healthy patch
Entropy	7.793	6.246	7.232
Energy	0.001	0.003	0.001
Homogeneity	0.158	0.264	0.198
Contrast	461.67	54.052	210.49
Dissimilarity	14.773	5.644	10.318

Table 3.3: GLCM values calculated for example regions in figure 3.10c (Sobel)

	eczema patch	lower healthy patch	upper healthy patch
Entropy	9.006	8.31	8.672
Energy	0	0	0
Homogeneity	0.075	0.117	0.109
Contrast	4240.2	651.25	1912.9
Dissimilarity	48.153	19.455	29.627

Table 3.4: GLCM values calculated for example regions in figure 3.10d (Sobel on equalised histogram)

Reconsider figure 3.10. Tables 3.1 to 3.4 show the values for the highlighted regions on each of the four images. The difference between the two healthy patches is that the "upper healthy patch" contains hand lines. If there is a notable difference between the two classes, this might indicate that those features are predictive of the texture's class and therefore might provide useful features for the SVM. For the greyscale examples, contrast and dissimilarity promise to differ the most between the two classes. In the L* channel of the CIELab colour space, the examples expose only a small difference. For the images after applying the Sobel operator, the most noticeable difference lies in the contrast value.

3.1.3 Colour Sensitive Extensions to the Model Histograms of Textons

This approach was already addressed as a possible variation in (Schnürle, 2016) and is easily integrated into previous work's code.

Figure 3.16: MR8 filter bank[23]

[23]source: (Varma and Zisserman, 2005)

The Texton histogram approach is based on the application of the MR8 filter bank shown in figure 3.16 on a hands image converted to greyscale. (Varma and Zisserman, 2005) describe these as "2 anisotropic filters (an edge and a bar filter, at 6 orientations and 3 scales), and 2 rotationally symmetric ones (a Gaussian and a Laplacian of Gaussian)". By reducing the anisotropic filter responses to their maximum response within each group of 6 filters, for each pixel, 8 filter responses are obtained. The extension to this is to simply incorporate the values for the three colour channels of each pixel as additional feature responses, such that for each pixel, 11 filter responses are calculated. Additionally, the responses are normalised to the interval $[0, 1]$. The normalisation was not applied in previous work since the range of the filter responses remained similar. However, the additional colour channel values exposed a wider range of values. Thus, without normalisation, finding the cluster centroids in the subsequent step would be dominated by the colour values, since the clustering algorithm (kmeans++, see (Arthur and Vassilvitskii, 2007)) relies on the calculation of the Euclidean distance. All subsequent steps in the process to finally calculate the model histograms remain the same.

Since it is not a priori clear which channels of which colour space captures the properties of healthy and eczematous skin best, the HSI space and the CIELab space have been chosen. Following (Nisar et al., 2013), those spaces might provide for best classification results, since they report their best results using the H (hue) channel of the HSI colour space and the a* (degree of green to red) channel of the CIELab colour space. Either the 3 HSI channels or the 3 CIELab channels are added as filter responses as described above, different experiments use different settings, see section 4.2.

3.1.4 Colour Moments of Pixels' Neighbourhood

Another approach than discussed in subsection 3.1.3 is to incorporate colour information as additional features which are provided to the SVM separately from model histograms. From the perspective of the SVM, the model histograms' bins are independent of each other. Each bin is a feature describing the pixel in question. Thus, additional features can be extracted from the pixel or its neighbourhood and added without difficulties to the existing model histogram features, or even be used to replace the model histogram features as a whole.

The calculation of certain *colour moments* can be considered, as (Al Abbadi et al., 2010) report, to capture the colour information of the whole pixel's neighbourhood. For a single channel, they calculate the mean, the standard deviation and the skewness of the colour. In this work, additionally, the kurtosis is computed. Each of these four measurements is calculated for each pixel's neighbourhood on each channel of the RGB, the HSI and the CIELab colour space (see subsection 3.1.1). The colour moments for them are calculated as shown in equations (3.6) to (3.9) (P denoting the set of all pixels in the considered region, whereas those pixels are represented by their value $p \in \mathbb{R}$ concerning the inspected channel):

- **Mean:**

$$\mu = \frac{1}{|P|} \sum_{p \in P} p \tag{3.6}$$

This value represents the average colour of the channel.

- **Standard deviation:**

$$\sigma = \sqrt{\frac{1}{|P|} \sum_{p \in P} (p - \mu)^2} \tag{3.7}$$

- **Skewness:**

$$\theta = \sqrt[3]{\frac{1}{|P|} \sum_{p \in P} (p - \mu)^3} \tag{3.8}$$

Skewness is a measure of symmetry, or more precisely, the lack of symmetry[24].

- **Kurtosis:**

$$\kappa = \sqrt[4]{\frac{1}{|P|} \sum_{p \in P} (p - \mu)^4} \tag{3.9}$$

Kurtosis is a measure of whether the data are heavy-tailed or light-tailed compared to a normal distribution[24].

In this thesis, these four colour moments have been calculated on different channels of different colour spaces, namely:

1. The R, G, and B channels of the commonly used RGB colour space.
2. The L*, a*, and b* channels of the images converted into CIELab colour space.
3. The H, S, and I channels of the conversion to HSI colour space.
4. All nine channels mentioned above, but calculating the colour moments after performing a histogram equalisation of each channel.

[24] source: http://www.itl.nist.gov/div898/handbook/eda/section3/eda35b.htm, (accessed 2016-06-12)

3.2 Pre-Processing the Data

3.2.1 Sampling the Training Examples

In all experiments conducted in this thesis, only a fraction of the data is used for training the SVM. This is because training time increases substantially in the number of training examples: "The complexity of training of non-linear SVMs with solvers such as LIBSVM has been estimated to be quadratic in the number of training examples" (Ben-Hur and Weston, 2010). By choosing a linear kernel, training time can be reduced to linear time complexity (Joachims, 2006). However, in this thesis, most of the experiments use the non-linear RBF kernel (see subsection 3.3.4). Since several experiments had to be conducted to gain a better understanding of how different parameter settings would influence classification quality, the training time had to be reduced. Thus, the first step in each experiment is the reduction of the training examples. In (Schnürle, 2016; Suter, 2012, 2013; Suter et al., 2014), this was achieved by training only with pixels chosen by a fixed stride of e.g. 20 pixels in each hands image's direction. This is not adopted in this thesis in favour of the following approaches, which presumably respect the data distributions more adequately:

1. Random sampling of training examples in a stratified manner from the two class strat to retain the ratio of positive to negative examples.
2. Random sampling with replacement (bootstrapping) in a stratified manner from the two class stata.
3. Random sampling with replacement (bootstrapping) from ten classes of different levels of experts' confidence.

The hypothesis behind the 3rd approach is to provide for a more informed sampling of the data. The features describing each data example in this thesis are extracted from a certain neighbourhood of a pixel, whereas the example's class is determined by the consensus diagnosis for said single pixel. From this observation, it follows that an example might be described by features stemming from pixels both labelled eczematous as well as healthy in its neighbourhood (this is e.g. true for regions at the edge of the consensus diagnosis, see figure 3.1). It is therefore hypothesised that it might prove beneficial to assert that such ambiguous examples are provided to the machine learning algorithm since they might contain important information to help determine a reliable decision boundary. Thus, the number of experts' votes has been added up for each patch and divided by the skin pixels of the patch (which can be less than $41 \cdot 41 = 1'681$ on the edges of the hands as already discussed in section 3.1). This number is interpreted as the level of confidence expressed by the experts for a patch being eczematous as a whole. Finally, the patches are binned into ten classes representing the intervals $[0, 0.1[, [0.1, 0.2[, [0.2, 0.3[, \ldots, [0.9, 1]$ of this experts' confidence. These are the ten classes referred to in the 3rd strategy listed above.

Figure 3.17: The SMOTE algorithm creating a new data example, $K = 6$[25]

(a) original dataset

(b) after SMOTE

(c) Tomek links identified

(d) Tomek links removed

Figure 3.18: Tomek links removal[26]

The first two strategies maintain the ratio of positive to negative examples present in the original training data, whereas the 3rd chooses the same number of examples from all classes. After this initial sampling, additional sampling strategies may be applied. In the literature, various strategies to compensate for imbalanced data are discussed, see section 2.2. In this thesis, the following are incorporated into certain experiments, see chapter 5:

- Application of the SMOTE algorithm for the minority class.
- Detection and removal of Tomek links.

The generation of new data examples is shown in figure 3.17. For each example of the positive class, the algorithm determines its K nearest neighbours of the positive class with respect to the Euclidean distance. Then, one of those K nearest neighbours is selected randomly, and a new data instance is created at a randomly chosen position on the line segment between the original data example and the selected neighbour. In this thesis, $K = 5$ always, and SMOTE is applied multiple times until the minority class is of similar size as the majority class.

[25] source: (He and Garcia, 2009)
[26] source: (He and Garcia, 2009)

There are some potential issues mentioned in subsection 2.2.2 when applying SMOTE. One strategy to mitigate them is the application of data cleaning techniques such as the removal of Tomek links. (Batista et al., 2004) provide this definition of Tomek links: Given two examples E_i and E_j belonging to different classes, and $d(E_i, E_j)$ is the distance between E_i and E_j. A pair (E_i, E_j) is called a Tomek link if there is not an example E_l, such that $d(E_i, E_l) < d(E_i, E_j)$ or $d(E_j, E_l) < d(E_i, E_j)$. See figure 3.18 for an example for oversampling the data via SMOTE and removing the Tomek links afterwards.

3.2.2 Feature Scaling

Before providing the examples to the SVM, their features are either *normalised* or *standardised*, depending on which method is chosen for a specific experiment.

When applying normalisation, the features are scaled according to equation (3.10). Analogously, equation (3.11) is applied when scaling the features using standardisation[27]. $\boldsymbol{x} \in \mathbb{R}^n$ is a data example with n elements representing the features of this example. X is the set of all available examples, whereas T is the set of training examples, $T \subset X$. Note that minimum, maximum, mean and sample standard deviation are all calculated from T only. Calculating these values from X would be incorrect: "An important point to make about the preprocessing is that any preprocessing statistics (e.g. the data mean) must only be computed on the training data, and then applied to the validation / test data. E.g. computing the mean and subtracting it from every image across the entire dataset and then splitting the data into train/val/test splits would be a mistake. Instead, the mean must be computed only over the training data and then subtracted equally from all splits (train/val/test)"[28].

$$\forall \boldsymbol{x} \in X, \forall i \in \{1, 2, \ldots, n\} : \boldsymbol{x'}_i := \frac{\boldsymbol{x}_i - \min\limits_{\boldsymbol{t} \in T} \boldsymbol{t}_i}{\max\limits_{\boldsymbol{t} \in T} \boldsymbol{t}_i - \min\limits_{\boldsymbol{t} \in T} \boldsymbol{t}_i}. \tag{3.10}$$

$$\begin{aligned} \forall i \in \{1, 2, \ldots, n\} : \boldsymbol{s}_i &:= \sqrt{\frac{1}{|T| - 1} \sum_{\boldsymbol{t} \in T} (\boldsymbol{t}_i - \bar{\boldsymbol{t}_i})^2} \\ \forall \boldsymbol{x} \in X, \forall i \in \{1, 2, \ldots, n\} : \boldsymbol{x'}_i &:= \frac{\boldsymbol{x}_i - \bar{\boldsymbol{t}_i}}{\boldsymbol{s}_i} \end{aligned} \tag{3.11}$$

[27] standardisation in the corresponding KNIME normaliser node is based on the unbiased sample variance: org.knime.base.data.normalize.Normalizer2.doZScoreNorm effectively delegates to org.apache.commons.math3.stat.descriptive.moment.Variance, see http://commons.apache.org/proper/commons-math/apidocs/org/apache/commons/math3/stat/descriptive/moment/Variance.html (accessed 2016-06-22)

[28] http://cs231n.github.io/neural-networks-2/ (accessed 2016-06-22)

3.2.3 Feature Transformation and Selection

In some experiments, only a subset of the available features is used. The choice of which features to include is based on the *Pearson product-moment correlation coefficient (PCC)*. Table 3.5 shows an example of the correlation coefficients calculated for the hands image #239. In experiments that exclude features which are only barely linearly correlated to the class, those features with an absolute PCC of less than 0.1 are removed.

In some experiments, the skewness feature of the colour moments x is replaced by calculating $\sqrt{x}$ and $\ln x$. This is because tests have shown their PCC increase when transferred, and it is hypothesised that this might prove beneficial for the SVM.

Feature	PCC	GLCM	CM	Textons	Feature (continued)	PCC	GLCM	CM	Textons
GLCM Grey Eq contrast	0.35	X			CM CIELab kurtosis	0.17		X	
GLCM L* Eq contrast	0.34	X			CM RGB Eq kurtosis	0.17		X	
GLCM Grey dissimilarity	0.3	X			MR8 CIELab 20 Texton 1	0.17			
GLCM Grey contrast	0.3	X			GLCM L* Eq homogeneity	-0.17	X		
GLCM Sobel entropy	0.29	X			CM HSI Eq mean	-0.17		X	
GLCM L* dissimilarity	0.29	X			CM RGB std. dev.	0.17		X	
GLCM Grey Eq dissimilarity	0.29	X			GLCM Grey Eq entropy	0.16	X		
GLCM L* Eq dissimilarity	0.28	X			MR8 CIELab 20 Texton 0	-0.16			
GLCM Sobel homogeneity	-0.27	X			CM CIELab Eq mean	0.16		X	
GLCM L* contrast	0.27	X			CM RGB Eq std. dev.	0.15		X	
GLCM Grey homogeneity	-0.27	X			CM HSI skewness	0.15		X	
GLCM Grey entropy	0.27	X			CM CIELab kurtosis	0.15		X	
GLCM L* homogeneity	-0.27	X			MR8 CIELab 20 texton 15	0.15			
GLCM Sobel energy	-0.27	X			MR8 20 texton 10	-0.15			X
GLCM L* entropy	0.26	X			CM CIELab std. dev.	0.15		X	
CM CIELab Eq std. dev.	0.26		X		MR8 CIELab 20 texton 2	0.14			
GLCM Sobel dissimilarity	0.25	X			MR8 40 texton 17	-0.14			X
GLCM L* Eq entropy	0.25	X			GLCM Grey Eq homogeneity	-0.14	X		
CM HSI Eq kurtosis	0.25		X		MR8 20 texton 17	0.14			X
CM CIELab std. dev.	0.24		X		GLCM Sobel Eq entropy	0.14	X		
CM CIELab Eq kurtosis	0.24		X		MR8 CIELab 20 Texton 13	0.14			
CM RGB Eq kurtosis	0.24		X		MR8 20 texton 18	0.14			X
GLCM Sobel Eq contrast	0.24	X			MR8 40 texton 18	0.14			X
GLCM Sobel Eq dissimilarity	0.24	X			MR8 HSI 20 texton 12	0.14			
CM CIELab Eq std. dev.	0.23		X		CM CIELab std. dev.	0.14		X	
CM RGB Eq kurtosis	0.23		X		CM RGB kurtosis	0.13		X	
CM HSI Eq std. dev.	0.22		X		MR8 CIELab 20 texton 3	0.13			
CM CIELab Eq kurtosis	0.22		X		MR8 20 texton 2	0.13			X
GLCM Grey energy	-0.22	X			MR8 10 texton 2	0.13			X
CM HSI Eq kurtosis	0.22		X		MR8 40 texton 35	0.13			X
CM CIELab Eq kurtosis	0.22		X		MR8 HSI 20 texton 13	0.13			
GLCM L* energy	-0.21	X			CM CIELab Eq skewness	-0.12		X	
MR8 HSI 20 texton 3	0.21				CM HSI skewness	0.12		X	
CM HSI Eq std. dev.	0.21		X		MR8 CIELab 20 texton 4	-0.12			
CM CIELab kurtosis	0.21		X		MR8 HSI 20 texton 0	-0.12			
CM RGB Eq std. dev.	0.21		X		CM RGB std. dev.	0.12		X	
CM RGB kurtosis	0.21		X		MR8 40 texton 26	0.12			X
CM RGB Eq std. dev.	0.2		X		MR8 40 texton 6	0.11			X
CM HSI Eq kurtosis	0.2		X		MR8 40 texton 0	-0.11			X
GLCM Sobel contrast	0.2	X			MR8 40 texton 36	-0.11			X
CM CIELab Eq std. dev.	0.19		X		MR8 40 texton 23	0.11			X
CM RGB kurtosis	0.19		X		MR8 HSI 20 texton 2	-0.11			
CM HSI Eq std. dev.	0.19		X		CM CIELab mean	-0.11		X	
CM HSI mean	-0.18		X		MR8 10 texton 5	-0.11			X
CM RGB std. dev.	0.18		X		MR8 10 texton 3	0.1			X
CM CIELab mean	0.18		X		MR8 HSI 20 texton 16	-0.1			
MR8 40 texton 9	0.18			X	MR8 10 texton 0	-0.1			X
...					...				

Table 3.5: Correlation coefficients for the most predictive features of the examples of image #239, sorted descending on their coefficient's absolute value

3.3 Support Vector Machine

In this section, a Support Vector Machine (SVM) is introduced. Parts of the following descriptions have already been discussed in (Schnürle, 2016), and are revisited here. For a more comprehensive discussion, consult (Ben-Hur and Weston, 2010; Chapelle, 2007; Ng, 2015; Vapnik, 2000; Zisserman, 2015). As already emphasised in (Schnürle, 2016), this introduction is mainly a summary of their work.

The task of the SVM in this thesis is to provide for a good separation between healthy and eczematous skin. Thus, it is a question of binary classification (of two distinct classes) as opposed to regression (of continuous values). As introduced in subsection 1.3.2, expert markings are available which highlight eczematous skin regions. Hence, the application of a supervised learning approach like the SVM is feasible, using the consensus diagnosis of the experts' markings as the data examples' labels.

An SVM, fundamentally, optimises for a low error rate and thus high accuracy, since it aims to perfectly separate the two classes. To optimise for the F_1 score, which is this thesis' main reported metric, additional aspects have to be considered, which are discussed in subsection 3.3.6.

Amongst the attractive properties of an SVM classifier is the fact that "training an SVM amounts to solving a convex quadratic programming problem", which means that the solution found is global, as (Burges and Crisp, 2000) observe and analyse in greater detail.

Since all the optimisation objectives discussed in the subsequent subsections are convex, the global optimum of the particular objective is obtained. For an SVM, the quadratic programming problem is usually efficiently solved by executing the *Sequential Minimal Optimisation* (SMO) algorithm introduced by (Platt, 2000): "SMO breaks this large QP problem into a series of smallest possible QP problems. These small QP problems are solved analytically, which avoids using a time-consuming numerical QP optimisation as an inner loop". LIBSVM, which is used for the most of this thesis' experiments, implements a variant of this algorithm (Chang and Lin, 2011).

By training the SVM on a subset of the available hands images (see subsection 1.3.3) and predicting previously unseen (but also labelled) hands images, a confusion matrix is calculated. This confusion matrix provides for all metrics reported in this theses, see section 4.3.

Figure 3.19: Hard-margin SVM for training examples of two dimensions[29]

3.3.1 Hard-Margin SVM

In its most fundamental formulation, the SVM calculates a margin as large as possible between two linearly separable classes and thereby provides for the optimal separating hyperplane as the decision boundary. Thus, the SVM is also known as *maximum margin classifier*. This is shown in figure 3.19, for training examples with two features only (black circles representing one, empty circles the other class). $\boldsymbol{w} \in \mathbb{R}^n$ for n features is the normal vector to the decision boundary, with $\frac{b}{\|\boldsymbol{w}\|}$ determining the offset from the origin ($b \in \mathbb{R}$) and $\frac{2}{\|\boldsymbol{w}\|}$ describing the width of the margin between the two classes. By obtaining the decision boundary maximising the margin between the two classes of training examples, the best possible separation is found which should also yield the best possible performance when predicting previously unseen examples. However, the latter is true only if training examples representative of the population are chosen.

To find this decision boundary for a training set of pairs $(\boldsymbol{x^{(i)}}, y^{(i)}), \boldsymbol{x^{(i)}} \in \mathbb{R}^n, y^{(i)} \in \{1, -1\}, i = 1, \ldots, m$, the optimisation problem specified in equation (3.12) has to be solved. Note that maximising the margin $\frac{2}{\|\boldsymbol{w}\|} = \frac{2}{\sqrt{\boldsymbol{w}^T\boldsymbol{w}}}$ is equivalent to minimising $\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}$. The specified constraint prevents training examples from ending up on the "wrong side" of the margin.

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} \\ \text{subject to} \quad & y^{(i)}(\boldsymbol{w}^T\boldsymbol{x^{(i)}} + b) \geq 1, i = 1, \ldots, m \end{aligned} \tag{3.12}$$

[29] "Svm max sep hyperplane with margin" by Cyc - Own work. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png#/media/File:Svm_max_sep_hyperplane_with_margin.png – source: https://en.wikipedia.org/wiki/Support_vector_machine (accessed 2016-06-12)

Figure 3.20: Sensitivity of a hard-margin SVM to outliers[30]

The examples exactly on the positive or the negative hyperplane (highlighted in figure 3.19) are called *support vectors*, hence the machine's name. The support vectors essentially determine the decision boundary. The optimisation problem in equation (3.12) has a convex quadratic objective (since $\boldsymbol{w}^T\boldsymbol{w} = \|\boldsymbol{w}\|^2$) with linear constraints and can thus be solved using quadratic programming.

After training the SVM by optimising equation (3.12), the class of an unseen example $\boldsymbol{z} \in \mathbb{R}^n$ can be predicted very efficiently since this only involves deciding on which side of the hyperplane $\boldsymbol{z}$ lies. This is determined by calculating equation (3.13).

$$sgn(\boldsymbol{w}^T\boldsymbol{z} + b) \tag{3.13}$$

3.3.2 Soft-Margin SVM

A hard-margin SVM can be very sensitive to outliers as shown in figure 3.20. Moreover, data is often noisy, such that it might be questionable whether a perfectly separating, hard-margin decision boundary is the classifier to strive for. These are reasons why soft-margin SVMs have been introduced, see (Cortes and Vapnik, 1995). The idea behind a soft-margin classifier is that it allows for examples to be wrongly classified, but at the price of increased misclassification cost. This has the additional benefit that the SVM is less prone to overfitting.

Figure 3.21 shows two situations which differ in a newly introduced parameter $C \in \mathbb{R}$ (note that the data, i.e. the examples and their labels, are identical in both figures). On the left-hand side, the separating margin is quite narrow, since the SVM tried to come as close as possible to a hard-margin classifier (represented by a large value of $C = 100$). On the right-hand side, the parameter C was relaxed to $C = 10$, which resulted in a wider margin, but also in more examples within the margin, which would be prohibitive for a hard-margin SVM.

[30] source: (Ng, 2015)
[31] source: (Ben-Hur and Weston, 2010)

Figure 3.21: Soft-margin SVM with two different values for C[31]

To achieve this soft-margin behaviour, an additional term is introduced to equation (3.12), the so-called *regularisation term*. This is a hinge loss function; equation (3.12) shows the linear hinge loss function for an L1-SVM (the SVM discussed in this section). Other hinge losses are possible, e.g. the squared hinge loss used in L2-SVMs (Tang, 2013). The regularisation term penalises examples within the margin as well as misclassified examples by a value which increases with the distance to the example's "correct" class' hyperplane.

$$\max(0, 1 - y^{(i)}(\boldsymbol{w}^T \boldsymbol{x}^{(\boldsymbol{i})} + b)) \tag{3.14}$$

Equation (3.14) incorporated into equation (3.12) yields equation (3.15); $\zeta \in \mathbb{R}^m$.

$$\begin{aligned} \min_{\boldsymbol{w},b,\boldsymbol{\zeta}} \quad & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{m}\zeta_i \\ & \zeta_i = \max(0, 1 - y^{(i)}(\boldsymbol{w}^T \boldsymbol{x}^{(\boldsymbol{i})} + b)) \\ \text{subject to} \quad & y^{(i)}(\boldsymbol{w}^T \boldsymbol{x}^{(\boldsymbol{i})} + b) \geq 1 - \zeta_i, i = 1, \ldots, m \\ & \zeta_i \geq 0, i = 1, \ldots, m \end{aligned} \tag{3.15}$$

C in equation (3.15) is used to balance the two goals of gaining a large margin and minimising the number of falsely classified examples. Those goals are represented by the two terms in the equation, the data loss on the left side, and the regularisation loss on the right side. The optimisation objective strives to minimise both of them to minimise the whole objective, while C puts an emphasis on one of them. With a large value for C, less erroneously examples are allowed since the regularisation term has to be minimised more emphatically.

The decision function equation (3.13) for unseen examples is unaffected by the introduction of the regularisation term and remains the same.

3.3.3 Dual Form

Both equations (3.12) and (3.15) describe the optimisation problem in its so-called *primal form*. This can be transferred to its corresponding *dual form* by solving for the Lagrangian dual of the optimisation problem in its primal form. The derivation of this transition is omitted here for brevity, see (Ng, 2015) for details. A key observation is that by this transformation, the optimisation problem can be written in terms of dot products only (e.g., $(\boldsymbol{x^{(i)}})^T\boldsymbol{x^{(j)}}$). This will allow for the efficient application of the so-called *kernel trick*, which is introduced in subsection 3.3.4.

The dual form of the optimisation problem for the hard-margin SVM as specified in equation (3.12) is stated in equation (3.16), $\alpha \in \mathbb{R}^m$.

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j(\boldsymbol{x^{(i)}})^T\boldsymbol{x^{(j)}} \\ \text{subject to} \quad & \alpha_i \geq 0, i = 1, \ldots, m \\ & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \end{aligned} \tag{3.16}$$

The optimisation problem of the soft-margin SVM as specified in its primal form in equation (3.15) can be transferred to its dual form with the same method, too, thereby yielding equation (3.17).

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j(\boldsymbol{x^{(i)}})^T\boldsymbol{x^{(j)}} \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, i = 1, \ldots, m \\ & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \end{aligned} \tag{3.17}$$

Remarkably, the addition of the regularisation term does not affect the optimisation's objective function in equation (3.17); the only difference to the hard-margin SVM is how the α_i are constrained.

Note that the reformulation in the dual form does not change the effective decision boundary, which is the same for the formulation of the algorithm in its primal as well as in its dual form. As mentioned initially in this section, the benefit of the dual form is that calculations can entirely be performed in terms of dot products, which in turn can easily be replaced by a kernel as described in subsection 3.3.4.

The training examples for which the $\alpha_i > 0$ "are those points that are on the margin, or within the margin when a soft-margin SVM is used"(Ben-Hur and Weston, 2010), as well as those even completely on the wrong side of the decision boundary. Those are the support vectors.

From the Lagrangian conversion, we obtain equation (3.18). After the optimal value for the α_i is found by solving equation (3.16) or equation (3.17), respectively, $\boldsymbol{w}$ can be calculated from equation (3.18). b can then be recovered, too, by calculating equation (3.19) for an $\boldsymbol{x^{(k)}}$ with $\alpha_k > 0$.

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y^{(i)} \boldsymbol{x^{(i)}} \tag{3.18}$$

$$\begin{aligned} y^{(k)}(\boldsymbol{w}^T \boldsymbol{x^{(k)}} + b) &= 1 \qquad\qquad , y^{(k)} \in \{1, -1\} \\ y^{(k)} y^{(k)}(\boldsymbol{w}^T \boldsymbol{x^{(k)}} + b) &= y^{(k)} \\ b &= y^{(k)} - \boldsymbol{w}^T \boldsymbol{x^{(k)}} \end{aligned} \tag{3.19}$$

To classify an unseen example $\boldsymbol{z} \in \mathbb{R}^n$, the decision function in equation (3.20) is calculated, which still conveys the same idea as equation (3.13) for the primal form: deciding on which side of the hyperplane the unseen example lies.

$$sgn(\boldsymbol{w}^T \boldsymbol{z} + b) = sgn((\sum_{i=1}^{m} \alpha_i y^{(i)} \boldsymbol{x^{(i)}})^T \boldsymbol{z} + b) \tag{3.20}$$

These calculations involving the α_i have therefore to be carried out only for the support vectors, since for all the other training examples, the $\alpha_i = 0$.

3.3.4 Kernel Trick

Even though the relaxation of allowing soft margins extends the applicability of an SVM, the constraint of having two linearly separable classes still holds. However, data being inseparable in their original feature space may very well be separable in a higher dimensional space.

This is easily formulated as shown in equation (3.21) (as formulated in (Hsu et al., 2003)). This differs from equation (3.15) only in replacing $\boldsymbol{x^{(i)}}$ by $\phi(\boldsymbol{x^{(i)}})$ ($\phi : \mathbb{R}^n \to \mathbb{R}^l$, whereas l, the dimension of the new space, depends on the kernel used; see the discussion of the polynomial kernel below). $\phi(\boldsymbol{x^{(i)}})$ represents a mapping to such a higher dimensional space.

$$\begin{aligned} \min_{\boldsymbol{w}, b, \boldsymbol{\zeta}} \quad & \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{m} \zeta_i \\ & \zeta_i = \max(0, 1 - y^{(i)}(\boldsymbol{w}^T \phi(\boldsymbol{x^{(i)}}) + b)) \\ \text{subject to} \quad & y^{(i)}(\boldsymbol{w}^T \phi(\boldsymbol{x^{(i)}}) + b) \geq 1 - \zeta_i, i = 1, \ldots, m \\ & \zeta_i \geq 0, i = 1, \ldots, m \end{aligned} \tag{3.21}$$

A naive approach to map the original features to a higher dimensional space is by explicitly defining a feature map. However, this does not scale, especially with an increasing number of features. Hence, the *kernel trick* is introduced to circumvent this explicit mapping. Equation (3.22) introduces the relationship between a kernel and dot products in the higher dimensional space.

$$\begin{aligned} k &: \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R} \\ k(\boldsymbol{x}, \boldsymbol{x'}) &= \phi(\boldsymbol{x})^T \phi(\boldsymbol{x'}) \end{aligned} \tag{3.22}$$

Now, the dual form specified in equation (3.17) can be written as equation (3.23) when considering the higher dimensional feature space and applying equation (3.22).

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j k(\boldsymbol{x^{(i)}}, \boldsymbol{x^{(j)}}) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, i = 1, \ldots, m \\ & \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \end{aligned} \tag{3.23}$$

The decision function for an unseen example $\boldsymbol{z}$ shown in equation (3.20) analogously becomes equation (3.24).

$$sgn(\boldsymbol{w}^T \phi(\boldsymbol{z}) + b) = sgn((\sum_{i=1}^{m} \alpha_i y^{(i)} k(\boldsymbol{x^{(i)}}, \boldsymbol{z}) + b) \tag{3.24}$$

Another intuition of kernels is thinking of them as a kind of similarity measurement between two examples $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{x'} \in \mathbb{R}^n$. The kernel should be large when $\boldsymbol{x}$ and $\boldsymbol{x'}$ are close together, and small when they are far apart (Ng, 2015). From this point of view, the kernel function in equation (3.25) would be a reasonable choice, a reasonable measurement of similarity. This is the Gaussian radial based function (RBF) kernel often used with SVMs in general, and in most of the experiments in this thesis, too.

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x'}\|^2), \gamma = \frac{1}{2\sigma^2} \tag{3.25}$$

Many other kernels exist in the literature; the most popular others are the linear kernel shown in equation (3.26) and the polynomial kernel shown in equation (3.27).

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \boldsymbol{x}^T \boldsymbol{x'} \tag{3.26}$$

$$k(\boldsymbol{x}, \boldsymbol{x'}) = (\gamma \boldsymbol{x}^T \boldsymbol{x'} + r)^d, \gamma > 0 \tag{3.27}$$

Figure 3.22: Polynomial feature transformation[32]

The linear kernel effectively applies no transformation to a higher dimensional feature space at all but therefore is faster to calculate. Moreover, the decision function for unseen examples in its primal form as shown in equation (3.13) can be used, which is faster to calculate, too.

An example of the feature mapping performed by a polynomial kernel with degree $d = 2$ and parameters $\gamma = 1, r = 0$ is shown in figure 3.22. The new feature space consists of all monomials of degree d. With $d = 2$, the original two features $x_1, x_2 \in \mathbb{R}$ are effectively mapped to $(x_1^2, \sqrt{2}x_1x_2, x_2^2)$. For $d = 1$, the polynomial kernel effectively degrades to the linear kernel.

The polynomial kernel is an illustrative example for the attractiveness of applying a kernel and thus circumventing the calculation of the higher dimensional features space. The number of monomials of degree d grows fast with increasing d and number of features n: $\binom{d+n-1}{d}$. For $d = 2$ and $n = 2$, as in the example above, this results in three terms, as shown explicitly. However, for an example more realistic for this thesis, let $d = 3, n = 50$. The number of terms to calculate would be 22'100 already. Calculating the kernel $(\boldsymbol{x}^T\boldsymbol{x'})^d$ involves only n multiplications, $n - 1$ additions and finally d multiplications.

For any $k(\boldsymbol{x}, \boldsymbol{x'})$ being feasible as a kernel, it is necessary and sufficient that its corresponding kernel matrix $\boldsymbol{K}$ is symmetric positive semi-definite. This observation is captured by *Mercer's theorem*. Suppose that $k(\boldsymbol{x}, \boldsymbol{x'})$ is a valid kernel that corresponds to some feature mapping ϕ. Then, for any finite set of m data examples, the symmetric real kernel matrix $\boldsymbol{K} \in \mathbb{R}^{m\times m}$ is defined such that its components (i, j) are given by $\boldsymbol{K}_{ij} = k(\boldsymbol{x^{(i)}}, \boldsymbol{x^{(j)}})$. For any vector $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{z}^T\boldsymbol{K}\boldsymbol{z} \geq 0$ holds. Since $\boldsymbol{z}^T\boldsymbol{K}\boldsymbol{z}$ is non-negative, $\boldsymbol{K}$ is positive semi-definite. Or, equivalently, if this is not true for a specific set of m data examples, then $\boldsymbol{K}$ is not positive semi-definite, and thus $k(\boldsymbol{x}, \boldsymbol{x'})$ is not a valid kernel. This is shown in more detail by (Ng, 2015).

[32]source: http://www.cs.columbia.edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf

3.3.5 Imbalanced Data

Figure 3.23: SVM on imbalanced data[33]

Imbalanced data might deteriorate classification quality as laid out in subsection 2.2.2. Such a situation is shown in figure 3.23, where on the left-hand image a soft-margin SVM is applied as described until now, while the right-hand image's SVM penalises falsely classified examples of the "plus sign" class heavier than those of the other class. The latter behaviour is achieved by applying different error costs to the examples of different classes. For this purpose, the regularisation parameter C is split into $C_+ \in \mathbb{R}$ applied on all training examples of the positive class and $C_- \in \mathbb{R}$ applied on the examples of the negative class. This is shown in equation (3.28).

$$C\sum_{i=1}^{m}\zeta_i \rightarrow C_+ \sum_{\{i|y^{(i)}=1\}} \zeta_i + C_- \sum_{\{i|y^{(i)}=-1\}} \zeta_i \tag{3.28}$$

A reasonable choice for C_+, C_- is the ratio of the number of negative to positive training examples (Ben-Hur and Weston, 2010; Musicant et al., 2003), which yields equation (3.29). Integration into equation (3.23) yields equation (3.30).

$$C_+ n_+ = C_- n_- \Leftrightarrow \frac{C_+}{C_-} = \frac{n_-}{n_+} \tag{3.29}$$

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j k(\boldsymbol{x^{(i)}}, \boldsymbol{x^{(j)}}) \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C_+, i = 1, \ldots, m, \text{if } y^{(i)} = 1 \\ & 0 \leq \alpha_i \leq C_-, i = 1, \ldots, m, \text{if } y^{(i)} = -1 \\ & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \end{aligned} \tag{3.30}$$

This approach can be generalised to apply an individual, different cost on each data example, which is the main idea behind *Fuzzy SVMs* as briefly introduced in subsection 2.2.4.

[33] source: (Ben-Hur and Weston, 2010)

3.3.6 Optimising For F1 Score

As stated at the beginning of this section, an SVM optimises for a low error rate. However, in this thesis, performance is measured in the F_1 score. To modify the decision boundary such that a high F_1 score is reached, the already introduced idea (see subsection 3.3.5) of assigning different weights to the regularisation term of the optimisation objective for different classes is used. Thus, the decision boundary is shifted and skewed (see figure 3.23) such that by reducing the number of false positives or false negatives, higher recall or precision is achieved. Thus, the difference between recall and precision can be reduced, resulting in a higher F_1 score. By varying the ratio of C_+ to C_-, more or less emphasis can be put upon one or the other class' errors. The larger C_+ or C_-, the more important it becomes for the optimisation objective to optimise for a more accurate classification of the positive or the negative training examples, respectively.

A high recall is reached when most, ideally all, positive training examples are on the correct side of the decision boundary separating both classes, such that there are no false negatives. High precision, on the other hand, is reached when most, ideally all, negative training examples are on the correct side of the decision boundary, such that there are no false positives. The F_1 score, the harmonic mean of recall and precision, takes both of these two often contradictory goals into account.

In a few experiments conducted in this thesis, the application of a formulation of an SVM which can directly optimise for multivariate metrics like the F_1 score (see (Joachims, 2005) as well as subsection 2.2.5) has been investigated.

3.4 Evaluation Setting

Disclaimer: This section 3.4 is joint work and originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

Three master theses are conducted on the SkinApp project in parallel. They all have the goal of improving classification quality on the available hands images in common. To compare the results of the theses effectively, a common evaluation scheme is needed. This section aims to describe this scheme precisely.

The experiments conducted are evaluated using *9-fold cross-validation* on the available 48 hands images. Since there are 24 images of front sides of hands as well as 24 images of back sides, those two sets are evaluated completely separate from each other. This accounts for the presumed differences in the skin's structure (e.g. wrinkles) as well as in specific skin features (e.g. hair density) between the front and back sides of hands, as previous work suggests, see (Schnürle, 2016; Suter, 2013; Suter et al., 2014). Thus, the following description applies to both sets, front and back sides of hands, respectively.

The hands images are associated with different numbers of experts' markings as shown in subsection 1.3.3. The six hands images with the least number of markings are held out from the subsequent validation process. The images in this *hold-out set* can be used, e.g., for a search for good hyperparameters for the machine learning algorithm. The hold-out set can be used as a test set e.g. in a separate step before the subsequent 9-fold cross-validation (in case hyperparameters common to all 9 validation folds are searched), or within each fold (in case the hyperparameters are searched for each of the 9 validation folds separately).

With six images excluded, the 9-fold cross validation is performed on the remaining 18 hands images, which define the *evaluation set*. For each fold, 16 hands images are used for training the machine learning algorithm, and the remaining two images are predicted by the trained classifier, thus validating its expected generalisation performance.

Figures 3.24 to 3.27 show two admissible approaches for the evaluation. In both variants, a 9-fold cross-validation is conducted. The hold-out set can be used as appropriate. The two variants suggested above are visualised exemplary: One variant uses the hold-out set to calculate hyperparameters before conducting the 9-fold cross-validation runs, and in another variant, the hold-out set is used as test set when calculating distinct hyperparameters within in each fold.

For each of the 9 folds, and for each image in each fold's test set, a confusion matrix is calculated, which allows for the evaluation of the classifier's performance on each fold and image. Table 3.6 assigns the hands images of the evaluation set to the 9 folds. Their assignment to the folds is purely random. Nevertheless, to provide for comparability between the three theses, the images predicted in each fold have to be specified unequivocally.

Figure 3.24: Evaluation settings, variant A

Figure 3.25: Evaluation settings, variant A, 9-fold cross-validation details

Figure 3.26: Evaluation settings, variant B

Figure 3.27: Evaluation settings, variant B, 9-fold cross-validation details

Front sides		Back sides	
ID	Fold	ID	Fold
1010	1	240	1
223	1	246	1
1020	2	1023	2
232	2	224	2
1018	3	231	3
1024	3	238	3
225	4	1019	4
228	4	229	4
1004	5	218	5
221	5	233	5
234	6	242	6
241	6	226	6
1008	7	1009	7
236	7	244	7
219	8	220	8
239	8	222	8
243	9	1001	9
227	9	1021	9

Table 3.6: Assignment of hands images to the folds

		Predicted value		
		positive	**negative**	**total**
Actual value	**positive**	True Positive (TP)	False Negative (FN)	Actual Positive (Ac_+)
	negative	False Positive (FP)	True Negative (TN)	Actual Negative (Ac_-)
	total	Predicted Positive (Pr_+)	Predicted Negative (Pr_-)	

Figure 3.28: Confusion matrix[34]*;* $Ac_+ = TP+FN, Ac_- = FP+TN, Pr_+ = TP+FP, Pr_- = FN+TN$

However, in the three theses, a single value for each metric for front sides as well as back sides of hands images is reported. Hence, the confusion matrices (see figure 3.28) of all individual 9 folds are added together. On this overall confusion matrix, all metrics (with the exception of area error, see below) are calculated. The main result reported is the F_1 score, for which all theses optimise. Additionally, accuracy, recall, and precision are reported. The area error is calculated on each image, not on the combined, overall confusion matrix for all images. Then, the arithmetic mean of the individual area errors is reported.

[34] source: http://tex.stackexchange.com/questions/20267/how-to-construct-a-confusion-matrix-in-latex/20284#20284 (accessed 2016-06-09)

3.5 Implementation

Figure 3.29: KNIME basic cross-validation workflow with preceding grid search for hyperparameters

The implementation of the approaches discussed in this chapter is based on extending the existing C++ code base of (Schnürle, 2016) as well as modelling different workflows in KNIME (see section 1.3). The latter has proved to be less time-consuming than implementing the workflows directly in C++, such that it has been possible to conduct more diverse experiments.

Image processing and feature extraction are implemented in C++. By calling the executable with a JSON configuration file as a parameter as introduced in (Schnürle, 2016), comma-separated values (CSV) files are written onto the file system. These CSVs are picked up by the various KNIME workflows, which are manually started. Since different CSVs are created for different feature groups (e.g., features calculated from a GLCM on greyscale images, or colour moments), the KNIME workflows can be configured flexibly depending on which features or feature sets should be added in which experiment.

Figure 3.29 shows the basic workflow used for most experiments in this thesis from a high-level perspective. The grey nodes are so-called "Meta Nodes", which in turn consist again of several nodes and have been individually configured in this thesis. The basic workflow consists of about 200 nodes: some of them representing very simple steps like conditional branching, others encapsulating the complete training of an SVM. The workflow corresponds to the "variant B" shown in figures 3.26 and 3.27, although from studying figure 3.29, this is not immediately apparent. This is because of computational considerations, such as the parallelisation of certain tasks. They do not alter the evaluation process as introduced in section 3.4 but result in a different workflow layout.

For example, as shown in figure 3.27, for each fold, separate hyperparameters are identified. The folds are computed iteratively. However, since the training sets and the test sets for the folds do not change, it is feasible to perform the grid searches and the prediction of the test images separately from each other. The prediction of one particular fold depends on the grid search of said fold, but while the predictions for one fold are calculated, the grid searches for the next folds can already be conducted. This is why on figure 3.29, there is no explicit connection visible between the node "Grid Search For Best Hyperparameters" and the node "K-Fold Cross-Validation". The synchronisation between these two process steps is implicit via the file system: After the grid search for one fold is completed, the results are written as CSV file to the file system. The validation task for the corresponding fold waits for this file being created and then executes the subsequent prediction of the images in the test set.

Many similar dependencies exist in the workflow, but most of them are explicitly defined by linking nodes. KNIME itself utilises multiple CPUs by itself already and starts tasks in parallel which are not dependent on each other. Since this thesis could benefit from a 32-CPU server provided by the Enterprise Lab team of the University, computation time could be reduced by parallelisation with comparatively small effort.

Several nodes provided by KNIME which perform computationally expensive tasks are slower than native implementations. This is especially apparent in the process steps most time-consuming in all the workflows used: training the SVM and subsequently predicting the unseen images. Even though there are different SVM nodes available for KNIME, they all fall either short of providing a means to set all parameters necessary in this thesis (e.g. setting different error costs for different classes is not supported by all SVM nodes), or take too much time to perform the calculations. Thus, the "External Tool" node has been used to execute bash scripts which in turn perform the calculation via the native LIBSVM executables. This resulted in great performance gains but at the cost of adding complexity (due to additional synchronisation between nodes via the file system) and mapping the data from KNIME to the format usable by the LIBSVM executables. Moreover, KNIME runs on the Java Virtual Machine, and certain tasks take a longer time to process than when they would have been written in e.g. C++. However, the performance gains with large sets of data outweigh the drawbacks by far.

For brevity, discussing every detail of the workflows implemented is omitted. Most nodes encapsulate only tiny and thus easily traceable process steps, such that the workflows are self-explanatory to a certain extent, especially for users experienced in using KNIME in general. However, to allow for a better intuition about KNIME and the workflows, a few interesting nodes are inspected.

Consider figure 3.30 which shows the "K-Fold Cross-Validation" meta node on the lower side of figure 3.29. The cross-validation node corresponds to figure 3.27. Unfortunately, there are many nodes responsible for technical details that slightly obfuscate the meta node's main functionality. For example, the lower left node "Row Filter – only front or back image IDs" simply removes either the front sides or back sides images irrelevant to a particular test run.

Figure 3.30: KNIME k-fold cross-validation meta node

The most important nodes in this meta node are certainly the loop nodes which delimit the actual cross-validation. Each fold's images are split according to the pre-defined distributions shown in table 3.6; this is carried out by the two "Reference Row Filter" nodes left of the middle of figure 3.30. As mentioned above, the grid search results are read by another meta node on the lower side which includes waiting for the file to be created.

The most interesting node is the "Single Run Images Granularity SVM RBF+Weights Ensemble5 ParallelPredict2" node: This node contains training SVMs and predicting the test data. In this specific workflow, it encapsulates an ensemble of five SVMs, and the prediction of the test data is parallelised by invoking two instances of the LIBSVM executable. Another benefit of defining meta nodes becomes apparent: when the input and output data match, i.e. the interface of a meta node is compatible to another, they can be easily exchanged, which was often done for different experiments. The content of this ensemble node is shown in figure 3.31 to address a few more details.

The ensembles have been explicitly modelled to provide for parallelisation, at the cost of scaling badly. However, since in this thesis, only an ensemble of five SVMs is examined, this drawback has been accepted. The data is split and distributed to the ensemble members. Then, the SVMs are executed in the "SVMTrainAndPredict Parallel2" meta nodes, and their predictions are collected, which is represented by a few nodes mainly for technical reasons. The "Math Formula" node at the end calculates the majority voting of the whole ensemble for every data example.

Figure 3.32 showing the "SVMTrainAndPredict Parallel2" meta node concludes this discussion on KNIME. After the data has been split according to the nodes' descriptions, the LIBSVM training executable runs on the training data. The external node writes the data on the file system, provides the executable with needed parameters and collects the data again. Collecting does not run very stable. Thus, a robust meta node was created, the "CSV Reader Safe" node. For the file names on the file system, unique strings have been used to provide for several SVM instances running in parallel on different data. The bash scripts executed by the external tool node are shown in Appendix B. To map the data from the CSV written on the file system by the external tool node to the input format expected by LIBSVM (Chang and Lin, 2011) or SVM-perf (Joachims, 2005), the python script "csv2libsvm.py"[35] is executed.

[35] available at `https://github.com/zygmuntz/phraug` (accessed 2016-06-22)

Figure 3.31: KNIME SVM ensemble

Figure 3.32: KNIME SVM training and prediction of unseen examples via the external tool node

Chapter 4

Experiments

This chapter describes the features that are used in the experiments in section 4.1 as well as a number of sets of experiments conducted, see section 4.2. Additionally, the metrics reported in this thesis are briefly discussed in section 4.3, and the metrics reported by a number of baseline classifiers are shown, see section 4.4.

4.1 Features Used In Experiments

Table 4.1 shows the features considered in this thesis. Each experiment utilises a subset of those, as described in each experiment's setting, see section 4.2.

4.2 Experiments

The following sets of experiments aim to provide for the best possible improvement of classification quality as well as insights on which features are the most promising concerning improving classification quality by providing a hypothesis for each experiment. In the hypothesis columns, notations like e.g. "$F_1 \approx$ @1.1" or "$F_1 >$ @6.1" refer to distinct run IDs. In the first example, this is to be read as "the F_1 score is expected to be approximately the same as in run 1.1" or "the F_1 score is expected to be greater than in run 6.1", respectively. The definition of sets of experiments instead of discussing them individually allows for observing the changes in performance when varying only a small number of parameters.

Set	Description	Features	Reference
MR8 20	The MR8 filter bank as used in previous work (see section 2.1) with 20 Textons in the Texton dictionary. 10 of the Textons stem from all pixels of the hands images in the hold-out set which are labelled as positive, and 10 from those labelled negative. For each pixel, the nearest Textons histogram for its neighbourhood is calculated.	20 Texton histogram bins	subsection 3.1.3
MR8 40	The same MR8 filter bank as in "MR8 20", but with 40 Textons instead of 20. 20 Textons stem from the positive, and 20 from the negative class.	40 Texton histogram bins	subsection 3.1.3
MR8 HSI 20	Application of the MR8 filter bank as in "MR8 20", adding additional 3 "filter responses" for each pixel corresponding to its values for the 3 channels of the HSI colour space (see subsection 3.1.1.	20 Texton histogram bins	subsection 3.1.3
MR8 CIELab 20	Application of the MR8 filter bank as in "MR8 20", adding additional 3 "filter responses" for each pixel corresponding to its values for the 3 channels of the CIELab colour space (see subsection 3.1.1.	20 Texton histogram bins	subsection 3.1.3
CM RGB	The first four colour moments of each pixels' neighbourhood are calculated on each of the three channels.	12 colour moments	subsection 3.1.4
CM HSI	As in "CM RGB", the colour moments of the HSI channels are calculated.	12 colour moments	subsection 3.1.4
CM CIELab	As in "CM RGB", the colour moments of the CIELab channels are calculated.	12 colour moments	subsection 3.1.4
CM RGB Eq	The histogram of each channel is equalised. Then, as in "CM RGB", the colour moments of the RGB channels are calculated.	12 colour moments	subsection 3.1.4
CM HSI Eq	As in "CM RGB Eq", the colour moments of the HSI channels are calculated.	12 colour moments	subsection 3.1.4
CM CIELab Eq	As in "CM RGB Eq", the colour moments of the CIELab channels are calculated.	12 colour moments	subsection 3.1.4
GLCM Grey	The RGB hands images are converted to greyscale. Then, the GLCM of each pixels' neighbourhood is calculated. Finally, the GLCMs' entropy, energy, homogeneity, contrast, and dissimilarity are calculated.	5 GLCM features.	subsection 3.1.2
GLCM L*	As in "GLCM Grey", the GLCMs are calculated, although not on the greyscale images, but rather on the CIELab L* channel.	5 GLCM features.	subsection 3.1.2
GLCM Sobel	As in "GLCM Grey", the greyscale image is calculated from the RGB channels. Then, the Sobel operator is applied. Finally, on the resulting edge detection image, the GLCM is calculated.	5 GLCM features.	subsection 3.1.2
GLCM Grey Eq	The histogram of the greyscale image is equalised. Then, as in "GLCM Grey", the GLCM is calculated.	5 GLCM features.	subsection 3.1.2
GLCM L* Eq	As in "GLCM Grey Eq" and "GLCM L*", the GLCM is calculated.	5 GLCM features.	subsection 3.1.2
GLCM Sobel Eq	As in "GLCM Grey Eq" and "GLCM Sobel", the GLCM is calculated.	5 GLCM features.	subsection 3.1.2

Table 4.1: Features chosen for the experiments

In the tables in the following subsections, some abbreviations are used. These are their meanings:

1. TSM: Method by which the training data has been sampled. Three methods are used in this thesis: random sampling of training examples on the two class strata (SC), bootstrapping training examples on the two class strata (BC), and bootstrapping training examples on ten experts' confidence levels (BA). These are introduced in subsection 3.2.1.
2. TS: Training data sample size. Depending on the value of the TSM column, this is either the percentage of examples sampled from the class strata or the absolute number of examples bootstrapped from each of the ten experts' confidence levels.
3. HP: Grid over which hyperparameters have been searched; the values displayed show the logarithm to base 2 of the actual values searched.
4. Trans: Whether some features have been transformed by calculating the natural logarithm as well as the square root instead of using the original feature. This transformation is applied only to the skewness feature of the colour moments, see subsection 3.2.3 for details.
5. FN: Feature normalisation. This is either MM for min-max normalisation according to equation (3.10), or a ZS for a Z-score standardisation according to equation (3.11).
6. E: Number of ensemble members of the SVM ensemble if applicable. In this thesis, this number either equals 1 for no ensemble or 5 for an ensemble of 5 members used.
7. SVM: The type of SVM and kernel used. This is RBF+DEC for the RBF kernel with different error costs (used in most experiments), Lin+DEC for the linear kernel with DEC, Poly+DEC for the polynomial kernel with DEC; or Lin for the linear kernel without DEC but with the SVM-perf implementation introduced in (Joachims, 2005) directly optimising for the F_1 score.
8. LCF: Features exposing a low absolute value in linear correlation are filtered out or not.

Two runs are always grouped in one row in the tables, providing for the fact that the same experiments are conducted on both the front side as well as the back side hands images. The ID of a run for back sides of images is equal to the ID of the corresponding run for front sides increased by 1'000. The changes in parameters especially investigated are set in a bold typeface. The values of the parameters never varied over the runs of a set of experiments are given in the text preceding the tables.

4.2.1 Textons Only

The set of experiments shown in table 4.2 compares different Texton-based approaches and thus provides a link to previous work in (Schnürle, 2016), see section 2.1. These experiments consider Texton features only, colour moments and GLCMs are ignored.

For all these runs, a min-max normalisation is performed before invoking the SVM. No SVM ensembles are utilised, and the SVMs all used the RBF kernel with different error costs for the two classes. Feature transformation is not applied.

Run	Features	TS	TSM	HP	Hypothesis and expectations
1.1 1001.1	MR8 40	0.1%	SC	C: 7..17 γ: -11..3	baseline against previous work (Schnürle, 2016) front: $F_1 \approx 35\%$ back: $F_1 \approx 26\%$
1.2 1001.2	MR8 40	0.1%	SC	C: 7..17 γ: -11..3	different training examples than @1.1 $F_1 \approx$ @1.1
5.1 1005.1	**MR8 20**	0.1%	SC	C: 7..17 γ: -11..3	fewer Textons, thus histograms less predictive of class $F_1 <$ @1.1/1.2
5.2 1005.2	**MR8 20**	0.1%	SC	C: 7..17 γ: -11..3	different training examples than @5.1 $F_1 \approx$ @5.1
2.1 1002.1	MR8 40	**0.15%**	SC	**C: 11..17** **γ: -5..1**	more examples than @1.1/1.2 (smaller grid only for computational reasons) $F_1 >$ @1.1/1.2
2.2 1002.2	MR8 40	**0.15%**	SC	**C: 11..17** **γ: -5..1**	different training examples than @2.1 $F_1 \approx$ @2.1
11.1 1011.1	MR8 40	**50**	**BA**	C: 7..17 γ: -11..3	presumably more informed sampling method $F_1 >$ @1.1/1.2
11.2 1011.2	MR8 40	**100**	**BA**	C: 7..17 γ: -11..3	different training examples than @11.1 $F_1 \approx$ @11.1
3.1 1003.1	**MR8 HSI 20**	0.1%	SC	C: 7..17 γ: -11..3	colour information of HSI space added $F_1 <$ @1.1/1.2
3.2 1003.2	**MR8 HSI 20**	0.1%	SC	C: 7..17 γ: -11..3	different training examples than @3.1 $F_1 \approx$ @3.1
4.1 1004.1	**MR8 CIELab 20**	0.1%	SC	C: 7..17 γ: -11..3	colour information of CIELab space added $F_1 <$ @1.1/1.2
4.2 1004.2	**MR8 CIELab 20**	0.1%	SC	C: 7..17 γ: -11..3	different training examples than @3.1 $F_1 \approx$ @3.1

Table 4.2: Experiment settings: Textons Only

4.2.2 Colour Moments, GLCM

This set of experiments in table 4.3 aims to investigate whether the features often used in the literature (see section 2.2 and section 3.1) provide for a good classification performance.

For all these runs, a min-max normalisation is performed before invoking the SVM. No SVM ensembles are utilised, and the SVMs all used the RBF kernel with different error costs for the two classes. The sample size of the training examples is 0.1%, sampled in a stratified manner on the class strata. The grid searched is C: 7..17, γ: -11..3.

Run	Features	Trans	Hypothesis and expectations
6.1 1006.1	**CM RGB** **GLCM Grey**	N	should perform better than texton experiments $F_1 >$ than most of @TextonsOnly
6.2 1006.2	CM RGB GLCM Grey	N	different training examples than @6.1 $F_1 \approx$ @6.1
8.1 1008.1	**CM RGB** **GLCM L***	N	slightly better than because of L* channel $F_1 >$ @6.1
8.2 1008.2	CM RGB GLCM L*	N	different training examples than @6.1 $F_1 \approx$ @8.2
7.1 1007.1	CM RGB GLCM Grey	**Y**	should perform better because of feature transformation $F_1 >$ @6.1
7.2 1007.2	CM RGB GLCM Grey	Y	different training examples than @6.1 $F_1 \approx$ @7.2

Table 4.3: Experiment settings: Colour Moments, GLCM

4.2.3 Textons, Colours, GLCM

These experiments investigate a combination of previous work's approaches combined with new features including colour information. The set is listed in table 4.4.

For all these runs, a min-max normalisation is performed before invoking the SVM. No SVM ensembles are utilised, and the SVMs all used the RBF kernel with different error costs for the two classes. The grid searched is C: 7..17, γ: -11..3.

Run	Features	TS	TSM	Trans	Hypothesis and expectations
21.1 1021.1	**MR8 HSI 20** **CM HSI** **GLCM Grey**	50	BA	N	colour information included, should perform better than Textons or colour moments with GLCM alone $F_1 >$ @1/@6
21.2 1021.2	see @21.1	**100**	BA	N	should perform better since more training examples $F_1 >$ @21.1
22.1 1022.1	**MR8 HSI 20** **CM HSI** **GLCM L*** **GLCM Sobel**	50	BA	N	GLCM on Sobel included, should perform better $F_1 >$ @21
22.2 1022.2	see @22.1	**100**	BA	N	should perform better since more training examples $F_1 >$ @22.1
23.1 1023.1	**MR8 40** **CM CIELab** **GLCM L*** **GLCM Sobel**	**0.1%**	**SC**	N	more textons, hence more features, should perform better $F_1 >$ @21/22
23.2 1023.2	see @23.1	0.1%	SC	N	different training examples than @23.1 $F_1 \approx$ @23.1
92.1 1092.1	**MR8 40** **CM RGB** **GLCM Grey**	50	BA	**Y**	Textons without colour information, should perform worse than @21.1 $F_1 >$ @21.1
92.2 1092.2	see @92.1	**100**	BA	Y	should perform better since more training examples $F_1 >$ @92.1
93.1 1093.1	**MR8 40** **CM RGB** **GLCM Grey** **GLCM Sobel**	50	BA	Y	GLCM on Sobel included, should perform better $F_1 >$ @92
93.2 1093.2	see @93.1	**100**	BA	Y	should perform better since more training examples $F_1 >$ @93.1

Table 4.4: Experiment settings: Textons, Colours, GLCM

4.2.4 Many Features

The experiments in table 4.5 explore the behaviour of the SVM when substantially more features are provided. All features from the GLCM on different channels, as well as all colour moments, are considered. In some experiments, Textons are added, too.

For all these runs, training examples are sampled by bootstrapping on ten experts' confidence level classes. All SVMs used the RBF kernel with different error costs for the two classes. The grid searched is C: 7..17, γ: -11..3.

Run	Features	TS	FN	Trans	E	LCF	Hypothesis and expectations
42.1 1042.1	**CM CIELab** **CM HSI** **CM RGB** **GLCM Grey** **GLCM L*** **GLCM Sobel**	50	MM	N	1	N	should perform better than experiments with less features, since the classes should be better separable with more information $F_1 >$ @runs with less features
42.2 1042.2	see @42.1	**100**	MM	N	1	N	should perform better since more training examples $F_1 >$ @42.1
71.1 1071.1	see @42.1	50	MM	N	1	N	same setting as @42.1 should perform better since more training examples $F_1 \approx$ @42.1
71.2 1071.2	see @42.1	100	MM	N	1	N	should perform better since more training examples $F_1 \approx$ @71.1/@42.1
43.1 1043.1	see @42.1	50	**ZS**	N	1	N	standardization instead of normalization; might provide for better performance since more information is conserved $F_1 >$ @42.1
43.2 1043.2	see @42.1	**100**	ZS	N	1	N	should perform better since more training examples $F_1 >$ @43.1
41.1 1041.1	see @42.1	50	MM	N	**5**	N	utilization of an ensemble of SVMs, should perform better since more confidence in classification $F_1 >$ @42.1
41.2 1041.2	see @42.1	**100**	MM	N	5	N	should perform better since more training examples $F_1 >$ @41.1
82.1 1082.1	**MR8 40** **CM CIELab** **CM HSI** **CM RGB** **GLCM Grey** **GLCM L*** **GLCM Sobel**	50	MM	N	1	N	additional Texton features, should perform better $F_1 >$ @42.1
82.2 1082.2	see @82.1	**100**	MM	N	1	N	should perform better since more training examples $F_1 >$ @82.1
81.1 1081.1	see @82.1	50	MM	**Y**	1	N	should perform better due to feature transformation $F_1 >$ @82.1
81.2 1081.2	see @82.1	**100**	MM	Y	1	N	should perform better since more training examples $F_1 >$ @81.1
94.1 1094.1	see @82.1	50	MM	Y	1	**Y**	should perform similarly even with less features, since probably only features with low information gain are removed $F_1 \approx$ @81.1
94.2 1094.2	see @82.1	**100**	MM	Y	1	Y	should perform better since more training examples $F_1 >$ @94.1
91.1 1091.1	**MR8 40** **CM CIELab Eq** **CM HSI Eq** **CM RGB Eq** **GLCM Grey Eq** **GLCM L* Eq** **GLCM Sobel Eq**	50	MM	Y	1	N	could perform better because of equalised histograms $F_1 >$ @82.1
91.2 1091.2	see @91.1	**100**	MM	Y	1	N	should perform better since more training examples $F_1 >$ @91.1

Table 4.5: Experiment settings: Many Features

4.2.5 SVM Comparison

This set of experiments focuses on different kernels and a different implementation of an SVM as shown in table 4.6. Only a few runs have been conducted, but they provide some insight nonetheless.

For all these runs, a min-max normalisation is performed before invoking the SVM. Training examples are sampled by bootstrapping on ten experts' confidence levels classes with 50 training examples sampled from each class. An ensemble of five SVMs is utilised for all runs.

Run	Features	SVM	HP	Hypothesis and expectations
31 1031	CM CIELab CM HSI CM RGB GLCM Grey GLCM L* GLCM Sobel	**Lin+DEC**	C: **-5..17**	should perform similar to @41.1 $F_1 \approx$ @41.1
32 1032	see @31	**Poly+DEC**	C: **-5..17** d: **3..7**	should perform similar to @41.1 $F_1 \approx$ @41.1
51 1051	see @31	**Lin**	C: **-5..17**	should perform similar to @41.1 $F_1 \approx$ @41.1

Table 4.6: Experiment settings: SVM Comparison

4.2.6 Oversampling, Data Cleaning

This set of experiments investigates the effect of different more advanced strategies such as applying the SMOTE algorithm and removal of Tomek links (see subsection 3.2.1). This is shown in table 4.7.

All the data in these runs is standardised according to equation (3.11). For all these runs, training examples are sampled by bootstrapping 0.05% of the data on the two class strata. All SVMs used the RBF kernel with different error costs for the two classes, an ensemble of five SVMs is utilised. The grid searched is C: 7..17, γ: -17..3.

The strategies investigated in this set of experiments are the following:

1. SMOTE: The positive training examples of each fold combined are oversampled with the SMOTE algorithm, see subsection 3.2.1. The number of new data examples created is chosen such that the new number of positives is approximately the same as the number of negative training examples.
2. SMOTE+TomekLinksRemoval: SMOTE is applied as described above, and afterwards, the Tomek links are searched and removed, see subsection 3.2.1. Since this involves a nearest neighbour search, for computational reasons, the TomekLinksRemoval step is executed only thrice. Afterwards, there are probably still Tomek links in the dataset, but experiments show an increase in performance nonetheless. The nearest neighbours are searched for the positive class only, and when a Tomek link is found, only the example of the negative class is removed from the dataset to not diminish the set of positive examples even more.

3. TomekLinkRemoval+SMOTE: This is the same procedure as SMOTE+TomekLinksRemoval, but first the Tomek links are removed, and afterwards SMOTE is applied. In the literature, both variants are used, such that a comparison seemed beneficial.
4. TomekLinkRemoval/Both+SMOTE: This is based on TomekLinkRemoval+SMOTE as well, but instead of removing only the negative example, the link's positive example is removed from the set of training examples as well. For data cleaning, this is the procedure (Batista et al., 2004) suggest.

Run	Features	LCF	Strategy	Hypothesis and expectations
63 1063	CM CIELab CM HSI CM RGB GLCM Grey GLCM L* GLCM Sobel	N	**SMOTE**	should perform better than other runs since the classes should more compact $F_1 >$ @41.1
62 1062	see @63	N	**SMOTE+ TomekLinksRemoval**	should perform better than @61 since data cleaning is applied $F_1 >$ @41.1
61 1061	see @63	N	**TomekLinksRemoval+ SMOTE**	should perform better than @63 since data cleaning is applied $F_1 >$ @62/@63
66 1066	**MR8 40** **CM CIELab** **CM HSI** **CM RGB** **GLCM Grey** **GLCM L*** **GLCM Sobel**	N	TomekLinksRemoval+ SMOTE	should perform better than other runs since more features are used $F_1 >$ @61
64 1064	see @66	**Y**	TomekLinksRemoval+ SMOTE	should perform better than other runs since the classes should expose more well-defined borders $F_1 >$ @41.1
65 1065	see @66	Y	**TomekLinksRemoval/Both+ SMOTE**	could perform worse sind the positive class is diminished even more $F_1 <$ @64

Table 4.7: Experiment settings: Oversampling, Data Cleaning

4.2.7 Hardware

Several virtual machines and a dedicated server in the University's Enterprise Lab[36] could be accessed for the experiments. The dedicated server is a Sun Fire X4600 with 8 AMD Opteron 8356 CPUs, providing for up to 32 parallel threads. 148 GB of RAM is installed, and four hard discs with 146GB each are available. The virtual machines are hosted on four Sun Fire X2270 servers with 2 Intel Xeon E5540 CPUs, and 96 GB RAM installed.

[36] see http://www.enterpriselab.ch/ (accessed 2016-06-23)

4.3 Metrics

Disclaimer: This section 4.3 extends joint work which originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

This section provides for an introduction and several intuitions on the reported metrics to enable a more thorough evaluation of the results. In all the following equations, the reported metrics are defined in terms of values from the confusion matrix, see figure 3.28. In this section, corresponding to figure 3.28: TP = true positives, TN = true negatives, FP = false positives, FN = false negatives, $Ac_+ = TP{+}FN$ = actual positives, $Ac_- = FP{+}TN$ = actual negatives, $Pr_+ = TP + FP$ = predicted positives, $Pr_- = FN + TN$ = predicted negatives.

4.3.1 Accuracy

$$\begin{aligned} \text{accuracy} &= \frac{TP + TN}{Ac_+ + Ac_-} \\ \text{error rate} &= 1 - \text{accuracy} \end{aligned} \tag{4.1}$$

Accuracy and error rate as defined in equation (4.1) are the metrics most often reported in the literature. However, due to the imbalanced available data in the theses, relying on this metric alone is problematic, as (He and Garcia, 2009) point out. One aspect is that high accuracy is achieved if the classifier simply predicts the majority class for all examples. As introduced in subsection 1.3.3, the available data is imbalanced with a ratio of 1:8 or 1:15, respectively. Hence, predicting the negative class always, yields about 90% accuracy (see section 4.4), although not a single example of the minority class is predicted correctly. Clearly, this is misleading.

Moreover, (He and Garcia, 2009) emphasise that those two metrics are sensitive to changes in data: "as class distribution varies, measures of the performance will change even though the underlying fundamental performance of the classifier does not". They attribute this to the proportion of the two rows in the confusion matrix (see figure 3.28: actual positives and actual negatives). Since they represent the class distribution, "any metric that uses values from both [rows] will be inherently sensitive to imbalances".

Accuracy is reported nonetheless since it is an intuitive metric often used in the literature.

4.3.2 Recall And Precision

$$\begin{array}{r} \text{recall} \\ \text{sensitivity} \\ \text{true positive rate (TPR)} \end{array} = \frac{TP}{TP + FN} \tag{4.2}$$

$$\text{precision} = \frac{TP}{TP + FP} \tag{4.3}$$

In equation (4.2), recall is a measure of *completeness* ("how many actually positive examples have been predicted correctly"). In equation (4.3), precision is a measure of *exactness* ("how many of the examples predicted as positive have been predicted correctly") (He and Garcia, 2009). Another intuition is a probabilistic interpretation. Recall captures the probability $P(Pr_+ \mid Ac_+)$, whereas precision represents $P(Ac_+ \mid Pr_+)$[37].

Recall and precision are reported separately, although the F_1 score combining both metrics is reported. However, since the harmonic mean might be less intuitive to interpret, precision and recall by themselves might provide more insight into the performance, e.g. when plotted into precision-recall space (see subsection 4.3.6).

4.3.3 F1 Score

$$\begin{aligned} F_1 \text{ score} &= 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{2 \cdot TP}{2 \cdot TP + FN + FP} \\ F_\beta \text{ score} &= (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} = \frac{(1 + \beta^2) \cdot TP}{(1 + \beta^2) \cdot TP + \beta^2 \cdot FN + FP} \end{aligned} \tag{4.4}$$

The F_1 score as shown in equation (4.4) is the main metric reported in the theses. F_β in equation (4.4) shows its generalisation which provides for the incorporation of a weight β on either precision or recall. Nevertheless, in this thesis, $\beta = 1$ always.

The F_1 score is the harmonic mean of recall and precision. Besides accuracy, this metric is often reported in the literature. (He and Garcia, 2009) interpret this metric "as a measure of the effectiveness of classification in terms of a ratio of the weighted importance on either recall or precision". Compared to the other two Pythagorean means, the arithmetic and the geometric mean, calculating the harmonic mean results in the lowest value of those three means. Thus, in the context of measuring the performance of a classifier, it might be interpreted as the most "conservative", or "cautious", metric: "the geometric and harmonic means penalise uneven performances, but the harmonic mean penalises them more heavily"[38].

[37] based on: http://stats.stackexchange.com/questions/7207/roc-vs-precision-and-recall-curves/7210#7210 (accessed 2016-06-17)

[38] source: http://www.cse.unsw.edu.au/~teachadmin/info/harmonic3.html (accessed 2016-06-17)

Figure 4.1: Geometric interpretation of the three Pythagorean means[39]

To provide for a comparison between the Pythagorean means, figure 4.1 shows a geometric interpretation. Note that $H \leq G \leq A$, always. Setting $a = recall$ and $b = precision$ or vice versa, links directly to the discussion of the F_1 score above.

Note that although a confusion matrix is calculated for each fold (see section 3.4), it is prohibitive to calculate the F_1 score for each fold's or each image's confusion matrix and report the arithmetic mean of all those F_1 scores as the final overall F_1 score of the whole experiment. As (Flach and Kull, 2015) observe, proceeding like this is "methodologically misguided", although they do not discuss their reasoning. (Forman and Scholz, 2010) provide an analysis of different strategies to calculate a final F_1 score when cross-validating. They conclude that the formulation in equation (4.4) "is the by far most unbiased method and should be used for computing F-measure, and [...] this distinction becomes important for greater degrees of class imbalance as well as for less accurate classifiers", because "F-measure has the drawback that it cannot be broken down into F-measures of arbitrary example subsets", as opposed to e.g. accuracy[40].

4.3.4 G Mean

$$\begin{array}{r} \text{specificity} \\ \text{true negative rate} \end{array} = \frac{TN}{TN + FP} \tag{4.5}$$

$$\text{G mean} = \sqrt{\text{sensitivity} \cdot \text{specificity}} = \sqrt{\frac{TP}{TP + FN} \cdot \frac{TN}{TN + FP}} \tag{4.6}$$

The G mean metric as defined in equation (4.6) is proposed by (Kubat and Matwin, 1997) and commented on in (He and Garcia, 2009): They interpret the metric as "degree of inductive bias in terms of a ratio of positive accuracy and negative accuracy". The term "positive accuracy" equals sensitivity (or recall), and "negative accuracy" equals specificity as defined in equation (4.5). Contrary to the F_1 score, the G mean incorporates true negatives. G mean is reported in this thesis since it might provide for interesting observations by incorporating the true negatives (which precision, recall, and F_1 score ignore).

[39] source: `http://mathworld.wolfram.com/PythagoreanMeans.html` (accessed 2016-06-17)
[40] "F-measure" and "F score" can be used interchangeably.

4.3.5 Receiver Operating Characteristics (ROC) Curves

Figure 4.2: ROC space; point A represents a perfect classifier; B predicts everything wrong (which, by negating it, provides for a perfect result nonetheless); C is the signature of a minority class classifier; D results from a majority class classifier; E is equivalent to a random classifier; F performs worse than a random classifier; G negates F's results; L_1 and L_2 are curves reported from a soft-type classifier, L_2 thereby providing for a better performance[41]

$$\text{false positive rate (FPR)} = \frac{FP}{FP + TN} \tag{4.7}$$

Although multivariate metrics like the F_1 score and the G mean provide for a more meaningful performance measure of classifiers, (He and Garcia, 2009) claim that "they are still ineffective in answering more generic questions about classification evaluations" and provide the comparison of different classifiers "over a range [of] sample distributions" as an example.

To overcome this, they suggest to utilise Receiver Operating Characteristics (ROC) curves, which plot the true positive rate (or recall) over the false positive rate as defined in equation (4.7). Note that both these metrics each consult only values from a single row of the confusion matrix, thus promising insensitivity to changes in data distributions. (He and Garcia, 2009) suggest that the ROC curve provides for a "visual representation of the relative trade-offs between the benefits (reflected by true positives) and costs (reflected by false positives) of classification in regards to data distributions".

[41] source: (He and Garcia, 2009)

Figure 4.3: Isolines for unbiased accuracy in ROC space

Hard-type classifiers reporting discrete class labels produce a single value in ROC space, while soft-type classifiers reporting continuous values can apply different thresholds on their output in order to produce a curve in ROC space. In fact, previous work (Suter, 2013; Suter et al., 2014) already applied this technique. Figure 4.2 shows interesting exemplary coordinates in ROC space. The closer a classifier to point A, the better is its performance. As (Fawcett, 2006) puts it: "classifiers appearing on the left-hand side of an ROC graph [...] may be thought of as 'conservative': they make positive classifications only with strong evidence so they make few false positive errors, but they often have low true positive rates as well. Classifiers on the upper right-hand side of an ROC graph may be thought of as 'liberal': they make positive classifications with weak evidence so they classify nearly all positives correctly, but they often have high false positive rates".

Another aspect to consider is imbalanced data. When plotting the unbiased accuracy isolines in ROC space, they are misleading when the population exposes skewed data distributions. Consider figure 4.3 showing the normally used, unbiased accuracy as defined in equation (4.8) in terms of true positive and true negative rates.

$$\text{accuracy} = \frac{TPR + 1 - FPR}{2} \tag{4.8}$$

However, since the available data in this thesis consists of imbalanced data (see subsection 1.3.3), the unbiased accuracy isolines are misleading. Hence, as (Flach, 2003) suggested, the skew ratio $c \in \mathbb{R}$ is introduced in equation (4.8), such that equation (4.9) is calculated. For $c = 1$, equation (4.8) arises.

$$\begin{aligned} c &= \frac{Ac_-}{Ac_+} \\ \text{biasedAccuracy} &= \frac{TPR + c(1 - FPR)}{1 + c} \end{aligned} \tag{4.9}$$

(a) Isolines for front sides of hands

(b) Isolines for back sides of hands

Figure 4.4: Isolines for biased accuracy in ROC space

The biased accuracy isolines with the ratios present in the evaluation sets on the front sides and the back sides of hands images (see subsection 1.3.3), respectively, are shown in figure 4.4. For example, when a majority class classifier predicts the data on the imbalanced data, in ROC space, the classifier will evaluate to zero false positive rate and zero true positive rate (no positives are predicted at all). But with the skewed accuracy isolines, this classifier is expected to reach an accuracy of about 85 to 95%. In section 4.4, this is visualised and indeed found to be true. Plotting the classifier in ROC space with unbiased accuracy isolines would have suggested a value of 50% for accuracy, which is implausible due to the skewed data.

4.3.6 Precision-Recall (PR) Curves

(He and Garcia, 2009) suggest that on highly skewed datasets, "ROC curve may provide an overly optimistic view of an algorithm's performance". To mitigate this, they propose the utilisation of Precision-Recall (PR) curves, which plots precision over recall. They correspond to each other in the sense that a "curve dominates in ROC space *if and only if* it dominates in PR space".

Although similar at first glance, PR curves as defined above do not provide for many desirable properties of ROC curves, as (Flach and Kull, 2015) observe. They give several examples, like an uninterpretable area ("the area under a PR curve takes the arithmetic mean of precision values whereas the [F_1] score applies the harmonic mean"), and suggest to transform the space to a different coordinate system such that Precision-Recall-Gain curves arise. A visualisation of the differences in ROC and PR space is shown in figure 4.5. The left-hand side shows the accuracy isolines for an unbiased accuracy ("the average of true positive and true negative rates" (Flach, 2003), see equation (4.8)). The right-hand side shows the hyperbolic isolines of the F_1 score in PR space.

A comparison between PR space and the transformation to PR-Gain space as proposed by (Flach and Kull, 2015) is shown in figure 4.6.

Nevertheless, (He and Garcia, 2009) observe that "many of the current research work in the community use PR curves for performance evaluations and comparisons". This thesis reports points in PR space, too. To indicate the issues stated above, some F_1 score isolines are plotted.

(a) Isolines for unbiased accuracy *(b) Isolines for F_1 score*

Figure 4.5: Comparison between the standard metrics in ROC and PR space

Figure 4.6: Comparison of PR space and PR-Gain space[42]

[42]source: (Flach and Kull, 2015)

4.3.7 Area Error

$$\text{area error} \quad = \quad \left| \frac{Ac_+ - Pr_+}{Ac_+ + Ac_-} \right| \quad = \quad \left| \frac{FN - FP}{Ac_+ + Ac_-} \right| \tag{4.10}$$

Finally, one last metric is reported in the thesis, which is the area error as specified in equation (4.10). This metric compares the actual eczematous area of each hands image according to its corresponding consensus diagnosis to the predicted eczematous area. Note that for this metric, whether those positive predictions are true or false, is ignored. The reasoning behind this is that certain scores have to be calculated by a dermatologist for patients suffering from hand eczema. A score often used is the *Hand Eczema Severity Index (HECSI)*, see (Held et al., 2005). Part of this score is the area affected by eczema. Thus, even if the results of the classifier are not precise, this measurement might suffice to calculate the HECSI score.

Unlike all the other metrics introduced above, the area error is not calculated on the final overall confusion matrix containing the values of all 9 folds combined, but rather on each image. The final value reported is the arithmetic mean of these ratios. The reason metrics like the F_1 score is calculated from the overall confusion matrix is that with these metrics, the performance of the machine learning algorithm should be indicated. With the area error, the perspective focuses on the individual hands image.

4.4 Baseline Classifiers

Disclaimer: This section 4.4 extends joint work which originates from a collaboration with Manuel Brun and Patrick Buchter, whose master theses are also conducted in the course of the SkinApp CTI project (see subsection 1.3.1).

To appraise the results reported from the more "sophisticated" classifiers developed in the theses, they may be compared to a baseline of simpler classifiers. The following four have been chosen, and their performance is calculated on the evaluation sets of both front sides of hands images as well as back sides:

1. CoinFlip: Predict each example's class randomly.
2. MajorityClass: Predict each example's class as the majority class.
3. MinorityClass: Predict each example's class as the minority class.
4. RealRatio: Predict each example's class randomly, but according to the ratio found in the evaluation sets (see subsection 1.3.3).

The results of all four baseline classifiers are shown in tables 4.8 and 4.9.

Classifier	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
CoinFlip	912'146	6'250'610	6'250'071	912'463	50	49.99	12.73	20.3	50	49.99	50	50	37.26
MajorityClass	0	0	12'500'681	1'824'609	87.26	0	n/a	n/a	100	0	0	0	12.74
MinorityClass	1'824'609	12'500'681	0	0	12.74	100	12.74	22.6	0	0	100	100	87.26
RealRatio	232'560	1'594'507	10'906'174	1'592'049	77.76	12.75	12.73	12.74	87.24	33.35	12.76	12.75	0.02

Table 4.8: Performance of four baseline classifiers on the 18 front sides hands images of the evaluation set; Ac_+ = 1'824'609, Ac_+ rate = 12.74%; area error is reported in percentage points

Classifier	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
CoinFlip	531'339	6'984'511	6'980'489	531'655	49.99	49.99	7.07	12.39	49.99	49.99	50.01	50.01	42.94
MajorityClass	0	0	13'965'000	1'062'994	92.93	0	n/a	n/a	100	0	0	0	7.07
MinorityClass	1'062'994	13'965'000	0	0	7.07	100	7.07	13.21	0	0	100	100	92.93
RealRatio	74'806	988'086	12'976'914	988'188	86.85	7.04	7.04	7.04	92.92	25.57	7.08	7.07	0

Table 4.9: Performance of four baseline classifiers on the 18 back sides hands images of the evaluation set; Ac_+ = 1'062'994, Ac_+ rate = 7.07%; area error is reported in percentage points

(a) Baseline classifiers for front sides *(b) Isolines for F_1 score*

Figure 4.7: Baseline classifiers in ROC space

(a) Baseline classifiers for front sides *(b) Isolines for F_1 score*

Figure 4.8: Baseline classifiers in PR space

The first three classifiers can always be consulted for any dataset and thus provide for a fair comparison to the more elaborate classifiers. The RealRatio classifier, however, exploits information from the dataset which is not available to a real classifier in this theses' evaluation settings (see section 3.4). It utilises the ratio of classes in the entire evaluation set, which is prohibitive for any real classifier: A real classifier must not access any information of each fold's test set. Since the hands images expose a large variance in said ratio (see subsection 1.3.3), the ratio of each fold's training set differs from the ratio calculated on the entire evaluation set. Thus, the performance of the RealRatio baseline classifier has to be interpreted with caution.

Some of the classification results are immediately evident, e.g. the values reported for accuracy considering the available data (see subsection 1.3.3). Nevertheless, the F_1 score reported is sometimes remarkably high, especially when considering the numbers reported in previous work (see section 2.1). However, as a whole, none of these baseline classifiers can provide for high accuracy as well as a high F_1 score and a small area error.

As expected, considering ROC space (see figure 4.2), the "CoinFlip" classifier lies on the same line as E, "MajorityClass" equals point D, "MinorityClass" equals point C and "RealRatio", like "CoinFlip", exposes a behaviour like a random classifier with respect to ROC space. As discussed in subsection 4.3.5, the baseline classifiers are visualised in ROC space with biased accuracy isolines in figure 4.7. Additionally, their performance is shown in PR space in figure 4.8. Note that the majority classifier cannot be plotted in PR space since its precision is always undefined.

Chapter 5

Results

This chapter presents the results according to the goals specified at the beginning of this thesis, see section 1.2: "Software Component" and "Improving Classification Quality".

As discussed in section 1.2, the development of the software component was excluded from this thesis. Nevertheless, a feasible architecture is presented in this chapter including a discussion on several important aspects, see section 5.1.

Improving the classification quality has been successful, which is presented in section 5.2. The experiments prepared in section 4.2 are discussed in detail concerning the research questions (see section 1.2) this thesis aims to provide an answer for.

5.1 Software Component

This section introduces the architecture proposed for providing a SkinApp web service. As already discussed in section 1.2, the service and components are not implemented in this thesis.

The components and the infrastructure are shown in figures 5.1 and 5.2. The following aspects have been considered:

- **Scalability**
  Depending on the features, algorithms and parameters chosen, the time needed for obtaining predictions for a new image might prove to be substantial. Thus, the architecture should allow for the distribution of the incoming requests to several nodes, or for parallelised calculations of the predictions.
- **Standards compliance**
  To provide a reliable and extensible service while enabling the rapid development of the application, already existing frameworks and standards should be relied upon.

Figure 5.1: Component diagram of the proposed architecture

Figure 5.2: Deployment diagram of the proposed architecture; the lower nodes are instances of the upper nodes

- **Pricing**
  Where possible, software with low or ideally no license costs should be considered.
- **Open-source**
  Open-source software should be preferred to mitigate vendor lock-in and to potentially contribute to those projects, implementing additional features in case existing features prove to be not optimally suited for the needs of SkinApp.
- **Loose coupling**
  It should be possible to develop the components as independent from each other as possible, and individual components should be exchangeable without having to rebuild the entire system.
- **Lightweightedness**
  To minimise configuration and operational cost, the components as well as the middleware should be as lightweight as possible.

These aspects are covered by the proposed architecture. All components are available as open-sourced, BSD- or Apache-type licensed products. The web service should expose a RESTful API[43], since most clients and frameworks nowadays can communicate RESTful via HTTP. Implementing the web service in Java should be beneficial, since with the help of the JAX-RS API[44], RESTful web services can be developed very rapidly. This is certainly true for other frameworks like Microsoft's .NET as well, but with Java, there apply no additional license fees for servers or development environments.

TomEE+ is recommended as the application server which hosts the web service application. It is certainly possible to run the application with a more lightweight servlet container like Tomcat, but then additional libraries and dependencies – like the JAX-RS framework – have to be deployed manually. Since the TomEE+ server is a Java EE 6 compliant server, JAX-RS is already included. Nonetheless, TomEE+ is a comparatively lightweight Java EE server compared e.g. to IBM's WebSphere application server, which would be much harder to configure and maintain.

To provide a reasonable level of decoupling as well as scalability, a middleware tier providing messaging services is established. The web service application server accepts the REST requests of the clients and delegates them to the SkinAppCore to perform the necessary calculations by utilising the messaging tier.

The standard messaging technology in Java EE is the Java Messaging Service (JMS) API[45]. TomEE+ integrates JMS. However, JMS is not particularly easy to use. Moreover, the SkinAppCore library (see subsection 1.3.4) is implemented in C++, for which it is hard to access JMS administered objects. Thus, Apache ActiveMQ[46] is suggested as middleware. It is open-sourced under

[43] see http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm (accessed 2016-06-23)
[44] see the reference implementation *Jersey*, https://jersey.java.net/ (accessed 2016-06-23)
[45] see http://docs.oracle.com/javaee/6/tutorial/doc/bnceh.html (accessed 2016-06-23)
[46] see http://activemq.apache.org/ (accessed 2016-03-23)

an Apache license and thus free to use for commercial purposes. Even though some configuration has to be done, ActiveMQ is much more lightweight than e.g. IBM's WebSphere MQ messaging solution. Most important of all, it provides interfaces and APIs to support a vast number of cross language clients, C++ via the C++ Messaging Service (CMS) among them[47]. Additionally, a RESTful service to the messaging engine is provided, such that even on the Java end of the pipeline, the utilisation of the more heavy-weight JMS API could be circumvented.

Finally, the core server is responsible for communicating with the messaging server and, most important of all, for the calculation of predictions for new hands images. The SkinAppCore library already exists, see subsection 1.3.4. Thus, only a lightweight component labelled "SkinAppCore Broker" would have to be developed which would be running as a Unix service, and whose main tasks are messaging and providing data to as well as consuming results from the SkinAppCore.

5.2 Improving Classification Quality: Summary

This section discusses the results in general and summarises the findings from the experiments conducted. For details on the sets of experiments defined in section 4.2, see section 5.3.

Tables 5.1 and 5.2 show the performance of the best run in each set of experiments, ordered descending by the F_1 score, both for front sides and back sides of hands images. The best measurements are highlighted.

Experiment setting	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
Oversampling, Data Cleaning	**89.29**	59.54	**57.69**	**58.6**	74.66	**6.37**	**4.6**
Many Features	85.88	68.69	46.33	55.34	**77.92**	11.62	6.68
SVM Comparison	80.82	**71.88**	36.98	48.84	76.83	17.88	11.56
Textons, Colours, GLCM	86.33	48	46.46	47.22	66.43	8.08	8.15
Colour Moments, GLCM	83.01	50.69	37.61	43.18	66.68	12.27	7.77
Textons Only	74.5	55.73	26.33	35.77	65.61	22.76	14.83

Table 5.1: Summary of the experiments conducted on the front sides of hands

Tables 5.3 and 5.4 show the best performance measurements achieved in all the runs, regardless of their membership in a set of experiments.

[47] see or http://activemq.apache.org/cms/ for more on this topic or http://activemq.apache.org/cms/example.html for example C++ code (accessed 2016-06-23)

Experiment setting	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
Many Features	**88.69**	63.09	**33.91**	**44.11**	75.62	9.36	7.36
Oversampling, Data Cleaning	88.17	63.48	32.68	43.15	75.61	9.95	8.26
SVM Comparison	84.8	**70.24**	27.51	39.54	**77.68**	14.09	10.84
Textons, Colours, GLCM	88.66	51.74	31.59	39.23	68.8	**8.53**	**6.95**
Textons Only	76.03	66.26	17.84	28.11	71.32	23.23	19.47
Colour Moments, GLCM	79.25	56.8	18.51	27.92	67.82	19.04	16.09

Table 5.2: Summary of the experiments conducted on the back sides of hands

Run	in experiment set	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
65	Oversampling, Data Cleaning	**89.96**	44.31	65.73	52.93	65.43	3.37	6.69
51	SVM Comparison	68.06	**85.44**	26.56	40.52	74.82	34.48	27.25
65	Oversampling, Data Cleaning	89.96	44.31	**65.73**	52.93	65.43	3.37	6.69
61	Oversampling, Data Cleaning	89.29	59.54	57.69	**58.6**	74.66	6.37	4.6
94.2	Many Features	85.88	68.69	46.33	55.34	**77.92**	11.62	6.68
65	Oversampling, Data Cleaning	89.96	44.31	65.73	52.93	65.43	**3.37**	6.69
61	Oversampling, Data Cleaning	89.29	59.54	57.69	58.6	74.66	6.37	**4.6**

Table 5.3: Summary of the experiments conducted on the front sides of hands

Run	in experiment set	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1091.2	Many Features	**89.36**	57.8	34.81	43.45	72.83	8.24	7.5
1051	SVM Comparison	74.45	**84.8**	19.69	31.96	79.04	26.33	23.33
1091.2	Many Features	89.36	57.8	**34.81**	43.45	72.83	8.24	7.5
1081.2	Many Features	88.69	63.09	33.91	**44.11**	75.62	9.36	7.36
1051	SVM Comparison	74.45	84.8	19.69	31.96	**79.04**	26.33	23.33
1091.2	Many Features	89.36	57.8	34.81	43.45	72.83	**8.24**	7.5
1022.2	Textons, Colours, GLCM	88.66	51.74	31.59	39.23	68.8	8.53	**6.95**

Table 5.4: Summary of the experiments conducted on the back sides of hands

Figure 5.3: Colour encoding on hands images corresponding to the confusion matrix

(a) Image #223, best front sides image in run 65, $F_1 = 74.32\%$

(b) Image #242, best back sides image in run 1094.1, $F_1 = 86.75\%$

Figure 5.4: Best images of all experiments

Figure 5.5: Excluded image #236, best in run 82.2, $F_1 = 91.97\%$

Figure 5.4 shows the images with the highest F_1 score over all experiments, both for front sides and back sides of images, with one exception: For the front sides of hands, image #236 (see figure 5.5) always has a high F_1 score because of the unusual eczema ratio, such that displaying this image would not provide for a good intuition about the classification quality reached by the experiments. The colour encoding scheme is displayed in figure 5.3 and is the same as in previous work (Schnürle, 2016; Suter, 2012, 2013; Suter et al., 2014).

The first remarkable observation is that indeed connected regions are predicted, and not scattered pixels all across the image. This is not necessarily self-evident since e.g. the baseline classifiers discussed in section 4.4 would show patterns similar to noise. Concerning the baseline classifiers, none of the experiments performed overall worse than one of those. Nevertheless, for singled out metrics, there are cases where a baseline classifier performs better (e.g., the "CoinFlip" classifier reached an F_1 score of 20.3%, which is not reached in run 3.2 of the "Textons Only" set).

Utilising additional features like colour moments or GLCM based features has proven to be very beneficial. The runs with such features reached higher scores than plain Texton approaches.

In all experiments, very few training examples have been utilised to train the SVM. Most experiments sample only about 0.1% of the training data; the best performing runs even consider only 0.05%. However, the experiments are inconclusive about whether more training examples would yield better performance. This might hint at the model being too simplified and thus underfitting the training data (i.e. exposing a high bias), although, with only comparatively few experiments, this cannot be claimed for certain. Nevertheless, the G mean score reached can be as high as 77.92%, hinting on the contrary at a low bias, such that both accuracies on the positive as well as on the negative class are high (Liu et al., 2006). More research should be conducted in future work with different sizes of training samples to explain these relationships.

Using an ensemble seems to provide for a small increase in classification quality, but the experiments conducted show that less than 1 percentage point is gained, which could also be caused by sampling a different subset from the training examples. More research or experiments with other ensemble settings should be conducted.

The runtime of the feature extraction from the images, training the SVM and predicting unseen images is not reported in this thesis. Exact numbers would not be meaningful since the experiments have been carried out on virtual machines and servers with multiple workflows running in parallel and at peak times generating more load on the system than CPUs were available. Additionally, depending on the memory load, KNIME by default automatically chooses to swap nodes consuming much memory to the file system, which was identified as a bottleneck and slowed down calculation time considerably. Nevertheless, one run has been started to provide at least for the following tentative numbers. The run used the front side of images, all colour moment and GLCM features, used no ensemble of SVMs and no other techniques investigated in other experiments.

1. Grid search start to end (C: 7..17, γ: -11..3), parallelisation of grid search: about 50 minutes on average for each fold.
2. Prediction of two images in the test set of a fold, parallelised to calculate the predictions for two different configurations at the same time: 94 minutes.
3. Time for LIBSVM training in a cross-validation fold: 2 minutes.
4. Time for LIBSVM prediction of two images in the test set of a fold: 10-30 minutes.

The most remarkable improvements are achieved by combining several steps and approaches in this order:

- Reduce the available training data by bootstrapping.
- Transform those features which are not or only barely linearly correlated with the class by taking their logarithm or square root.
- Standardise each example's feature.
- Scan for Tomek links and removing only the linked negative class' data example.
- Oversample the positive class via SMOTE to approximately the same size as the negative.
- Utilise an ensemble of SVMs to elicit the final prediction by majority vote.

5.3 Improving Classification Quality: Details

The subsections below report on the individual sets of experiments and are all composed of the same structure, which is described in the following and applies to front sides as well as back sides of hands images analogously. Following a discussion on the results of the experiments, each left column of figures and tables is showing the results for front sides, whereas the right column shows results for back sides of hands images.

First, two hands images are displayed. From all runs of each set of experiments, the run with the highest F_1 score is chosen. From this run, the image achieving the highest F_1 score is shown in the upper image, with the exception of excluding image #236 for reasons discussed in section 5.2. Thus, the second-best image is chosen when image #236 has the highest score. The lower image is extracted from the same run, too. But regardless of the run's performance on this image, it is always image #239, which was chosen as comparatively representative with respect to the eczema ratios in the evaluation set (see subsection 1.3.3). For the back sides of hands in the opposite column, this is image #220. The upper image aims to show the best possible performance with the specific settings in each set of experiments. The lower figure shows the same hands image predicted in different experiments, such that it might provide for comparability beyond the bounds of sets of experiments.

Afterwards, four visualisations of each run's results are plotted. Firstly, these are two plots in ROC space. The lower plot magnifies the region of interest from the upper plot, the upper displaying the whole ROC space. Biased accuracy isolines respecting the ratio of classes in the evaluation set are plotted to visualise the accuracy achieved (see subsection 4.3.5). Secondly, similar plots are displayed, but in PR space together with the hyperbolic isolines of the F_1 score (see subsection 4.3.6). In these plots, the runs are distinguished by their run ID found in the corresponding tables for experiments as well as results. Some of the runs are plotted with red, some with blue symbols. This aims to highlight pairs of runs with parameters only different in the subset of training examples chosen. This subset is equal in size for both runs, or the runs differ in the number of examples drawn. For each run, this is exactly described in section 4.2.

Next, the table with the most interesting values calculated from the confusion matrix is shown. This includes F_1 score reached, which is the primary metric this thesis reports, and several other numbers to provide for an intuition of each runs strengths and weaknesses. The highest metrics achieved are set in a bold typeface.

As a last figure, the area error is displayed. This consists of a box plot of the experts' area error compared to the consensus diagnoses. The box plot aims to show the baseline that should be reached by the predictions of the trained classifiers. The area errors of the runs are plotted in the same figure, just right to the central axis the box plot is aligned on. Thus, the x-axis is without meaning. The box plot itself remains static for all sets of experiments.

The box plots are configured such that values outside the whiskers are plotted as outliers. Additionally, the arithmetic mean of all values is plotted as a blue dot. "The plotted whisker extends to the adjacent value, which is the most extreme data value that is not an outlier"[48].

[48]source: http://www.mathworks.com/help/stats/boxplot.html (accessed 2016-03-23)

5.3.1 Textons Only

The main result of this set of experiments is that it establishes the link to previous work in (Schnürle, 2016) successfully. The F_1 score reached is comparable to previous work, underpinning the validity of the workflows implemented in this thesis. Using fewer Textons seems to negatively affect the F_1 score, up to about 3 percentage points (pp) less is reported for 20 Textons compared to 40 Textons.

Randomly choosing different training examples results in a difference in the F_1 score of approximately up to 1 pp, thus suggesting that the workflows show a robust performance when different training examples are chosen.

Contrary to expectations, utilising 50% more training examples seems not to improve prediction quality in general; on the front sides of hands, about 1-2 pp on the F_1 score are gained, whereas, on the back sides of hands, the F_1 score even drops by 4-5 pp. Choosing training examples from the ten experts' confidence level classes instead of the two classes "healthy" and "eczematous" seems beneficial, about 2 pp on the F_1 score are gained.

Enriching the MR8 filter responses with the patch's centre pixel's colour information (see subsection 3.1.3) proves to be mostly detrimental, contrary to the hypotheses proposed. For front sides of hands, adding information from the HSI colour space, the F_1 score deteriorates up to about 13 pp. For back sides, about 7 pp are lost. For the CIELab colour space, results are inconclusive: For front sides of hands, up to about 7 pp on the F_1 score are lost. For back sides of hands, 5-6 pp are gained. This cannot be explained without additional experiments.

The low F_1 score is caused by a precision much lower than recall, similar to what was reported in previous work (Schnürle, 2016). The area error is comparatively high, and both accuracy and G mean fail to reach a satisfactory level.

The most successful run utilises 40 Textons without colour information and draws training examples from the ten experts' confidence level classes. Nevertheless, adding colour information from CIELab space seems beneficial for back sides of hands.

(a) Image #1004, best in run 11.1, $F_1 = 51.85\%$

(b) Image #242, best in run 1004.1, $F_1 = 65.56\%$

(c) Image #239 in run 11.1, $F_1 = 34.87\%$

(d) Image #220 in run 1004.1, $F_1 = 42.76\%$

Figure 5.6: Experiments results: Textons Only: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.7: Experiments results: Textons Only: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.8: Experiments results: Textons Only: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1.1	71.54	55.5	23.67	33.19	64.03	26.12	18.6
1.2	71.87	54.87	23.8	33.2	63.87	25.65	18.22
5.1	68.58	60.06	22.51	32.75	64.76	30.17	22.06
5.2	67.89	**60.81**	22.21	32.54	64.74	31.08	23.04
2.1	73.8	56.74	25.89	35.55	**65.79**	23.71	15.5
2.2	72.72	56.4	24.85	34.5	65.08	24.9	16.68
11.1	74.5	55.73	26.33	**35.77**	65.61	22.76	14.83
11.2	**75.65**	52.44	**26.75**	35.43	64.38	**20.96**	**13.88**
3.1	64.05	36.86	14.4	20.71	50.07	31.98	24.01
3.2	65.27	33.62	14.01	19.78	48.47	30.12	23.53
4.1	72.81	40.05	20.69	27.29	55.75	22.41	16.6
4.2	72.65	38.05	19.94	26.16	54.37	22.3	16.5

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1001.1	77.4	53.6	16.41	25.13	65.16	20.79	16.31
1001.2	78.06	54.73	17.12	26.08	66.1	20.17	15.72
1005.1	70.72	61.85	14.13	23.01	66.45	28.61	23.71
1005.2	70.58	60.84	13.9	22.64	65.87	28.68	23.61
1002.1	67.51	60.51	12.6	20.85	64.17	31.95	26.62
1002.2	66.57	61.53	12.41	20.66	64.18	33.05	27.69
1011.1	79.43	54.35	**18.15**	27.21	66.49	18.66	14.68
1011.2	**80.71**	52.49	18.9	27.8	65.95	**17.14**	**13.31**
1003.1	62.39	51.35	9.61	16.19	56.98	36.77	31.77
1003.2	63.21	55.58	10.46	17.61	59.54	36.21	30.85
1004.1	76.03	66.26	17.84	**28.11**	71.32	23.23	19.47
1004.2	74.84	**68.67**	17.47	27.86	**71.91**	24.69	20.27

(b) back side hands images

Table 5.5: Experiments results: Textons Only

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.9: Experiments results: Textons Only: area error

5.3.2 Colour Moments, GLCM

All of the experiments in this set perform better than the "Textons Only" set for front sides of hands. However, contrary to the hypotheses proposed, on the back sides of hands, the results are ambiguous and do perform similarly to *Textons Only* experiments. This is rooted in the precision metric: For front sides of hands, precision is substantially increased, which is not observed for back sides of hands.

For front sides of hands, utilising features from the GLCM calculated on the CIELab L* channel gains up to about 5 pp on the F_1 score compared to the GLCM calculated on the greyscale image. This effect is not observed on back sides of hands images.

Feature transformation gains 5-6 pp on the F_1 score for front sides of hands – on back sides, feature transformation is detrimental, and the F_1 score loses up to about 2 pp.

The area error for front sides of hands is about the same compared to human experts. For back sides, the area error cannot be improved beyond the level of the "Textons Only" experiments.

Again, the difference between front and back sides of hands cannot be explained with these experiments, and future research should be conducted.

The most successful run for front sides of hands images utilise the colour moments from an RGB image and feature transformation together with features from a GLCM calculated on a greyscale image. For back sides, the same setting but without feature transformation performs best.

(a) Image #241, best in run 7.1, $F_1 = 63.15\%$

(b) Image #226, best in run 1006.1, $F_1 = 66.63\%$

(c) Image #239 in run 7.1, $F_1 = 28.69\%$

(d) Image #220 in run 1006.1, $F_1 = 20.3\%$

Figure 5.10: Experiments results: Textons Only: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.11: Experiments results: Colour Moments, GLCM: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.12: Experiments results: Colour Moments, GLCM: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
6.1	82.45	44.55	32.32	37.46	62.44	12.48	9.96
6.2	**83.77**	47.46	35.82	40.83	64.86	**11.37**	9.22
8.1	82.21	49.02	35.59	41.24	65.32	12.95	9.98
8.2	82.35	**51.72**	36.42	42.74	**67.01**	13.18	9.6
7.1	83.01	50.69	**37.61**	**43.18**	66.68	12.27	**7.77**
7.2	82.74	50.54	37	42.72	66.47	12.56	8.41

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1006.1	**79.25**	56.8	**18.51**	**27.92**	**67.82**	19.04	**16.09**
1006.2	78.94	55.29	17.93	27.08	66.81	19.26	16.4
1008.1	78.87	56.59	18.15	27.48	67.53	19.43	16.39
1008.2	78.38	**57.1**	17.85	27.2	67.59	20	**17.13**
1007.1	77.41	54.23	16.54	25.35	65.52	20.83	18.12
1007.2	77.39	55.9	16.86	25.91	66.46	20.98	18.04

(b) back side hands images

Table 5.6: Experiments results: Colour Moments, GLCM

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.13: Experiments results: Colour Moments, GLCM: area error

5.3.3 Textons, Colours, GLCM

These experiments provide for higher F_1 scores than the experiment sets "Textons Only" and "Colour Moments, GLCM". As in "Textons Only", using plain MR8 Textons rather than Textons enriched with colour information seems to provide for higher F_1 scores.

Contrary to hypotheses proposed, adding features from the GLCM calculated on an image pre-processed with the Sobel operator does not affect the F_1 score.

Most of the experiments show a decline in F_1 score of up to about 1 pp when twice as many examples are used for training.

In most experiments, the accuracy for both front and back sides of hands is considerably higher than 80%, whereas the G mean is not improved compared to previous experiment sets. Although higher precision is reached in this set's experiments, recall declines compared to previous experiments, resulting in a lower G mean.

The best run for front sides of hands images utilises 40 MR8 Textons, colour moments from CIELab space, GLCM on L* channel as well as on the image resulting from applying the Sobel operator. However, for back sides, 20 Textons with HSI colour information, colour moments from HSI space and the same GLCM features as for front sides perform best.

(a) Image #1024, best in run 23.1, $F_1 = 61.33\%$

(b) Image #242, best in run 1022.2, $F_1 = 80.42\%$

(c) Image #239 in run 23.1, $F_1 = 45.62\%$

(d) Image #220 in run 1022.2, $F_1 = 13.59\%$

Figure 5.14: Experiments results: Textons, Colours, GLCM: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.15: Experiments results: Textons, Colours, GLCM: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.16: Experiments results: Textons, Colours, GLCM: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
21.1	78.67	55.22	31.04	39.74	67.33	17.91	13.97
21.2	78.54	54.16	30.63	39.13	66.68	17.9	15
22.1	79.12	59.61	32.55	42.11	69.9	18.03	14.39
22.2	79.35	56.94	32.35	41.26	68.59	17.38	14.72
23.1	**86.33**	48	**46.46**	**47.22**	66.43	**8.08**	**8.15**
23.2	85.82	47.1	44.62	45.83	65.64	8.53	8.56
92.1	82.18	61.13	37.69	46.63	72.19	14.75	9.1
92.2	82.42	60.8	38.08	46.83	72.13	14.43	9.15
93.1	80.33	**65.45**	35.32	45.88	**73.48**	17.49	11.67
93.2	81.12	63.69	36.27	46.22	73	16.33	11.17

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1021.1	86.37	51.94	24.75	33.52	67.91	11.19	10.75
1021.2	88.55	50.04	28.92	36.65	67.59	8.72	8.21
1022.1	88.28	53.33	30.95	39.17	69.64	9.06	7.06
1022.2	**88.66**	51.74	**31.59**	**39.23**	68.8	**8.53**	**6.95**
1023.1	86.27	**58.13**	27.63	37.46	**71.69**	11.59	9.97
1023.2	85.35	55.8	25.51	35.01	69.91	12.4	11.14
1092.1	84.04	56.64	23.71	33.43	69.85	13.87	11.24
1092.2	84.64	54.34	24.07	33.36	68.74	13.05	10.67
1093.1	84.49	57.45	24.53	34.38	70.51	13.46	10.56
1093.2	84.39	55.54	23.96	33.48	69.34	13.42	10.74

(b) back side hands images

Table 5.7: Experiments results: Textons, Colours, GLCM

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.17: Experiments results: Textons, Colours, GLCM: area error

5.3.4 Many Features

Almost all of the experiments conducted in this set outperform previous experiments. For front sides of hands, the F_1 score is rarely below 50%, and for back sides, the F_1 score is rarely below 40%. This is mostly caused by an increased recall of between 60 - 70%. The G mean reached is higher than in most previous experiments, too. The area error is very low and exposes similar values like human experts.

Adding the MR8 Textons to the colour moments and features extracted from GLCMs results in a gain in F_1 score of up to about 2 pp.

Performing standardisation instead of normalisation does not affect the F_1 score, both methods report similar results.

Contrary to the hypotheses proposed, for the front sides of hands, utilising an ensemble of five SMVs resulted in an increase in F_1 score of only about 1 pp. For back sides of hands, this effect is not present. Even on front sides, the gain is comparatively small and may well be caused by other effects such as choosing different training examples.

Applying feature transformation results in a gain in F_1 score of up to about 1.5 pp. Removing features which are not or only barely correlated to the class results in an increase in F_1 score of up to about 3 pp for front sides of hands, whereas for back sides of hands, this effect is not very prominent.

Equalising the image's channels' histograms proves to be detrimental for front sides of hands, the F_1 score reported loses up to about 5 pp. For back sides of hands, this effect is not present.

The best setting for front sides of hands is utilising all features and performing feature transformation as well as feature selection with no ensemble of SVMs used. For back sides of hands images, the best setting is almost the same but for abstaining from feature selection.

(a) *Image #241, best in run 94.2,* $F_1 = 61.5\%$

(b) *Image #242, best in run 1081.2,* $F_1 = 83.14\%$

(c) *Image #239 in run 94.2,* $F_1 = 48.88\%$

(d) *Image #220 in run 1081.2,* $F_1 = 39.34\%$

Figure 5.18: Experiments results: Many Features: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.19: Experiments results: Many Features: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.20: Experiments results: Many Features: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
42.1	84.38	63.22	42.42	50.77	74.37	12.53	7.38
42.2	84.86	61.76	43.37	50.96	73.82	11.77	7.48
71.1	82.39	64.01	38.5	48.08	73.8	14.92	9.14
71.2	84.3	61.44	42.04	49.92	73.38	12.36	7.22
43.1	85.81	61.92	42.18	50.17	74.2	11.08	6.86
43.2	**85.97**	60.46	42.43	49.87	73.48	**10.7**	7.25
41.1	84.73	63.75	43.24	51.53	74.81	12.21	7.51
41.2	84.88	61.96	43.44	51.07	73.93	11.77	7.47
82.1	84.59	67.49	43.27	52.73	76.66	12.91	7.53
82.2	84.57	67.04	43.19	52.53	76.43	12.87	7.59
81.1	83.79	**70.59**	41.91	52.59	77.79	14.28	8.64
81.2	85.04	69.2	44.4	54.1	77.75	12.65	7.43
94.1	85.53	68.69	45.5	54.74	77.74	12.01	7.02
94.2	85.88	68.69	**46.33**	**55.34**	**77.92**	11.62	**6.68**
91.1	81.74	63.43	37.26	46.94	73.17	15.59	10.28
91.2	82.45	62.19	38.35	47.44	72.88	14.59	9.78

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1042.1	86.8	62.86	29.6	40.25	74.64	11.38	9.23
1042.2	87.25	61.14	30.19	40.42	73.87	10.76	8.64
1071.1	87.28	59.93	30.01	39.99	73.18	10.64	8.61
1071.2	86.72	60.71	29.03	39.28	73.39	11.3	9.01
1043.1	87.4	63.43	30.95	41.6	75.23	10.77	8.63
1043.2	87.34	61.18	30.38	40.6	73.93	10.67	8.52
1041.1	86.69	63.17	29.45	40.17	74.76	11.52	9.36
1041.2	86.73	62.36	29.37	39.93	74.33	11.42	9.38
1082.1	88.75	60.19	33.55	43.09	73.98	9.07	**7.07**
1082.2	88.44	60.4	32.78	42.49	73.96	9.43	7.24
1081.1	87.71	63.65	31.67	42.29	75.5	10.45	8.43
1081.2	88.69	63.09	33.91	**44.11**	75.62	9.36	7.36
1094.1	88.15	64.87	32.89	43.65	76.38	10.08	8.4
1094.2	88.2	**65.14**	33.04	43.84	**76.55**	10.05	8.38
1091.1	88.43	59.99	32.68	42.31	73.72	9.41	8.2
1091.2	**89.36**	57.8	**34.81**	43.45	72.83	**8.24**	7.5

(b) back side hands images

Table 5.8: Experiments results: Many Features

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.21: Experiments results: Many Features: area error

5.3.5 SVM Comparison

The best run in this set of experiments is utilising the linear kernel of the LIBSVM implementation. The polynomial kernel performs worse than the linear kernel. The SVM-perf implementation directly optimising for the F_1 score reaches a higher recall than the other two methods, but with less precision, such that the overall F_1 score is less than the best run.

For all runs in this set, the recall is very high, whereas precision is comparatively low, resulting in a low F_1 score. However, the accuracy, as well as G mean, are high. If future research can reduce the false positives, experiments with other kernels should perform very well.

(a) Image #223, best in run 31, $F_1 = 67.22\%$

(b) Image #242, best in run 1031, $F_1 = 87.36\%$

(c) Image #239 in run 31, $F_1 = 40.25\%$

(d) Image #220 in run 1031, $F_1 = 40.1\%$

Figure 5.22: Experiments results: SVM Comparison: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.23: Experiments results: SVM Comparison: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.24: Experiments results: SVM Comparison: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
31	**80.82**	71.88	**36.98**	**48.84**	**76.83**	**17.88**	**11.56**
32	78.22	71.77	33.45	45.63	75.38	20.84	13.95
51	68.06	**85.44**	26.56	40.52	74.82	34.48	27.25

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1031	**84.8**	70.24	**27.51**	**39.54**	77.68	**14.09**	**10.84**
1032	84.52	69.91	27.03	38.99	77.37	14.36	11.41
1051	74.45	**84.8**	19.69	31.96	**79.04**	26.33	23.33

(b) back side hands images

Table 5.9: Experiments results: SVM Comparison

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.25: Experiments results: SVM Comparison: area error

5.3.6 Oversampling, Data Cleaning

Most of the experiments in this set perform very well. When only applying SMOTE, the resulting F_1 score is the lowest for front sides of hands. However, applying SMOTE and removing Tomek links afterwards gains 11 pp in F_1 score for front sides of hands, but results in a loss of 9 pp on the back sides. This cannot be explained with the experiments conducted, and additional research is necessary to examine this behaviour in greater detail.

When first applying the removal of Tomek links and SMOTE afterwards, the highest results in this set of experiments both for front and back sides of hands is reached. Not only is the F_1 score very high, but the accuracy is, too. The area error is very low.

In these experiments, adding MR8 Textons to the other features performs slightly worse, contrary to previous experiments. Additionally, removing features which are not or only barely correlated to the class results in no performance gain.

Removing not only the negative example but the positive example of each Tomek link, too, results in a loss in F_1 score of up to about 4 pp.

The best run in this setting is refraining from adding MR8 Textons, removing Tomek links and afterwards applying SMOTE.

(a) *Image #241, best in run 61,* $F_1 = 71.15\%$

(b) *Image #242, best in run 1061,* $F_1 = 76.56\%$

(c) *Image #239 in run 61,* $F_1 = 47.72\%$

(d) *Image #226 in run 1061,* $F_1 = 71.49\%$

Figure 5.26: Experiments results: Oversampling, Data Cleaning: best images

(a) Accuracy front

(b) Accuracy back

(c) Accuracy front (magnified)

(d) Accuracy back (magnified)

Figure 5.27: Experiments results: Oversampling, Data Cleaning: accuracy

(a) F_1 score front

(b) F_1 score back

(c) F_1 score front (magnified)

(d) F_1 score back (magnified)

Figure 5.28: Experiments results: Oversampling, Data Cleaning: F_1 score

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
63	82.74	51.11	37.1	42.99	66.82	12.65	8.36
62	88.82	51.08	56.81	53.79	69.41	5.67	5.78
61	89.29	59.54	57.69	**58.6**	74.66	6.37	**4.6**
66	86.53	**67.92**	47.97	56.23	**77.86**	10.75	7.39
64	88.57	57.57	54.91	56.21	73.21	6.9	7.03
65	**89.96**	44.31	**65.73**	52.93	65.43	**3.37**	6.69

(a) front side hands images

Run	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	G mean (%)	False positive rate (%)	area error (PP)
1063	87.44	62.98	30.94	41.49	74.99	10.7	9.43
1062	79.06	**75.12**	21.69	33.67	77.21	20.64	18.2
1061	**88.17**	63.48	**32.68**	**43.15**	75.61	9.95	8.26
1066	86.22	71.43	30.06	42.32	**78.99**	12.65	10.55
1064	88.14	59.55	31.88	41.53	73.34	**9.69**	**7.86**
1065	87.94	57.57	31	40.3	72.08	9.75	8.17

(b) back side hands images

Table 5.10: Experiments results: Oversampling, Data Cleaning

(a) Area error (front side hands images)

(b) Area error (back side hands images)

Figure 5.29: Experiments results: Oversampling, Data Cleaning: area error

Chapter 6

Conclusion

This thesis ascertains that "Improving Classification Quality of Support Vector Machines on Hand Eczema Images" not only is possible in principle but even more excitingly reports a performance of substantially higher quality than it was reached in previous work. Thus, the goal and main research question on whether such an improvement is possible with methods utilising SVMs must unequivocally be answered with yes.

The goals and research questions for this thesis are the following (see section 1.2), or concern the following topics, respectively:

- Software Component (relaxed to design an architecture for a SkinApp web service)
- Improving Classification Quality

- Classification Quality
- Image Material
- Hybrid Approaches
- Features
- Area Measurements

A draft of a feasible web service architecture is suggested in this thesis. Together with the business partner, this should be revisited and discussed whether a prototype should be implemented as a proof of concept.

Concerning the features used to describe the data examples, this thesis suggests combining previous work's statistical learning approach with additional features or even refrain from using them as a whole. With newly introduced features like the colour moments or values calculated from the GLCM of the neighbourhood of each pixel in question, considerable improvements in classification quality are achieved.

The difference between the predicted area and the actual eczematous area is very low. In fact, compared to mean error when calculating this score for each expert, the SVM classifier shows similar performance.

Regarding the utilisation of ensembles of SVMs, there is a small improvement measurable, but this could well be caused by the training sample chosen. Ensembles of entirely different classifiers have not been evaluated because of time constraints.

Several scores are calculated for the available images, such as the ratio between eczematous skin to the entire hands area. The images in the hold-out set have the least markings associated, and their ratio of eczematous skin is lower compared to the images in the evaluation set. The images in the hold-out set are used as validation set during the grid search for hyperparameters, and since the 9-fold cross-validation based on these hyperparameters reports good results, it may be deduced that the images in the hold-out set with only a few markings provide for a raw material of sufficient quality. However, there has to be conducted more research before this can be claimed to be universally true.

This thesis did only conduct tentative tests with other classifiers on small subsets of data. Decision trees or k-means clustering was investigated in KNIME, but they failed to provide for a classification quality similar to the SVM. Since these have been only tentative tests, the results are not representative and thus not reported here.

Outlook

This thesis has shown considerable improvements on classification quality of SVMs. With this foundation, many different approaches might prove beneficial for future work:

1. The investigation of metaheuristics seems promising since the grid search can only account for evaluating a comparatively low number of hyperparameter combinations.
2. Since using different features proved especially beneficial for this thesis, researching more possibilities to capture the two different classes on the hands images seems beneficial.
3. More sophisticated strategies in undersampling the training data could provide for more informed examples presented to the SVM in the training stage, and thus the calculation of a better separating hyperplane may be possible.

There is much related work on the topic of this thesis, of which only an excerpt could be presented and discussed. The author is confident that future work will increase the results reported in this thesis even more. The potential for improving classification quality of Support Vector Machines on hand eczema images is certainly not exhausted yet.

List of Figures

1.1 SkinAppWeb 7

1.2 Class diagram of SkinAppCore 12

2.1 Hyperplanes on imbalanced data: Undersampling[50] 20

2.2 Hyperplanes on imbalanced data: Different error costs (DEC) and SMOTE[52] 20

2.3 Ensemble of separately trained SVM component classifiers and final classification[54] 22

2.4 Hybrid approach 23

2.5 Convolutional Neural Network with two convolutional and two fully connected layers[56] 25

2.6 Hybrid approach combining CNN with and SVM replacing the output layer[58] 25

3.1 Exemplary image #239 28

3.2 Exemplary image #239 28

3.3 Texton approach developed in previous work 29

3.4 RGB channels of image #239 30

3.5 HSI channels of image #239 31

3.6 HSI colour model[60] 31

3.7 HSI hue channel rotated by 180° 32

3.8 CIELab channels of image #239 33

3.9 CIELab colour model[62] 33

3.10 Conversions and filters 34

3.11 GLCM example 35

3.12 Nearest neighbours of a pixel in different directions.[64] 36

3.13 GLCM eczema patch 37

3.14 GLCM "lower" healthy patch 37
3.15 GLCM "upper" healthy patch (with hand lines) 37
3.16 MR8 filter bank[66] 40
3.17 The SMOTE algorithm creating a new data example, $K = 6$[68] 44
3.18 Tomek links removal[70] 44
3.19 Hard-margin SVM for training examples of two dimensions[72] 49
3.20 Sensitivity of a hard-margin SVM to outliers[74] 50
3.21 Soft-margin SVM with two different values for C[76] 51
3.22 Polynomial feature transformation 55
3.23 SVM on imbalanced data[78] 56
3.24 Evaluation settings, variant A 59
3.25 Evaluation settings, variant A, 9-fold cross-validation details 60
3.26 Evaluation settings, variant B 61
3.27 Evaluation settings, variant B, 9-fold cross-validation details 62
3.28 Confusion matrix 63
3.29 Polynomial feature transformation 64
3.30 Polynomial feature transformation 66
3.31 Polynomial feature transformation 68
3.32 Polynomial feature transformation 69

4.1 Geometric interpretation of the three Pythagorean means[80] 81
4.2 ROC space 82
4.3 Isolines for unbiased accuracy in ROC space 83
4.4 Isolines for biased accuracy in ROC space 84
4.5 Comparison between the standard metrics in ROC and PR space 85
4.6 Comparison of PR space and PR-Gain space[82] 85
4.7 Baseline classifiers in ROC space 88
4.8 Baseline classifiers in PR space 88

5.1 Component diagram of the proposed architecture 90

5.2 Deployment diagram 90
5.3 Colour encoding on hands images corresponding to the confusion matrix 94
5.4 Best images of all experiments 94
5.5 Excluded image #236, best in run 82.2, $F_1 = 91.97\%$ 94
5.6 Experiments results: Textons Only: best images 99
5.7 Experiments results: Textons Only: accuracy 100
5.8 Experiments results: Textons Only: F_1 score 100
5.9 Experiments results: Textons Only: area error 101
5.10 Experiments results: Textons Only: best images 103
5.11 Experiments results: Colour Moments, GLCM: accuracy 104
5.12 Experiments results: Colour Moments, GLCM: F_1 score 104
5.13 Experiments results: Colour Moments, GLCM: area error 105
5.14 Experiments results: Textons, Colours, GLCM: best images 107
5.15 Experiments results: Textons, Colours, GLCM: accuracy 108
5.16 Experiments results: Textons, Colours, GLCM: F_1 score 108
5.17 Experiments results: Textons, Colours, GLCM: area error 109
5.18 Experiments results: Many Features: best images 111
5.19 Experiments results: Many Features: accuracy 112
5.20 Experiments results: Many Features: F_1 score 112
5.21 Experiments results: Many Features: area error 113
5.22 Experiments results: SVM Comparison: best images 115
5.23 Experiments results: SVM Comparison: accuracy 116
5.24 Experiments results: SVM Comparison: F_1 score 116
5.25 Experiments results: SVM Comparison: area error 117
5.26 Experiments results: Oversampling, Data Cleaning: best images 119
5.27 Experiments results: Oversampling, Data Cleaning: accuracy 120
5.28 Experiments results: Oversampling, Data Cleaning: F_1 score 120
5.29 Experiments results: Oversampling, Data Cleaning: area error 121

A.1 Wrongly labelled diagnosis 134

List of Tables

1.1 List of the available images 10

1.2 List of the available images with their skin pixels 10

3.1 GLCM values (greyscale) 39

3.2 GLCM values (CIELAB L* channel) 39

3.3 GLCM values (Sobel) 39

3.4 GLCM values (Sobel on equalised histogram) 40

3.5 Correlation coefficients for the examples of image #239 47

3.6 Assignment of hands images to the folds 63

4.1 Features chosen for the experiments 72

4.2 Experiment settings: Textons Only 74

4.3 Experiment settings: Colour Moments, GLCM 74

4.4 Experiment settings: Textons, Colours, GLCM 75

4.5 Experiment settings: Many Features 76

4.6 Experiment settings: SVM Comparison 77

4.7 Experiment settings: Oversampling, Data Cleaning 78

4.8 Baseline classifiers performance on front sides 87

4.9 Baseline classifiers performance on back sides 87

5.1 Summary of the experiments conducted on the front sides of hands 92

5.2 Summary of the experiments conducted on the back sides of hands 93

5.3 Summary of the experiments conducted on the front sides of hands 93

5.4 Summary of the experiments conducted on the back sides of hands 93

5.5 Experiments results: Textons Only 101

5.6 Experiments results: Colour Moments, GLCM 105

5.7 Experiments results: Textons, Colours, GLCM 109

5.8 Experiments results: Many Features 113

5.9 Experiments results: SVM Comparison 117

5.10 Experiments results: Oversampling, Data Cleaning 121

C.1 Experiments results details: Textons Only (front side hands images) 142

C.2 Experiments results details: Textons Only (back side hands images) 142

C.3 Experiments results details: Colour Moments, GLCM (front side hands images) 143

C.4 Experiments results details: Colour Moments, GLCM (back side hands images) 143

C.5 Experiments results details: Textons, Colours, GLCM (front side hands images) 144

C.6 Experiments results details: Textons, Colours, GLCM (back side hands images) 144

C.7 Experiments results details: Many Features (front side hands images) 145

C.8 Experiments results details: Many Features (back side hands images) 145

C.9 Experiments results details: SVM Comparison (front side hands images) 146

C.10 Experiments results details: SVM Comparison (back side hands images) 146

C.11 Experiments results details: Oversampling, Data Cleaning (front side hands images) 147

C.12 Experiments results details: Oversampling, Data Cleaning (back side hands images) 147

Listings

A.1 Delete data of certain users 133
A.2 Determine experts shown duplicated images 135
A.3 Delete illicit markings of experts 136
B.1 libsvm-train.sh 137
B.2 libsvm-predict.sh 138
B.3 svmperf-train.sh 138
B.4 svmperf-predict.sh 139

A SkinAppWeb Data Cleaning

A.1 Delete Users

```
-- administrative users
DELETE FROM markings WHERE user_id IN (SELECT id FROM users WHERE admin = 1);
DELETE FROM users WHERE admin = 1;
-- test users
DELETE FROM markings WHERE user_id IN (SELECT id FROM users WHERE users.fullname IN ('Monitoring␣(don\'t delete)',␣'***** ***** (USER)',␣'***** ***** (USER)',␣'***** ***** (USER)',␣'***** ***** (USER)',␣'***** ***** (USER)',␣'***** *****',␣'***** *****',␣'***** *****',␣'***** *****'));
DELETE␣FROM␣users␣WHERE␣users.fullname␣IN␣('Monitoring (don\'t␣delete)', '*****␣*****␣(USER)', '*****␣*****␣(USER)', '*****␣*****␣(USER)', '*****␣*****␣(USER)', '*****␣*****␣(USER)', '*****␣*****', '*****␣*****', '*****␣*****', '*****␣*****');
-- unusable markings
DELETE FROM markings WHERE user_id IN (SELECT id FROM users WHERE users.fullname IN ('*****␣*****', '*****␣*****', '*****␣*****'));
DELETE FROM users WHERE users.fullname IN ('*****␣*****', '*****␣*****', '*****␣*****');
-- experts that did not participate
DELETE FROM markings WHERE user_id NOT IN (SELECT * FROM (SELECT user_id FROM markings WHERE baseMarking != '' GROUP BY user_id) AS temp);
DELETE FROM users WHERE id NOT IN (SELECT user_id FROM markings);
-- experts that provided empty markings only
DELETE FROM markings WHERE user_id = (SELECT id FROM users WHERE fullname = '*****␣*****');
DELETE FROM users WHERE fullname = '*****␣*****';
```

Listing A.1: Delete data of certain users (names have been masked to provide for privacy)

For an example of unusable markings, see figure A.1.

Figure A.1: Wrongly labelled diagnosis

A.2 Determine Experts Having Seen Duplicated Images

```
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%X887X69D.jpg"
UNION
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YX8XX5XD.jpg"
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%X887X69P.jpg"
UNION
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YX8XX5XP.jpg"
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YY68Y75D.jpg"
UNION
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YY8X5X8D.jpg"
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YY68Y75P.jpg"
UNION
SELECT * FROM markings LEFT JOIN images ON markings.image_id = images.id
    LEFT JOIN users ON markings.user_id = users.id WHERE images.url LIKE
    "%YY8X5X8P.jpg"
```

Listing A.2: Determine experts shown duplicated images

A.3 Delete Illicit Markings

1. Users #424, #622, #820 have seen both images #229 and #235; transfer other users' markings from image #235 to image #229.
2. Users #457, #688 have seen both images #239 and #245; transfer other users' markings from image #245 to image #239.
3. User #666 has seen both images #246 and #237; transfer other users' markings from image #237 to image #246.
4. Users #534, #578 have seen both images #227 and #247; transfer other users' markings from image #247 to image #227.

```
1  DELETE FROM markings WHERE id IN (428, 627, 829, 430, 632, 825);
2  UPDATE markings SET image_id = 229 WHERE image_id = 235;
3  DELETE FROM images WHERE id = 235;
4
5  DELETE FROM markings WHERE id IN (458, 698, 462, 692);
6  UPDATE markings SET image_id = 239 WHERE image_id = 245;
7  DELETE FROM images WHERE id = 245;
8
9  DELETE FROM markings WHERE id IN (672, 671);
10 UPDATE markings SET image_id = 246 WHERE image_id = 237;
11 DELETE FROM images WHERE id = 237;
12
13 DELETE FROM markings WHERE id IN (541, 581, 535, 587);
14 UPDATE markings SET image_id = 227 WHERE image_id = 247;
15 DELETE FROM images WHERE id = 247;
```

Listing A.3: Delete illicit markings of experts

B KNIME SVM Bash Scripts

See section 3.5 for an introduction where the following scripts are used.

B.1 LIBSVM train

```
#!/bin/sh

RANDOMNUMBER=$1
BASEPATH=/home/taschnue/Downloads
BASETEMP=$2

$BASEPATH/csv2libsvm.py $BASETEMP/$RANDOMNUMBER-libsvm.train.csv
    $BASETEMP/$RANDOMNUMBER-libsvm.train 0 True && echo "done" >
    $BASETEMP/$RANDOMNUMBER-libsvm.train.start && $BASEPATH/libsvm-3.21/
    svm-train ${@:3} $BASETEMP/$RANDOMNUMBER-libsvm.train $BASETEMP/
    $RANDOMNUMBER-libsvm.train.model && echo "done" > $BASETEMP/
    $RANDOMNUMBER-libsvm.train.done && rm $BASETEMP/$RANDOMNUMBER-libsvm.
    train.csv && sync && rm $BASETEMP/$RANDOMNUMBER-libsvm.train && echo
    "done" > $BASETEMP/$RANDOMNUMBER-libsvm.done
```

Listing B.1: libsvm-train.sh

B.2 LIBSVM predict

```
1 #!/bin/sh
2
3 RANDOMNUMBER=$1
4 BASEPATH=/home/taschnue/Downloads
5 BASETEMP=$2
6
7 $BASEPATH/csv2libsvm.py $BASETEMP/$RANDOMNUMBER-libsvm.test.csv $BASETEMP
      /$RANDOMNUMBER-libsvm.test 0 True && echo "done" > $BASETEMP/
      $RANDOMNUMBER-libsvm.predicted.start && $BASEPATH/libsvm-3.21/svm-
      predict $BASETEMP/$RANDOMNUMBER-libsvm.test $BASETEMP/$RANDOMNUMBER-
      libsvm.train.model $BASETEMP/$RANDOMNUMBER-libsvm.predicted && echo "
      done" > $BASETEMP/$RANDOMNUMBER-libsvm.predict.done && sync && rm
      $BASETEMP/$RANDOMNUMBER-libsvm.test.csv && rm $BASETEMP/$RANDOMNUMBER
      -libsvm.test && echo "done" > $BASETEMP/$RANDOMNUMBER-libsvm.
      predicted.done
```

Listing B.2: libsvm-predict.sh

B.3 SVMPerf train

```
1 #!/bin/sh
2
3 RANDOMNUMBER=$1
4 BASEPATH=/home/taschnue/Downloads
5 BASETEMP=$2
6
7 $BASEPATH/csv2libsvm.py $BASETEMP/$RANDOMNUMBER-libsvm.train.csv
      $BASETEMP/$RANDOMNUMBER-libsvm.train 0 True && cat $BASETEMP/
      $RANDOMNUMBER-libsvm.train | sed "s/^1/-1/g" > $BASETEMP/
      $RANDOMNUMBER-svmperf.train && sed -i "s/^255/+1/g" $BASETEMP/
      $RANDOMNUMBER-svmperf.train && echo "done" > $BASETEMP/$RANDOMNUMBER-
      libsvm.train.start && $BASEPATH/svmperf/svm_perf_learn ${@:3}
      $BASETEMP/$RANDOMNUMBER-svmperf.train $BASETEMP/$RANDOMNUMBER-svmperf
      .train.model && echo "done" > $BASETEMP/$RANDOMNUMBER-libsvm.train.
      done && sync && rm $BASETEMP/$RANDOMNUMBER-libsvm.train.csv && rm
      $BASETEMP/$RANDOMNUMBER-libsvm.train && rm $BASETEMP/$RANDOMNUMBER-
      svmperf.train && echo "done" > $BASETEMP/$RANDOMNUMBER-libsvm.done
```

Listing B.3: svmperf-train.sh

B.4 SVMperf predict

```
1 #!/bin/sh
2
3 RANDOMNUMBER=$1
4 BASEPATH=/home/taschnue/Downloads
5 BASETEMP=$2
6
7 $BASEPATH/csv2libsvm.py $BASETEMP/$RANDOMNUMBER-libsvm.test.csv $BASETEMP
      /$RANDOMNUMBER-libsvm.test 0 True && cat $BASETEMP/$RANDOMNUMBER-
      libsvm.test | sed "s/^1/-1/g" > $BASETEMP/$RANDOMNUMBER-svmperf.test
      && sed -i "s/^255/+1/g" $BASETEMP/$RANDOMNUMBER-svmperf.test && echo
      "done" > $BASETEMP/$RANDOMNUMBER-libsvm.predicted.start && $BASEPATH/
      svmperf/svm_perf_classify $BASETEMP/$RANDOMNUMBER-svmperf.test
      $BASETEMP/$RANDOMNUMBER-svmperf.train.model $BASETEMP/$RANDOMNUMBER-
      svmperf.predicted && cp $BASETEMP/$RANDOMNUMBER-svmperf.predicted
      $BASETEMP/$RANDOMNUMBER-libsvm.predicted && sed -i "s/^-.*$/NEG/g"
      $BASETEMP/$RANDOMNUMBER-libsvm.predicted && sed -i "s/^[^N].*$/255/g"
       $BASETEMP/$RANDOMNUMBER-libsvm.predicted && sed -i "s/^NEG$/1/g"
      $BASETEMP/$RANDOMNUMBER-libsvm.predicted && echo "done" > $BASETEMP/
      $RANDOMNUMBER-libsvm.predict.done && sync && rm $BASETEMP/
      $RANDOMNUMBER-libsvm.test.csv && rm $BASETEMP/$RANDOMNUMBER-libsvm.
      test && rm $BASETEMP/$RANDOMNUMBER-svmperf.test && echo "done" >
      $BASETEMP/$RANDOMNUMBER-libsvm.predicted.done
```

Listing B.4: svmperf-predict.sh

C Experiments Details

This chapter provides the exact numbers of the confusion matrices over all test runs conducted (see section 4.2), such that additional metrics could be calculated which are not reported in this thesis.

C.1 Textons Only

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1.1	1'012'625	3'265'541	9'235'140	811'984	71.54	55.5	23.67	33.19	73.88	64.03	26.12	29.86	18.6
1.2	1'001'159	3'206'039	9'294'642	823'450	71.87	54.87	23.8	33.2	74.35	63.87	25.65	29.37	18.22
5.1	1'095'836	3'772'035	8'728'646	728'773	68.58	60.06	22.51	32.75	69.83	64.76	30.17	33.98	22.06
5.2	1'109'534	3'885'140	8'615'541	715'075	67.89	60.81	22.21	32.54	68.92	64.74	31.08	34.87	23.04
2.1	1'035'224	2'964'044	9'536'637	789'385	73.8	56.74	25.89	35.55	76.29	65.79	23.71	27.92	15.5
2.2	1'029'057	3'112'685	9'387'996	795'552	72.72	56.4	24.85	34.5	75.1	65.08	24.9	28.91	16.68
11.1	1'016'929	2'844'916	9'655'765	807'680	74.5	55.73	26.33	35.77	77.24	65.61	22.76	26.96	14.83
11.2	956'829	2'620'556	9'880'125	867'780	75.65	52.44	26.75	35.43	79.04	64.38	20.96	24.97	13.88
3.1	672'491	3'998'336	8'502'345	1'152'118	64.05	36.86	14.4	20.71	68.02	50.07	31.98	32.61	24.01
3.2	613'488	3'764'674	8'736'007	1'211'121	65.27	33.62	14.01	19.78	69.88	48.47	30.12	30.56	23.53
4.1	730'773	2'801'006	9'699'675	1'093'836	72.81	40.05	20.69	27.29	77.59	55.75	22.41	24.65	16.6
4.2	694'262	2'788'171	9'712'510	1'130'347	72.65	38.05	19.94	26.16	77.7	54.37	22.3	24.31	16.5

Table C.1: Experiments results details: Textons Only (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1001.1	569'806	2'902'674	11'062'326	493'188	77.4	53.6	16.41	25.13	79.21	65.16	20.79	23.11	16.31
1001.2	581'725	2'816'478	11'148'522	481'269	78.06	54.73	17.12	26.08	79.83	66.1	20.17	22.61	15.72
1005.1	657'500	3'995'260	9'969'740	405'494	70.72	61.85	14.13	23.01	71.39	66.45	28.61	30.96	23.71
1005.2	646'770	4'004'933	9'960'067	416'224	70.58	60.84	13.9	22.64	71.32	65.87	28.68	30.95	23.61
1002.1	643'196	4'462'241	9'502'759	419'798	67.51	60.51	12.6	20.85	68.05	64.17	31.95	33.97	26.62
1002.2	654'083	4'615'529	9'349'471	408'911	66.57	61.53	12.41	20.66	66.95	64.18	33.05	35.07	27.69
1011.1	577'772	2'605'464	11'359'536	485'222	79.43	54.35	18.15	27.21	81.34	66.49	18.66	21.18	14.68
1011.2	557'939	2'393'534	11'571'466	505'055	80.71	52.49	18.9	27.8	82.86	65.95	17.14	19.64	13.31
1003.1	545'849	5'135'350	8'829'650	517'145	62.39	51.35	9.61	16.19	63.23	56.98	36.77	37.8	31.77
1003.2	590'821	5'056'718	8'908'282	472'173	63.21	55.58	10.46	17.61	63.79	59.54	36.21	37.58	30.85
1004.1	704'352	3'243'872	10'721'128	358'642	76.03	66.26	17.84	28.11	76.77	71.32	23.23	26.27	19.47
1004.2	729'920	3'447'378	10'517'622	333'074	74.84	68.67	17.47	27.86	75.31	71.91	24.69	27.8	20.27

Table C.2: Experiments results details: Textons Only (back side hands images)

C.2 Colour Moments, GLCM

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
6.1	1'013'619	2'122'600	14'887'468	1'261'525	82.45	44.55	32.32	37.46	87.52	62.44	12.48	16.26	9.96
6.2	1'079'783	1'934'605	15'075'463	1'195'361	83.77	47.46	35.82	40.83	88.63	64.86	11.37	15.63	9.22
8.1	894'349	1'618'717	10'881'964	930'260	82.21	49.02	35.59	41.24	87.05	65.32	12.95	17.54	9.98
8.2	943'747	1'647'523	10'853'158	880'862	82.35	51.72	36.42	42.74	86.82	67.01	13.18	18.09	9.6
7.1	924'825	1'534'219	10'966'462	899'784	83.01	50.69	37.61	43.18	87.73	66.68	12.27	17.17	7.77
7.2	922'089	1'570'307	10'930'374	902'520	82.74	50.54	37	42.72	87.44	66.47	12.56	17.4	8.41

Table C.3: Experiments results details: Colour Moments, GLCM (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1006.1	603'827	2'658'885	11'306'115	459'167	79.25	56.8	18.51	27.92	80.96	67.82	19.04	21.71	16.09
1006.2	587'712	2'689'790	11'275'210	475'282	78.94	55.29	17.93	27.08	80.74	66.81	19.26	21.81	16.4
1008.1	601'596	2'713'385	11'251'615	461'398	78.87	56.59	18.15	27.48	80.57	67.53	19.43	22.06	16.39
1008.2	606'942	2'792'505	11'172'495	456'052	78.38	57.1	17.85	27.2	80	67.59	20	22.62	17.13
1007.1	576'411	2'908'430	11'056'570	486'583	77.41	54.23	16.54	25.35	79.17	65.52	20.83	23.19	18.12
1007.2	594'183	2'929'364	11'035'636	468'811	77.39	55.9	16.86	25.91	79.02	66.46	20.98	23.45	18.04

Table C.4: Experiments results details: Colour Moments, GLCM (back side hands images)

C.3 Textons, Colours, GLCM

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
21.1	1'007'572	2'238'671	10'262'010	817'037	78.67	55.22	31.04	39.74	82.09	67.33	17.91	22.66	13.97
21.2	988'161	2'238'028	10'262'653	836'448	78.54	54.16	30.63	39.13	82.1	66.68	17.9	22.52	15
22.1	1'087'591	2'253'389	10'247'292	737'018	79.12	59.61	32.55	42.11	81.97	69.9	18.03	23.32	14.39
22.2	1'038'980	2'172'672	10'328'009	785'629	79.35	56.94	32.35	41.26	82.62	68.59	17.38	22.42	14.72
23.1	875'868	1'009'502	11'491'179	948'741	86.33	48	46.46	47.22	91.92	66.43	8.08	13.16	8.15
23.2	859'469	1'066'577	11'434'104	965'140	85.82	47.1	44.62	45.83	91.47	65.64	8.53	13.45	8.56
92.1	1'115'449	1'844'040	10'656'641	709'160	82.18	61.13	37.69	46.63	85.25	72.19	14.75	20.66	9.1
92.2	1'109'424	1'803'618	10'697'063	715'185	82.42	60.8	38.08	46.83	85.57	72.13	14.43	20.33	9.15
93.1	1'194'121	2'186'614	10'314'067	630'488	80.33	65.45	35.32	45.88	82.51	73.48	17.49	23.6	11.67
93.2	1'162'020	2'041'792	10'458'889	662'589	81.12	63.69	36.27	46.22	83.67	73	16.33	22.36	11.17

Table C.5: Experiments results details: Textons, Colours, GLCM (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1022.1	566'909	1'264'999	12'700'001	496'085	88.28	53.33	30.95	39.17	90.94	69.64	9.06	12.19	7.06
1022.2	550'011	1'191'326	12'773'674	512'983	88.66	51.74	31.59	39.23	91.47	68.8	8.53	11.59	6.95
1021.1	803'953	2'444'448	19'399'796	744'023	86.37	51.94	24.75	33.52	88.81	67.91	11.19	13.89	10.75
1021.2	774'620	1'904'113	19'940'131	773'356	88.55	50.04	28.92	36.65	91.28	67.59	8.72	11.45	8.21
1023.1	617'950	1'618'501	12'346'499	445'044	86.27	58.13	27.63	37.46	88.41	71.69	11.59	14.88	9.97
1023.2	593'099	1'731'832	12'233'168	469'895	85.35	55.8	25.51	35.01	87.6	69.91	12.4	15.47	11.14
1092.1	602'103	1'937'423	12'027'577	460'891	84.04	56.64	23.71	33.43	86.13	69.85	13.87	16.9	11.24
1092.2	577'658	1'822'690	12'142'310	485'336	84.64	54.34	24.07	33.36	86.95	68.74	13.05	15.97	10.67
1093.1	610'646	1'879'060	12'085'940	452'348	84.49	57.45	24.53	34.38	86.54	70.51	13.46	16.57	10.56
1093.2	590'355	1'873'701	12'091'299	472'639	84.39	55.54	23.96	33.48	86.58	69.34	13.42	16.4	10.74

Table C.6: Experiments results details: Textons, Colours, GLCM (back side hands images)

C.4 Many Features

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
42.1	1'153'571	1'566'085	10'934'596	671'038	84.38	63.22	42.42	50.77	87.47	74.37	12.53	18.98	7.38
42.2	1'126'962	1'471'298	11'029'383	697'647	84.86	61.76	43.37	50.96	88.23	73.82	11.77	18.14	7.48
71.1	1'167'979	1'865'700	10'634'981	656'630	82.39	64.01	38.5	48.08	85.08	73.8	14.92	21.18	9.14
71.2	1'120'976	1'545'461	10'955'220	703'633	84.3	61.44	42.04	49.92	87.64	73.38	12.36	18.61	7.22
43.1	1'258'167	1'724'966	13'848'185	773'823	85.81	61.92	42.18	50.17	88.92	74.2	11.08	16.94	6.86
43.2	1'228'545	1'666'659	13'906'492	803'445	85.97	60.46	42.43	49.87	89.3	73.48	10.7	16.45	7.25
41.1	1'163'145	1'526'678	10'974'003	661'464	84.73	63.75	43.24	51.53	87.79	74.81	12.21	18.78	7.51
41.2	1'130'490	1'471'716	11'028'965	694'119	84.88	61.96	43.44	51.07	88.23	73.93	11.77	18.17	7.47
82.1	1'231'427	1'614'358	10'886'323	593'182	84.59	67.49	43.27	52.73	87.09	76.66	12.91	19.87	7.53
82.2	1'223'184	1'609'120	10'891'561	601'425	84.57	67.04	43.19	52.53	87.13	76.43	12.87	19.77	7.59
81.1	1'287'934	1'785'121	10'715'560	536'675	83.79	70.59	41.91	52.59	85.72	77.79	14.28	21.45	8.64
81.2	1'262'636	1'580'908	10'919'773	561'973	85.04	69.2	44.4	54.1	87.35	77.75	12.65	19.85	7.43
94.1	1'253'336	1'501'499	10'999'182	571'273	85.53	68.69	45.5	54.74	87.99	77.74	12.01	19.23	7.02
94.2	1'253'383	1'452'081	11'048'600	571'226	85.88	68.69	46.33	55.34	88.38	77.92	11.62	18.89	6.68
91.1	1'157'340	1'949'001	10'551'680	667'269	81.74	63.43	37.26	46.94	84.41	73.17	15.59	21.68	10.28
91.2	1'134'692	1'824'163	10'676'518	689'917	82.45	62.19	38.35	47.44	85.41	72.88	14.59	20.65	9.78

Table C.7: Experiments results details: Many Features (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1042.1	668'204	1'588'908	12'376'092	394'790	86.8	62.86	29.6	40.25	88.62	74.64	11.38	15.02	9.23
1042.2	649'947	1'503'057	12'461'943	413'047	87.25	61.14	30.19	40.42	89.24	73.87	10.76	14.33	8.64
1071.1	637'102	1'486'168	12'478'832	425'892	87.28	59.93	30.01	39.99	89.36	73.18	10.64	14.13	8.61
1071.2	645'383	1'577'417	12'387'583	417'611	86.72	60.71	29.03	39.28	88.7	73.39	11.3	14.79	9.01
1043.1	674'234	1'504'017	12'460'983	388'760	87.4	63.43	30.95	41.6	89.23	75.23	10.77	14.49	8.63
1043.2	650'345	1'490'496	12'474'504	412'649	87.34	61.18	30.38	40.6	89.33	73.93	10.67	14.25	8.52
1041.1	671'528	1'608'827	12'356'173	391'466	86.69	63.17	29.45	40.17	88.48	74.76	11.52	15.17	9.36
1041.2	662'914	1'594'538	12'370'462	400'080	86.73	62.36	29.37	39.93	88.58	74.33	11.42	15.02	9.38
1082.1	639'850	1'267'063	12'697'937	423'144	88.75	60.19	33.55	43.09	90.93	73.98	9.07	12.69	7.07
1082.2	642'017	1'316'780	12'648'220	420'977	88.44	60.4	32.78	42.49	90.57	73.96	9.43	13.03	7.24
1081.1	676'619	1'459'959	12'505'041	386'375	87.71	63.65	31.67	42.29	89.55	75.5	10.45	14.22	8.43
1081.2	670'613	1'307'059	12'657'941	392'381	88.69	63.09	33.91	44.11	90.64	75.62	9.36	13.16	7.36
1094.1	689'562	1'407'135	12'557'865	373'432	88.15	64.87	32.89	43.65	89.92	76.38	10.08	13.95	8.4
1094.2	692'426	1'403'346	12'561'654	370'568	88.2	65.14	33.04	43.84	89.95	76.55	10.05	13.95	8.38
1091.1	637'670	1'313'703	12'651'297	425'324	88.43	59.99	32.68	42.31	90.59	73.72	9.41	12.98	8.2
1091.2	614'456	1'150'751	12'814'249	448'538	89.36	57.8	34.81	43.45	91.76	72.83	8.24	11.75	7.5

Table C.8: Experiments results details: Many Features (back side hands images)

C.5 SVM Comparison

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
31	1'311'485	2'234'658	10'266'023	513'124	80.82	71.88	36.98	48.84	82.12	76.83	17.88	24.75	11.56
32	1'309'589	2'605'367	9'895'314	515'020	78.22	71.77	33.45	45.63	79.16	75.38	20.84	27.33	13.95
51	1'558'903	4'310'343	8'190'338	265'706	68.06	85.44	26.56	40.52	65.52	74.82	34.48	40.97	27.25

Table C.9: Experiments results details: SVM Comparison (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1031	746'643	1'967'189	11'997'811	316'351	84.8	70.24	27.51	39.54	85.91	77.68	14.09	18.06	10.84
1032	743'136	2'005'849	11'959'151	319'858	84.52	69.91	27.03	38.99	85.64	77.37	14.36	18.29	11.41
1051	901'471	3'677'547	10'287'453	161'523	74.45	84.8	19.69	31.96	73.67	79.04	26.33	30.47	23.33

Table C.10: Experiments results details: SVM Comparison (back side hands images)

C.6 Oversampling, Data Cleaning

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
63	932'477	1'580'839	10'919'842	892'132	82.74	51.11	37.1	42.99	87.35	66.82	12.65	17.54	8.36
62	931'995	708'625	11'792'056	892'614	88.82	51.08	56.81	53.79	94.33	69.41	5.67	11.45	5.78
61	1'086'383	796'607	11'704'074	738'226	89.29	59.54	57.69	58.6	93.63	74.66	6.37	13.14	4.6
66	1'239'275	1'344'099	11'156'582	585'334	86.53	67.92	47.97	56.23	89.25	77.86	10.75	18.03	7.39
64	1'050'380	862'508	11'638'173	774'229	88.57	57.57	54.91	56.21	93.1	73.21	6.9	13.35	7.03
65	808'410	421'397	12'079'284	1'016'199	89.96	44.31	65.73	52.93	96.63	65.43	3.37	8.58	6.69

Table C.11: Experiments results details: Oversampling, Data Cleaning (front side hands images)

Run	TP	FP	TN	FN	Accuracy (%)	Recall (%)	Precision (%)	F_1 score (%)	Specificity (%)	G mean (%)	False positive rate (%)	Pr_+ rate (%)	area error (PP)
1063	669'478	1'494'321	12'470'679	393'516	87.44	62.98	30.94	41.49	89.3	74.99	10.7	14.4	9.43
1062	798'503	2'882'206	11'082'794	264'491	79.06	75.12	21.69	33.67	79.36	77.21	20.64	24.49	18.2
1061	674'813	1'390'059	12'574'941	388'181	88.17	63.48	32.68	43.15	90.05	75.61	9.95	13.74	8.26
1066	759'335	1'766'508	12'198'492	303'659	86.22	71.43	30.06	42.32	87.35	78.99	12.65	16.81	10.55
1064	633'028	1'352'856	12'612'144	429'966	88.14	59.55	31.88	41.53	90.31	73.34	9.69	13.21	7.86
1065	611'966	1'361'963	12'603'037	451'028	87.94	57.57	31	40.3	90.25	72.08	9.75	13.14	8.17

Table C.12: Experiments results details: Oversampling, Data Cleaning (back side hands images)

Bibliography

R. Akbani, S. Kwek, and N. Japkowicz. Applying support vector machines to imbalanced datasets. In *Proceedings of the European Conference on Machine Learning*, pages 39–50. Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-540-30115-8_7. URL http://link.springer.com/content/pdf/10.1007%2F978-3-540-30115-8_7.pdf.

N. K. Al Abbadi, N. S. Dahir, M. A. AL-Dhalimi, and H. Restom. Psoriasis detection using skin color and texture features. *Journal of Computer Science*, 6(6):648–652, 2010. ISSN 1549-3636. doi: 10.3844/jcssp.2010.648.652.

M. S. Arifin, M. G. Kibria, A. Firoze, M. A. Amini, and H. Yan. Dermatological disease diagnosis using color-skin images. 5:1675–1680, 2012. doi: 10.1109/ICMLC.2012.6359626. URL http://ieeexplore.ieee.org/iel5/6338447/6359622/06359626.pdf?arnumber=6359626.

R. Arnold, F. Gutzwiller, T. Koller, R. Meier, M. Pouly, C. Suter, and A. Navarini. Detection and quantification of hand eczema by visible spectrum skin pattern analysis: Internal project proposal, 2012.

D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007735. URL http://dl.acm.org/ft_gateway.cfm?id=1007735&type=pdf.

R. Batuwita and V. Palade. Class imbalance learning methods for support vector machines. In H. He and Y. Ma, editors, *Imbalanced learning*, pages 83–99. John Wiley & Sons, Inc, Hoboken, New Jersey, 2013. ISBN 9781118646106. doi: 10.1002/9781118646106.ch5.

A. Ben-Hur and J. Weston. A user's guide to support vector machines. *Methods in molecular biology (Clifton, N.J.)*, 609:223–239, 2010. ISSN 1940-6029. doi: 10.1007/978-1-60327-241-4_13.

C. J. Burges and D. J. Crisp. Uniqueness of the svm solution. In *Proceedings of the Advances in Neural Information Processing Systems 12: Proceedings of the 1999 Conference*, volume 12, page 223, 2000.

G. J. Burghouts and J.-M. Geusebroek. Color textons for texture recognition. In *Proceedings of the BMVC*, pages 1099–1108, 2006.

Q. Cao and S. Wang. Applying over-sampling technique based on data density and cost-sensitive svm to imbalanced learning. 2:543–548, 2011. doi: 10.1109/ICIII.2011.276. URL http://ieeexplore.ieee.org/iel5/6114261/6114634/06116764.pdf?arnumber=6116764.

C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):1–27, 2011. ISSN 21576904. doi: 10.1145/1961189.1961199.

O. Chapelle. Training a support vector machine in the primal. *Neural computation*, 19(5):1155–1178, 2007. ISSN 0899-7667. doi: 10.1162/neco.2007.19.5.1155.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159, 2002. ISSN 1573-0565. doi: 10.1023/A:1012450327387. URL http://link.springer.com/content/pdf/10.1023%2FA%3A1012450327387.pdf.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002. ISSN 1076-9757.

Y.-W. Chen and C.-J. Lin. Combining svms with various feature selection strategies, 2006. URL http://link.springer.com/content/pdf/10.1007%2F978-3-540-35488-8_13.pdf.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. ISSN 0885-6125. doi: 10.1007/BF00994018.

N. Das, A. Pal, S. Mazumder, S. Sarkar, D. Gangopadhyay, and M. Nasipuri. An svm based skin disease identification using local binary patterns, 2013. URL http://ieeexplore.ieee.org/iel7/6685964/6686316/06686372.pdf?arnumber=6686372.

T. L. Diepgen. Occupational skin-disease data in europe. *International archives of occupational and environmental health*, 76(5):331–338, 2003. ISSN 0340-0131. doi: 10.1007/s00420-002-0418-1.

J. English. Current concepts of irritant contact dermatitis. *Occupational and Environmental Medicine*, 61(8):722–726, 2004. ISSN 1470-7926. doi: 10.1136/oem.2003.010710.

T. Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. ISSN 0167-8655. doi: 10.1016/j.patrec.2005.10.010.

P. Flach and M. Kull. Precision-recall-gain curves: Pr analysis done right. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 838–846, 2015. URL http://papers.nips.cc/paper/5867-precision-recall-gain-curves-pr-analysis-done-right.pdf.

P. A. Flach. The geometry of roc space: understanding machine learning metrics through roc isometrics. In *Proceedings of the ICML*, pages 194–201, 2003.

G. Forman and M. Scholz. Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. *ACM SIGKDD Explorations Newsletter*, 12(1):49–57, 2010. ISSN 1931-0145.

H. Frohlich, O. Chapelle, and B. Scholkopf. Feature selection for support vector machines by means of genetic algorithm. In *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, pages 142–148, 2003. doi: 10.1109/TAI.2003.1250182.

P. Gaspar, J. Carbonell, and J. L. Oliveira. On the parameter optimization of support vector machines for binary classification. *Journal of integrative bioinformatics*, 9(3):201, 2012. ISSN 1613-4516. doi: 10.2390/biecoll-jib-2012-201.

A. Gebejes and R. Huertas, editors. *Texture Characterization based on Grey-Level Co-occurrence Matrix*, volume 2, 2013. EDIS - Publishing Institution of the University of Zilina. ISBN 978-80-554-0648-0.

M. Hald, N. D. Berg, J. Elberling, and J. D. Johansen. Medical consultations in relation to severity of hand eczema in the general population. *The British journal of dermatology*, 158(4):773–777, 2008. ISSN 0007-0963. doi: 10.1111/j.1365-2133.2007.08431.x.

R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973. ISSN 0018-9472. doi: 10.1109/TSMC.1973.4309314. URL http://ieeexplore.ieee.org/iel5/21/4309300/04309314.pdf?arnumber=4309314.

H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009. ISSN 1041-4347. doi: 10.1109/TKDE.2008.239.

E. Held, R. Skoet, J. D. Johansen, and T. Agner. The hand eczema severity index (hecsi): a scoring system for clinical assessment of hand eczema. a study of inter- and intraobserver reliability. *The British journal of dermatology*, 152(2):302–307, 2005. ISSN 0007-0963. doi: 10.1111/j.1365-2133.2004.06305.x.

J. Hofstetter, V. Djamei, A. Navarini, M. Pouly, T. Koller, R. Arnold, and T. vor der Brück. Skin-app: Medical severity grading of hand eczema by automated image analysis: Internal project proposal, 2016.

C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification, 2003. URL https://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

C.-L. Huang and C.-J. Wang. A ga-based feature selection and parameters optimizationfor support vector machines. *Expert Systems with Applications*, 31(2):231–240, 2006. ISSN 0957-4174. doi: 10.1016/j.eswa.2005.09.024.

T. Joachims. A support vector method for multivariate performance measures, 2005. URL http://dl.acm.org/ft_gateway.cfm?id=1102399&type=pdf.

T. Joachims. Training linear svms in linear time, 2006. URL http://dl.acm.org/ft_gateway.cfm?id=1150429&type=pdf.

J. D. Johansen, M. Hald, B. L. Andersen, G. Laurberg, A. Danielsen, C. Avnstorp, B. Kristensen, O. Kristensen, K. Kaaber, J. Thormann, T. Menne, and N. Veien. Classification of hand eczema: clinical and aetiological types. based on the guideline of the danish contact dermatitis group. *Contact dermatitis*, 65(1):13–21, 2011. ISSN 0105-1873. doi: 10.1111/j.1600-0536.2011.01911.x.

M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection, 1997.

Y. Lei and L. Huan. Feature selection for high-dimensional data: A fast correlation-based filter solution, 2003.

C.-F. Lin and S.-D. Wang. Fuzzy support vector machines. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 13(2):464–471, 2002. ISSN 1045-9227. doi: 10.1109/72.991432.

X.-Y. Liu, J. Wu, and Z.-H. Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009. ISSN 1083-4419. doi: 10.1109/TSMCB.2008.2007853. URL http://ieeexplore.ieee.org/iel5/3477/4802395/04717268.pdf?arnumber=4717268.

Y. Liu, A. An, and X. Huang. Boosting prediction accuracy on imbalanced datasets with svm ensembles. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 107–118. Springer Berlin Heidelberg, 2006. doi: 10.1007/11731139_15. URL http://link.springer.com/content/pdf/10.1007%2F11731139_15.pdf.

B. Meding, K. Wrangsjo, and B. Jarvholm. Fifteen-year follow-up of hand eczema: persistence and consequences. *The British journal of dermatology*, 152(5):975–980, 2005. ISSN 0007-0963. doi: 10.1111/j.1365-2133.2005.06494.x.

S. Meenalosini, J. Janet, and E. Kannan. A novel approach in malignancy detection of computer aided diagnosis. *American Journal of Applied Sciences*, 9(7):1020–1029, 2012. ISSN 1546-9239. doi: 10.3844/ajassp.2012.1020.1029. URL http://thescipub.com/pdf/10.3844/ajassp.2012.1020.1029.

K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring, 1999. URL https://www.researchgate.net/profile/Katharina_Morik/publication/2379828_Combining_Statistical_Learning_with_a_Knowledge-Based_Approach_-_A_Case_Study_in_Intensive_Care_Monitoring/links/09e415060ea338c292000000.pdf.

D. R. Musicant, V. Kumar, and O. Ozgur. Optimizing f-measure with support vector machines. In *Proceedings of the FLAIRS Conference*, pages 356–360, 2003. URL http://www.aaai.org/Papers/FLAIRS/2003/Flairs03-069.pdf.

A. Ng. Part v - support vector machines, 2015. URL http://cs229.stanford.edu/notes/cs229-notes3.pdf.

H. Nisar, Y. K. Ch'ng, T. Y. Chew, V. V. Yap, K. H. Yeap, and J. J. Tang. A color space study for skin lesion segmentation. In *Proceedings of the 2013 IEEE International Conference on Circuits and Systems (ICCAS)*, pages 172–176, 2013. doi: 10.1109/CircuitsAndSystems.2013.6671629.

X.-X. Niu and C. Y. Suen. A novel hybrid cnn–svm classifier for recognizing handwritten digits. *Pattern Recognition*, 45(4):1318–1325, 2012. ISSN 0031-3203. doi: 10.1016/j.patcog.2011.09.021.

R. Parekh and A. Mittra. 2011: Automated detection of skin diseases using texture features, 2011. URL https://www.researchgate.net/profile/Ranjan_Parekh/publication/260563625_2011_Automated_Detection_of_Skin_Diseases_using_Texture_Features/links/004635374715bc1a61000000.pdf.

J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, 2000. URL https://www.researchgate.net/profile/John_Platt/publication/2594015_Probabilistic_Outputs_for_Support_Vector_Machines_and_Comparisons_to_Regularized_Likelihood_Methods/links/004635154cff5262d6000000.pdf.

M. Pouly, R. Arnold, A. Navarini, R. Meier, T. Koller, and S. Schnürle. Harnessing medical doctors' expert knowledge and crowd wisdom for automated classification of hand eczema: Internal project proposal, 2015.

C. A. Poynton. *Digital video and HD: Algorithms and interfaces*. Morgan Kaufmann, Waltham, MA, 2nd ed. edition, 2012. ISBN 0123919320.

H. Sain and S. W. Purnami. Combine sampling support vector machine for imbalanced data classification. *Procedia Computer Science*, 72:59–66, 2015. ISSN 18770509. doi: 10.1016/j.procs.2015.12.105.

S. J. Sangwine and R. E. N. Horne. *The Colour Image Processing Handbook*. Springer US and Imprint and Springer, Boston, MA, 1998. ISBN 1461376475.

R. E. Schapire. A brief introduction to boosting, 1999.

S. Schnürle. Skinapp: Detection and classification of hand eczema by means of supervised machine learning: Training a support vector machine on visible spectrum skin features and multiple experts' consensus diagnoses: Conducted as project vertiefungsarbeit 2 within the scope of the master's studies, 2016. URL http://ecom.d3s.ch/docs/2016_SkinApp_VM2_Schnuerle.pdf.

A. B. Sprikkelman, R. A. Tupker, H. Burgerhof, J. P. Schouten, P. L. Brand, H. S. Heymans, and W. M. van Aalderen. Severity scoring of atopic dermatitis: a comparison of three scoring systems. *Allergy*, 52(9):944–949, 1997. ISSN 0105-4538.

C. Suter. Automatische ekzemerkennung mittels textur-analyse: Conducted as project vertiefungsarbeit 1 within the scope of the master's studies, 2012.

C. Suter. Eczema detection with texture classification: Conducted as project vertiefungsarbeit 2 within the scope of the master's studies, 2013.

C. Suter, A. Navarini, M. Pouly, R. Arnold, F. S. Gutzwiller, R. Meier, and T. Koller. Detection and quantification of hand eczema by visible spectrum skin pattern analysis. In T. Schaub, G. Friedrich, and B. O'Sullivan, editors, *ECAI 2014*, volume volume 263 of *Frontiers in Artificial Intelligence and Applications*, pages 1101–1102. IOS Press, Amsterdam, 2014. ISBN 1614994196. doi: 10.3233/978-1-61499-419-0-1101.

Y. Tang. Deep learning using linear support vector machines. 2013.

I. Tomek. Two modifications of cnn. *IEEE Transactions on Systems Man and Cybernetics*, 6(11), 1976. ISSN 0018-9472. doi: 10.1109/TSMC.1976.4309452.

V. N. Vapnik. *The nature of statistical learning theory*. Statistics for engineering and information science. Springer, New York, 2nd ed. edition, 2000. ISBN 9780387987804.

M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1/2):61–81, 2005. ISSN 0920-5691. doi: 10.1023/B:VISI.0000046589.39864.ee.

V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques, 2003.

B. X. Wang and N. Japkowicz. Boosting support vector machines for imbalanced data sets. *Knowledge and Information Systems*, 25(1):1–20, 2010. ISSN 0219-1377. doi: 10.1007/s10115-009-0198-y.

J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for svms. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 668–674, 2001. URL http://papers.nips.cc/paper/1850-feature-selection-for-svms.pdf.

G. Wu and E. Y. Chang. Class-boundary alignment for imbalanced dataset learning. In *Proceedings of the ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*, pages 49–56, 2003.

X. Zhu and I. Davidson. *Knowledge discovery and data mining: Challenges and realities*. Gale virtual reference library. Information Science Reference, Hershey, 2007. ISBN 1599042541.

A. Zisserman. Lecture 3: Svm dual, kernels and regression, 2015. URL http://www.robots.ox.ac.uk/~az/lectures/ml/lect3.pdf.

Curriculum Vitae

Stefan Schnürle

Personal Details

Date of Birth	1st June 1977
Marital Status	Married
Nationality	German
E-Mail	stefan.schnuerle@schnurstracks.ch

Education

Lucerne University of Applied Sciences and Arts
in Horw, Switzerland
BSc FHZ in Computer Science with
Specialization in Software Systems, 2013

Secondary School
in Steinheim am Albuch, Germany
General Certificate of Secondary Education (Mittlere Reife), 1993

Teaching

Lucerne University of Applied Sciences and Arts
in Horw, Switzerland
Teaching Assignment: Enterprise Applications, 2015-today

Work Experience

Lucerne University of Applied Sciences and Arts
in Horw, Switzerland
Research Associate, 2015-today

Lucerne University of Applied Sciences and Arts
in Horw, Switzerland
Assistant, 2013-2015

Self-Employed as IT-Consultant
in Basel, Switzerland
Owner, 2003-2013

Languages

German	Mother Tongue
English	Professional Proficiency, Level CECR C1 (Certificate in Advanced English (CAE))
French	Basic