UPPSALA
UNIVERSITET

Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology 1371

Image Analysis and Deep Learning for Applications in Microscopy

OMER ISHAQ

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2016

ISSN 1651-6214
ISBN 978-91-554-9567-1
urn:nbn:se:uu:diva-283846

Dissertation presented at Uppsala University to be publicly examined in 2446, ITC, Lägerhyddsvägen 2, Hus 2, Uppsala, Thursday, 9 June 2016 at 10:15 for the degree of Doctor of Philosophy. The examination will be conducted in English. Faculty examiner: Associate Professor Bernd Rieger (Delft University of Technology, Department of Imaging Physics).

Abstract

Ishaq, O. 2016. Image Analysis and Deep Learning for Applications in Microscopy. *Digital Comprehensive Summaries of Uppsala Dissertations from the Faculty of Science and Technology* 1371. 76 pp. Uppsala: Acta Universitatis Upsaliensis. ISBN 978-91-554-9567-1.

Quantitative microscopy deals with the extraction of quantitative measurements from samples observed under a microscope. Recent developments in microscopy systems, sample preparation and handling techniques have enabled high throughput biological experiments resulting in large amounts of image data, at biological scales ranging from subcellular structures such as fluorescently tagged nucleic acid sequences to whole organisms such as zebrafish embryos. Consequently, methods and algorithms for automated quantitative analysis of these images have become increasingly important. These methods range from traditional image analysis techniques to use of deep learning architectures.

Many biomedical microscopy assays result in fluorescent spots. Robust detection and precise localization of these spots are two important, albeit sometimes overlapping, areas for application of quantitative image analysis. We demonstrate the use of popular deep learning architectures for spot detection and compare them against more traditional parametric model-based approaches. Moreover, we quantify the effect of pre-training and change in the size of training sets on detection performance. Thereafter, we determine the potential of training deep networks on synthetic and semi-synthetic datasets and their comparison with networks trained on manually annotated real data. In addition, we present a two-alternative forced-choice based tool for assisting in manual annotation of real image data. On a spot localization track, we parallelize a popular compressed sensing based localization method and evaluate its performance in conjunction with different optimizers, noise conditions and spot densities. We investigate its sensitivity to different point spread function estimates.

Zebrafish is an important model organism, attractive for whole-organism image-based assays for drug discovery campaigns. The effect of drug-induced neuronal damage may be expressed in the form of zebrafish shape deformation. First, we present an automated method for accurate quantification of tail deformations in multi-fish micro-plate wells using image analysis techniques such as illumination correction, segmentation, generation of branch-free skeletons of partial tail-segments and their fusion to generate complete tails. Later, we demonstrate the use of a deep learning-based pipeline for classifying micro-plate wells as either drug-affected or negative controls, resulting in competitive performance, and compare the performance from deep learning against that from traditional image analysis approaches.

Keywords: Machine learning, Deep learning, Image analysis, Quantitative microscopy, Bioimaging

Omer Ishaq, Department of Information Technology, Division of Visual Information and Interaction, Box 337, Uppsala University, SE-751 05 Uppsala, Sweden.

ISSN 1651-6214
ISBN 978-91-554-9567-1
urn:nbn:se:uu:diva-283846 (http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-283846)

My family

List of papers

This thesis is based on the following papers, which are referred to in the text by their Roman numerals.

I **Ishaq, O.**, Elf, J., Wählby, C. (2014) *An Evaluation of the Faster STORM Method for Super-resolution Microscopy*, In Proceedings of the 22nd International Conference on Pattern Recognition (ICPR 2014), IEEE, pp. 4435–4440.

II **Ishaq, O.**, Ćurić, V., Wählby, C. *Evaluation of Deep learning for Detection of Fluorescent Spots in Real Data*, Submitted for conference publication.

III **Ishaq, O.**, Ćurić, V., Wählby, C. *Training of Machine Learning Methods for Fluorescent Spot Detection*, Submitted for journal publication.

IV Clausson, C.M., Arngården, L., **Ishaq, O.**, Klaesson, A., Kühnemund, M., Grannas, K., Koos, B., Qian, X., Ranefall, P., Krzywkowski, T., Brismar, H., Nilsson, M., Wählby, C., Söderberg, O. (2015) *Compaction of Rolling Circle Amplification Products increases Signal Integrity and Signal-to-noise Ratio*, Scientific Reports, 5.

V Mignardi, M., **Ishaq, O.**, Qian, X., Wählby, C. (2016) *Bridging Histology and Bioinformatics - Computational Analysis of Spatially Resolved Transcriptomics*, in Proceedings of the IEEE, no. 99, pp. 1–12.

VI **Ishaq, O.**, Negri, J., Bray, M.A., Pacureanu, A., Peterson, R.T., Wählby, C. (2013) *Automated Quantification of Zebrafish Tail Deformation for High-throughput Drug Screening*, In Proceedings of the 10th International Symposium on Biomedical Imaging (ISBI 2013), IEEE, pp. 902–905.

VII **Ishaq, O.**[†], Sadanandan, S.K.[†], Wählby, C. *Deep Fish: Deep Learning-based Classification of Zebrafish Deformation for High-throughput Screening*, Manuscript for conference publication.

[†] These authors contributed equally to this work.

Reprints were made with permission from the publishers.

The method development and writing for the Papers I, II and III were done almost entirely by the author. For Paper IV, the author was involved in the experiments and writing of the section on measurement of signal integrity, that is, finding the overlap of the two differently colored signals, analysis of occurrence of same colored neighboring signal, simulation and its comparison with experimental results. Moreover, statistical testing and generation of the result plots was also done by the author. The author was primarily involved in the image analysis part of Paper V, that is, the section dealing with signal detection and localization. For Paper VI, the method development and writing were done almost entirely by the author. For Paper VII, the author was involved in method development and writing in equal collaboration with Sajith Kecheril Sadanandan.

Related Work

In addition to the papers included in this thesis, the author has also written or contributed to the following publications.

Wells, D.M.[†], French, A.P.[†], Naeem, A.[†], **Ishaq, O.**[†], Traini, R., Hijazi, H., Bennett, M.J., Pridmore, T.P. (2012), *Recovering the Dynamics of Root Growth and Development using Novel Image Acquisition and Analysis Methods*, Philosophical Transactions of the Royal Society of London B: Biological Sciences, vol. 367, no. 1595, pp. 1517–1524.

Changizi, N., Hamarneh, G., **Ishaq, O.**, Ward, A., Tam, R. (2010), *Extraction of the Plane of Minimal Cross-sectional Area of the Corpus Callosum using Template-driven Segmentation*, In Proceedings of the Medical Image Computing and Computer-Assisted Intervention (MICCAI 2010), pp. 17–24.

Ishaq, O., Hamarneh, G., Tarn, R., Traboulsee, A. (2007), *Longitudinal, Regional and Deformation-specific Corpus Callosum Shape Analysis for Multiple Sclerosis* In Proceedings of the 29th International Conference of the Engineering in Medicine and Biology Society (EMBS 2007), pp. 2110–2113.

Ishaq, O., Hamarneh, G., Tam, R., Traboulsee, A., Li, D. (2006), *Effects of Mid Sagittal Plane Selection on Corpus Callosal Area*, Multiple Sclerosis (special supplementary issue of ECTRMS 2006 oral presentations), vol. 12, no. 1, p. S173.

Ishaq, O., Hamarneh, G., Tam, R., Traboulsee, A. (2006), *Effects of Mid-Sagittal Plane Perturbation and Image Interpolation on Corpus Callosum Area Calculation* In Proceedings of the 6th International Symposium on Signal Processing and Information Technology (ISSPIT 2006), pp. 197–202, IEEE.

[†] These authors contributed equally to this work.

Contents

1. Introduction and overview

1.1 Quantitative microscopy

Quantitative microscopy is the process of extracting quantitative measurements from samples observed under a microscope. For biomedical applications, these measurements may represent information (extracted from images) such as intricate details of biological structure and function at minuscule proportions ranging from a few millimeters to tenths of nanometers. These images may be acquired from systems such as brightfield-, fluorescence- and electron-microscopes.

The last two decades have seen major improvements in the microscopy equipment as well as the techniques for specific labeling of biological samples. Moreover, the data acquisition process has become more automated, thereby resulting in generation of large image datasets. Furthermore, these improvements have allowed exploration of biology at ever increasing resolutions (e.g., single molecule localization experiments) and scales (e.g., high throughput drug discovery campaigns) than was possible in the past. That is to say, we are experiencing an ongoing change in both the size and type of the produced data.

1.2 Challenges for automated image analysis

These aforementioned developments pose new challenges for image analysts and developers to create novel solutions, and adapt existing ones to enable accurate, efficient, reproducible and unbiased quantitative analysis of these image datasets. More specifically, there is a need to create new solutions as well as augment and tailor existing methods to new types of data and experimental objectives. Furthermore, recently introduced methods, for instance, deep learning and compressed sensing should be adapted for problems in biomedical image analysis and benchmarked against traditional image analysis frameworks.

This thesis is an effort to address some of these aforementioned issues. To briefly mention, we develop, present and compare methods and solutions ranging from contemporary image analysis pipelines based on staple techniques such as segmentation, branch-free skeletonization etc., to data-driven approaches based on learning deep representations from augmented datasets.

These solutions are applied to challenging practical problems posed by biological samples ranging in scale from whole-body, yet microscopic, organisms to specifically labeled subcellular structures.

Since many of these image analysis solutions are designed for applications pertaining to specific biological models, an introductory overview of these biological models and the methods used in this thesis is provided in Section 1.3 and Section 1.4, respectively.

1.3 Biological models

The study of biological structure and function can take place at different scales, and within different organisms, depending on the target application. For example, exploration of subcelluar function may be suited for understanding biological pathways, whereas gross phenotypic changes may be better interpreted at an organ, or even an organism scale. Furthermore, when targeting human physiology it may be advantageous and risk-averse to first perform experiments in non-human species which can serve as suitable models for human physiology.

1.3.1 Fluorescently labeled subcellular structures

Cells contain minuscule subcellular structures/entities such as deoxyribonucleic acid (DNA), ribonucleic acid (RNA) and proteins etc. These entities can be specifically labeled with fluorescent probes and biomarkers. These probes can be observed under fluorescence microscopes as bright spots over a dark background. A set of identified probes/spots is shown in Figure 1.1. Precise localization of these spots is an important application in biomedical image analysis since it enables detailed mapping of the subcellular structure as well as the study of clustering patterns of these entities. Moreover, the estimated locations serve as starting points for tracking these entities over time, which is important for determining their biological function. However, one needs to reliably identify or detect these spots before they can be localized.

Together, spot detection and localization constitute important steps of a spot processing pipeline (for overview of spot detection and localization please refer to [1, 2]). These detection and localization operations are confounded by factors such as sample and readout noise, photobleaching, short exposure times and high density of fluorophores in the acquired image, thus posing significant image analysis challenges.

From an application viewpoint, the first five papers in this thesis, that is, Papers I – V are focused on presenting and extending methods dealing with different aspects of spot processing. More precisely, Paper I focuses on spot localization through a compressed sensing approach, where as Papers II and III present methods for spot detection based on deep learning. Furthermore,

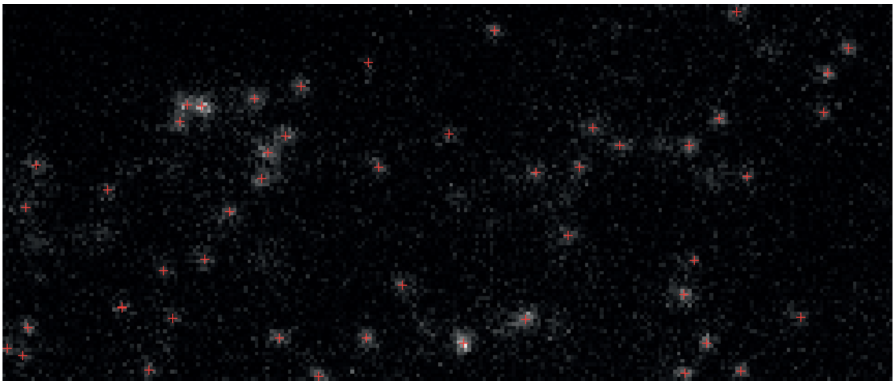


Figure 1.1. A set of biomarkers identified (marked in red color) on a fluorescence microscopy image.

in Paper IV we perform spot detection using thresholding methods and also undertake a form of cluster analysis. Finally, Paper V provides a review of techniques for study of transcriptional variations, including their spot detection and localization components.

We realize that in a typical spot processing framework detection is followed by spot localization. However, we initially started working with localization, which in turn emphasized for us the need for having better spot detection methods as precursors. Therefore, we have followed the same chronological order in this thesis and placed the localization paper before the detection ones.

1.3.2 Zebrafish embryos

Zebrafish is a freshwater fish that has been widely used as a vertebrate model organism [3, 4]. Zebrafish undergo rapid embryonic development, this feature combined with the ability to observe their internal organs due to the transparency of the embryos contributes to the suitability of zebrafish as a model organism.

Zebrafish can undergo shape deformation during embryonic development upon exposure to certain chemical compounds, which act as inhibitors of DNA repair. This deformation may be expressed as bending of its tail or as a change in the shape of the yolk-ball etc. Examples of affected (i.e., deformed shape) and non-affected/regular (i.e., regular shape) zebrafish embryos are shown in Figure 1.2.

Papers VI and VII of this thesis are focused on image-based quantification and classification of the shape deformation of zebrafish embryos. The image datasets for these papers were acquired through a brightfield microscope. Primary contributions are summarized below, for details refer to the full length papers. In Paper VI, we employ the curvature of the tails of the zebrafish embryos as a user-specified feature for measuring and classifying the shape

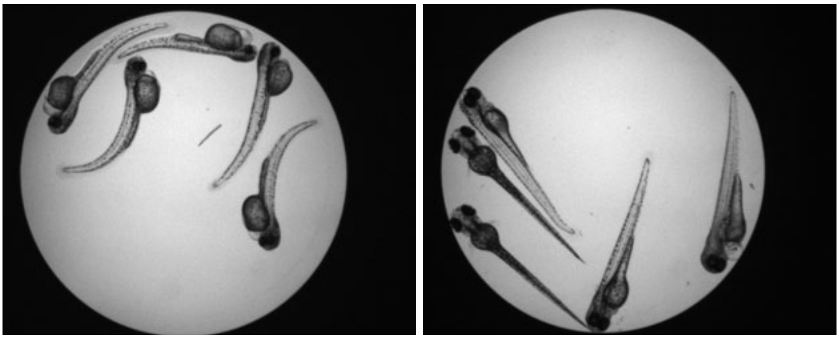


Figure 1.2. Example of deformed (left) and regular zebrafish (right).

deformation. We employ an image analysis pipeline based on smoothing, illumination correction, segmentation, branch-free skeletonization and tail-fusion operations. In Paper VII, we address the classification problem from a deep learning approach.

1.4 Methods

We have presented a wide variety of techniques and methods in this thesis. As alluded to earlier, these methods have ranged from the traditional and staple image analysis methods such as illumination correction, segmentation, branch-free skeletonization, threshold based binarization to both traditional and more recently introduced deep machine learning techniques. Moreover, methods such as compressed sensing have also been employed.

Our choice of any particular type of method has transcended the underlying biological model. For instance, deep learning is a recently introduced data-driven machine learning technique which builds hierarchical representation of data for addressing the classification problem. Deep methods are also termed as part of representation learning, since deep methods automatically learn the discriminative features of the data as part of classifier training. This is in contrast with traditional machine learning where hand-crafted features are provided by the users. Deep methods have proven useful for many image analysis applications. In our experiments, we have performed deep learning for both the spot detection task in Papers II and III, as well as for the identification of the deformed zebrafish in Paper VII. Of course, this thematic categorization of these papers with a particular method is a crude one, since in each of these papers the application of deep learning has been accompanied by other contributions which are unique to that particular paper. For instance, Paper III is also focused on training these networks from different types of synthetic and semi-synthetic data.

Compressed sensing is a signal processing technique which enables high quality signal reconstruction by solving under-determined linear systems. The quality of the reconstruction is subject to constraints on the sparsity of the signal as well as assumptions about the signal acquisition process. We apply compressed sensing for the spot localization task in Paper I, especially for images with high fluorophore densities.

Traditional techniques for detection of fluorescent spots such as thresholding on image intensity are applied in Paper IV. Moreover, we present a distance based cluster analysis method for measurement of signal integrity. In Paper VI, we present an image analysis solution, for the zebrafish tail curvature quantification and classification problem, comprising a wide variety of image analysis methods such as noise reduction, illumination correction, foreground segmentation, branch-free skeletonization, computation of curvature-based features as well as traditional machine learning based classification. Table 1.1 provides an overview of the organization of the papers around the biological models as well as the image analysis methods.

Paper number	Biological model	Primary method
Paper I	Fluorescent spots	Compressed sensing
Paper II	Fluorescent spots	Deep learning
Paper III	Fluorescent spots	Deep learning
Paper IV	Fluorescent spots	Traditional spot detection
Paper V	Fluorescent spots	Review
Paper VI	Zebrafish embryos	Image analysis pipeline
Paper VII	Zebrafish embryos	Deep learning

Table 1.1. *An overview of the organization of the papers around the biological models as well as the image analysis methods.*

1.5 Organization of thesis

Image analysis-based spot detection and localization, deep learning and compressed sensing constitute important techniques employed in the papers which form this thesis. These three techniques are discussed in Chapters 2, 3 and 4, respectively.

Chapter 5 briefly summarizes the methods developed during the course of the Ph.D. studies as well as the contributions for each paper included in the thesis. Each section represents one of the included papers. Conclusions and future perspectives are discussed in Chapter 6.

2. Spot detection and localization

2.1 Introduction

Many fundamental biological functions occur at the subcellular scale. Some examples include the replication of the DNA, transcription of DNA, formation of proteins and actions associated with different biological pathways etc. Therefore, understanding and manipulation of the structure and function of the minuscule entities within a cell has become an increasingly important part of biological research.

Fluorescence microscopy has emerged as a key technology enabling the visual observation of these entities, driven in part by the discovery and development of fluorescent biomarkers such as organic dyes and fluorescent proteins etc [5]. These biomarkers can be used to bind to specific entities such as proteins, nucleic acid sequences, receptors etc. Under fluorescence microscopes, these biomarkers can typically be viewed as bright spots over a dark background (depending on the imaging technique) as shown in Figure 2.1. The specific labeling ensures that ideally, only the objects or sequences of interest appear as the foreground and the other types of entities present in the sample appear as background. Recent advancements in fluorescence microscopy in the form of single molecule localization techniques have made it possible to map out complex biological structures and track single particle at ever increasing resolutions (i.e.,$\sim$ 10 nm) [2]. Moreover, fluorescence microscopy allows live cell imaging which is not possible under some higher resolution modalities, for instance, electron microscopy.

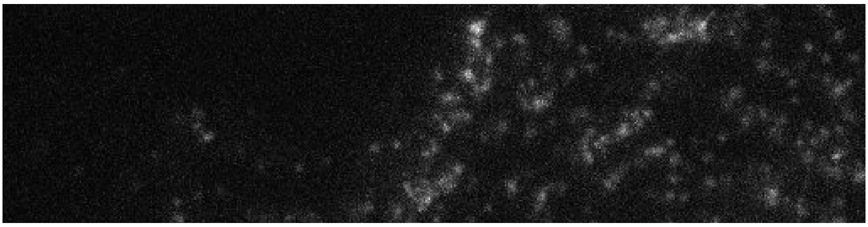


Figure 2.1. Bright fluorescent spots over a dark background.

2.2 Image-based spot processing and analysis

In addition to the advancements in imaging systems, biology and chemistry, a large body of computational methods has emerged for processing and analysis of these fluorescent spots. These computational methods span a wide range of applications. Typically the first step in processing and analysis of fluorescent spots is to identify whether a particular location in the image contains a spot or not [1, 6]. This step has been alternatively referred to as spot detection, spot segmentation or region of interest (ROI) detection. Typically, spot detection is a binary classification problem with spot/foreground and background being the two possible labels. For some applications, for instance quantification of gene expression, the spot detection step may be enough to fulfill the requirements of the analysis. For other applications, for instance mapping the structure of microtubules, it may be necessary to have a subsequent second step to precisely determine the locations of the detected fluorophores through localization algorithms. The localization methods range from center of mass based calculations to more complex model-fitting techniques [2, 5, 7]. Nowadays the term 'spot localization' has increasingly come to imply a single molecule localization method. This part of the thesis focuses on detection and localization of fluorescent spots and an overview of the two areas is provided in Sections 2.5 and 2.6, respectively. The Papers II and III in this thesis focus on deep learning based spot detection, whereas Paper I is focused on a compressed sensing-based approach for spot localization. Sometimes these spot processing and analysis methods may be used to validate the effectiveness of a biomarker. For instance, in Paper IV we use spot analysis methods for measuring the intensity and integrity of fluorescent spots after introduction of an oligonucleotide.

Although not a part of this thesis, however we should mention that the precise locations of biomarkers, obtained from the localization algorithms, can also be used in subsequent tracking steps which trace the trajectory of a fluorophore over time. An additional application of the location information is called cluster analysis and focuses on quantitative comparison of the clustering pattern of the biomarkers against a randomly distributed sample [2].

2.3 Image acquisition and appearance

In a standard fluorescence microscopy system, the fluorescent biomarkers are exposed to light with wavelength which falls within the excitation spectrum of the fluorophore. The electrons in the fluorophore absorb light and transition to an excited (i.e., high energy) state. Subsequently these electrons return to the ground (i.e., low energy) state and this transition results in the emission of light from the fluorophore, with a longer wavelength (i.e., lower frequency and energy). The emitted light has a different (i.e., longer) wavelength than the excitation beam and consequently the two beams can be separated through optical filters. Finally only the emitted fluorescence is displayed to the human observer or the imaging device such as a camera. In the absence of autofluorescence and noise, the fluorophores appear as bright objects on a dark background as shown in Figure 2.1.

2.4 Resolving objects in diffraction limited images

The exact appearance of each fluorescence biomarker is dependent on a number of factors including the type of microscopy system. In typical diffraction limited systems such as the standard fluorescence microscopes, the resolution d (i.e., the minimum spatial distance at which two point sources can be distinguished) is limited by the wavelength λ of the illumination light beam as well as the numerical aperture NA of the imaging system. One formulation of this measure, called Abbe diffraction limit is given as,

$$d = \frac{\lambda}{2 \cdot NA}. \tag{2.1}$$

Due to the physical limitations on both the wavelength of the illumination light beam (w.r.t., its suitability for illumination of biological samples) as well as the numerical aperture, the resolution d is bounded at $\sim$200–250 nm. In these diffraction limited systems the fluorophores, ideally point sources of light, appear as spread out bright spots which lack a well defined boundary (i.e., a boundary specified by sharp edges) as shown in Figure 2.2. The image of a point source/fluorophore under a microscope is similar to the PSF of that microscope. These spots can be modeled by the Gaussian function [8].

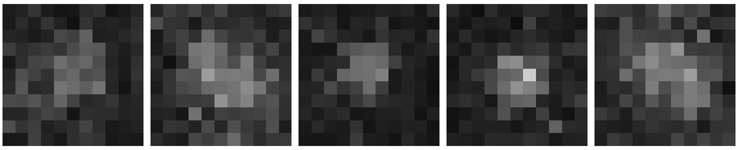


Figure 2.2. Example of diffraction limited spots. Observe that the spots lack a well defined boundary.

We emphasize that the aforementioned resolution limit only specifies the minimum distance at which two adjacent spots can be individually resolved. However, if we know that there exists only one isolated spot then the position of the spot can be localized to fairly high precision through computational methods such as fitting a Gaussian model. In other words, in diffraction limited systems, the density of the spots in a field of view affects the accuracy with which the spots can ultimately be resolved. This simple observation, combined with the development of photoswitchable fluorophores which can be turned on or off multiple times in an acquisition cycle, forms the basis of the increasingly used single molecule localization methods such as STORM and PALM [9, 10].

In STORM [9], the fluorophore density is reduced by exciting or activating only a small subset of the fluorophores at a time and splitting the imaging of a sample over a number of frames. The number of activated fluorophores is small enough that there is a low probability of any two fluorophores overlapping in a given frame. These individual fluorophores, though diffraction limited, can still be localized to high precision by techniques such as Gaussian fitting. Iterative activation of different subsets of fluorophores ensures that most of these fluorophores can be individually localized in different frames and the localizations across frames can then be combined to provide a single high resolution image.

In other words, the frames generated by a single molecule localization microscopy setup such as STORM/PALM are diffraction limited and not qualitatively different from the ones acquired under standard fluorescent microscopy systems. It is the low fluorophore density and the computational methods which enable extraction of high resolution from these images. A brief discussion of computational methods for single molecule localization is provided in section 2.6.

2.5 Spot detection

Typically, the detection of spots entails three steps: (i) In the first step the background noise is reduced, commonly through a filtering operation; (ii) The next step is a foreground enhancement step where the foreground spots are amplified; (iii) The last step is a generalized thresholding one where selection of an appropriate threshold helps to distinguish the spots from the background [1].

In spot detection applications, the noise suppression and spot enhancement steps (i.e., steps (i) and (ii) as described above) may be integrated into a single operation. For example, convolution of an image with a Gaussian function of standard deviation equal to the PSF of the imaging system may result in both suppression of the background noise as well as enhancement of the spots. A large number of spot detection methods exist in literature, examples include [11, 12, 13, 14, 15, 16, 17]. For review and quantitative evaluation of some of these techniques the reader is referred to [1, 6].

2.5.1 Categories of spot detection methods

The quantitative survey by Smal et al., [1], categorizes the spot detection methods into ‘supervised’ and ‘unsupervised’ methods [1]. The supervised methods refer to data-driven machine learning-based techniques which entail availability of ground truth data and labels for training. In such methods, either user specified feature detectors or detectors *learnt* as part of classifier training (i.e., a representation learning approach) are used to compute features for each training example (we use and evaluate both types of the methods in Papers II and III). Subsequently, these sets of features are employed to *learn* a decision boundary (i.e., a threshold) for separating the spots and the background in an automated manner. Examples include the use of adaptive boosting technique [11] as well as the deep convolution neural networks (DCNNs) which we have employed in Papers II and III. As claimed in [1], supervised methods appear to have better detection performance for images with low signal to noise ration (SNR).

The unsupervised methods refer to non-data-driven techniques where *user specified* values are used for both the parametric template or model used for noise-suppressions/spot-enhancement as well as for the threshold for distinguishing between spots and background. Typically, such methods neither require availability of labeled training data nor a *learning* framework for the threshold selection. An example of such a method is provided in [18].

We have followed a similar categorization scheme (i.e., supervised vs. unsupervised). However, we have primarily focused on use of machine-learning methods and particularly deep learning methods, and to avoid confusion with the terms supervised and unsupervised as used in the context of machine learning we have referred to the two above mentioned categories of detection methods as ‘machine learning-based’ and ‘model-based’ methods in Papers II, III and V.

2.5.2 Model-based spot detection methods

An example of such a method is the wavelet multiscale product [18]. Initially the method computes k different wavelet planes. These planes represents the decomposition of image features at different scales. Then noise reduction is performed on each plane by thresholding the coefficients with a threshold which is a fixed multiple of the standard deviation of the coefficients at that particular scale. Subsequently, the remaining non-zero coefficients at different scales are multiplied together under the assumption that foreground spots are more stable across different scales and more likely to have survived the previous thresholding step at different wavelet planes. Therefore the product of their coefficients at different scales is more likely to be significantly larger than the product of coefficients representing the image background. The resulting product map is thresholded with a fixed threshold to yield the identified spots. We use an implementation of this method for comparison with deep learning-based spot detectors in the Paper II. The method was selected because it is a commonly used method for such applications and has previously been employed in quantitative comparisons of spot detection methods [1, 6]. Additional examples of such methods and applications include [12, 13, 14, 15, 16, 17].

In Paper IV, we evaluate the effectiveness of a compaction oligonucleotide by measuring its SNR, spatial colocalization and its propensity to disintegrate into clusters. The detection of spots is carried out through manual and automated thresholding schemes.

2.5.3 Data-driven spot detection

In spot detection literature, data-driven or machine learning-based spot detectors are less prevalent than their model-based counterparts due to the need for labeled training data. However, the suitability of machine learning-based detectors for low SNR images was emphasized by the quantitative comparison undertaken in [1].

One of the oft-quoted example of a traditional data-driven method is the boosted framework proposed by Jiang et al., [11]. In our Papers II and III, this method was used as one of the detectors for quantitative comparison against deep learning and model-based techniques and provided impressive spot detection performance. Typical boosting frameworks are designed around the idea that a number of weak classifiers can be combined together in a weighted sum to yield a strong classification ensemble. For this particular implementation, a number of Haar-like features (shown in Figure 2.3), intuitively similar to the Haar wavelets, are used as weak classifiers which are combined together through an Adaboost framework, the framework determines the relative weights for each weak classifier during the training process. Such an approach necessitates the availability of labeled data for training, model validation and testing, a set of suitable preselected features for transforming the input data into a feature space and finally a classification framework which can combine the data and features to yield useful predictions.

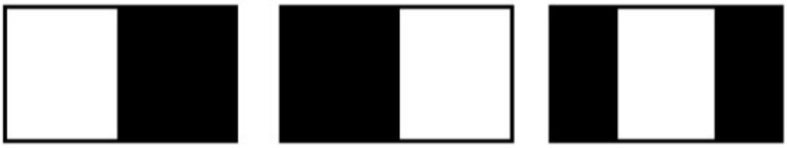


Figure 2.3. Example of a few Haar-like features.

The use of deep learning methods has been a key advancement in machine learning and pattern recognition in the past few years. Deep architectures have been applied to a number of computer vision and image analysis applications where these methods have recorded state of the art performance [34, 32]. Deep learning differs from traditional machine learning by the fact that the features are learnt as part of the classifier training and not predetermined. Such a framework where feature learning and classifier training takes place simultaneously is known as a representation learning framework. Details of how to build and train these deep architectures are discussed in Chapter 3. We use these deep architectures for the spot detection task in Paper II and Paper III. To the best of our knowledge, ours is the first work to apply deep learning for spot detection. We tested a number of popular deep architectures for spot detection and recorded performance competitive with the state of the art detectors.

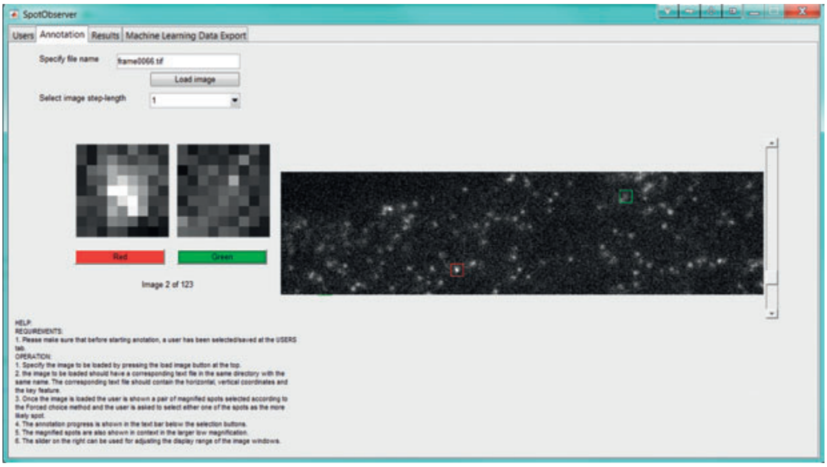
Typical deep learning pipelines entail the following steps: (i) Collection and labeling of image data, in our case the data comprises small image patches or regions of interest (ROI) centered around local maxima in the image; (ii) Augmentation of data through transformations such as flips and rotations; (iii) Division of data into training, model validation and test datasets; (iv) Training and testing on architectures.

Additional analysis of these deep architectures for spot detection focused on the effect of pretraining these detectors on external datasets, identification of minimum number of training examples for competitive detection performance. Use of synthetic and semi-synthetic image sets for training deep architectures and the comparison of their detection performance against architectures trained on real data. For details of the experiments and analysis of the results please refer to the Papers II and III.

2.5.4 Tool for manual annotation of fluorescent spots

Machine learning methods (more specifically, supervised machine learning methods) require labeled data for training, validation and testing. In Papers II and III, the testing was performed on extensively annotated real images containing fluorescent spots. The data labeling process is dependent on the availability of domain experts to identify the ground truth spots. These positions may be identified through a manual point-and-click approach; where an image is displayed to a rater who uses a mouse, to mark the visually distinct spots. Such an approach suffers from a few drawbacks: (i) it is biased towards annotating brighter spots, therefore the annotated spots are potentially fewer than the total number of spots; (ii) the absence of meaningful borderline- and negative-examples (i.e., dull spots and background, respectively). In Paper III, we address these issues through development of a tool (i.e., *SpotObserver*) for facilitating manual annotation of positive as well as negative examples of spots. The tool is driven by the two-alternative forced-choice (2AFC) approach. Such an approach has previously been found to be quite useful for annotation of biomedical data [19]. *SpotObserver* ensures that all types of candidate spots, including background noise and artifacts, are evaluated by the user resulting in both positive and negative examples. A quantitative analysis of *SpotObserver* against a point-and-click approach is available in Paper III.

The tool was used for assisting in the manual annotation of real data in Paper II and Paper III, resulting in two manually annotated datasets (i.e., one for each paper). A screen shot of the tool is shown in Figure 2.4.



Specify file name: frame0066.tif

Select image step-length: 1

Image 2 of 123

HELP:
REQUIREMENTS:
1. Please make sure that before starting anotation, a user has been selected/saved at the USERS tab.
OPERATION:
1. Specify the image to be loaded by pressing the load image button at the top.
2. the image to be loaded should have a corresponding text file in the same directory with the same name. The corresponding text file should contain the horizontal, vertical coordinates and the key feature.
3. Once the image is loaded the user is shown a pair of magnified spots selected according to the Forced choice method and the user is asked to select either one of the spots as the more likely spot.
4. The annotation progress is shown in the text bar below the selection buttons.
5. The magnified spots are also shown in context in the larger low magnification.
6. The slider on the right can be used for adjusting the display range of the image windows.

Figure 2.4. Screenshot of the *SpotObserver* tool used for assisting in the manual annotation of the real data. The tool is based on a 2AFC approach where regions of interest (ROIs) are shown to the annotator in a side by side configuration. The annotator can decide which of these ROIs appears more spot like, resulting in an ordering of the ROIs in terms of their 'spot-like' appearance.

2.6 Single molecule localization

We have carried out spot localization in Paper I, with a non-traditional compressed sensing based technique. However for the sake of completeness, here we provide a brief overview of single molecule localization and refer to Paper I as well.

As explained earlier, spot localization methods typically rely on a preceding spot detection step to identify an ROI or a spot. Once a spot has been positively identified, then its location can be precisely localized with a spot localization method, provided certain assumptions about the density of the fluorophore, image quality and the type of localization method are satisfied. We reiterate, that most widely used methods for spot localization are based on fitting a Gaussian model [2, 5]. Alternate methods based on radial symmetry [20] and compressed sensing [21] have also been proposed.

Since there is no qualitative difference between images from standard fluorescence microscopes or a STORM/PALM setup (they are actually almost the same except some potential differences between the light sources and the type of fluorophores used), in principal the localization techniques used for identifying a single spot are the same in both the cases.

Most of the widely used localization methods are based on a model fitting approach. These localization algorithms fit a PSF estimate (i.e., a Gaussian approximating an Airy pattern) to each spot. The localization of a spot can be treated as an optimization process. For 2D data, the optimization parameters may include the coordinates of the Gaussian mean, standard deviation of Gaussian, estimate of the background etc. In the case of a non-symmetric PSF, the Gaussian would include two different estimates of the standard deviations as well as a parameter for orienting the Gaussian in 2D. The fitting is typically carried out using a maximum likelihood estimation (MLE) method, or through a least square (LS) fitting approach [22].

In single molecule localization microscopy for fixed spots, an image may be composed of hundreds of image frames. Moreover, a particular fluorophore may remain active across a number of these frames. Therefore, a localization method provides an estimate of the fluorophore position for each of these frames. These location estimates may differ from each other slightly due to the stochastic nature of the emission of photons from the fluorophores. The mean of these frame-estimates, for a given fluorophore, can be used as a final computed fluorophore position. The standard deviation of these frame-estimates, signifying their spread, is called the precision of the localization method. The distance between the mean estimate and the true position of the fluorophore, if available, is called the accuracy of the localization method [22].

A lower bound to the localization precision of an unbiased estimator is provided by the Cramer-Rao Lower Bound (CRLB). A good approximation for the CRLB is provided in [23].

In single molecule localization microscopy methods, a large number of frames are required to completely image a given sample. The imaging can take significant time and result in photobleaching. Recently methods have been proposed which minimize the number of acquired frames by localizing spots in images with considerably higher fluorophore density including overlapping spots. Examples of such methods include the multi-emitter fitting method [24]. Another such method is the compressed sensing-based *Faster STORM* algorithm which treats the localization problem as an under-determined linear systems and seeks to recover a sparse solution to the problem [21]. This method is specifically targeted for processing and analysis of very high fluorophore density images. In Paper I, we quantitatively analyze and augment this compressed sensing-based method. The original method was found to be computationally expensive and therefore time consuming. We improved the speed by multi-core parallelization. *Faster STORM* requires an estimate of the PSF. We quantified the change in localization performance against deviations from a suitable PSF estimate and found that performance degrades significantly when the PSF estimate is not a suitable one.

3. Deep Neural Networks

3.1 Background

The use of neural models of computing extends as far back as mid 1900s including early contributions such as the Hebb and the Perceptron models in the 40s and 60s, respectively [25, 26]. An important milestone was the use of backpropagation for learning family tree structures by Rumelhart et al. [27]. Presently, backpropagation remains the dominant paradigm for training different neural network architectures. Later, LeCun et al. used backpropagation for training increasingly complex neural networks (which also included convolution layers), applied to practically significant problems such as the recognition of hand-written digits [28, 29]. Note that the aforementioned uses of backpropagation and convolution layers were significant, but not seminal ones. For details of the evolution of backpropagation-based optimization, use of convolution layers and deep models please refer to the excellent review by Schmidhuber [30].

After years of decreased research focus, in part due to the lack of computational resources for training deeper (i.e., with more layers) and more complex networks as well as the non-availability of large annotated image sets, interest in these networks was revived when Hinton et al. were able to iteratively train sequential layers in a sufficiently deep auto-encoder for the task of dimensionality reduction [31]. In the next few years, improvements in computational resources such as the use of graphical processing units (GPUs) combined with better understanding of ways and means of initializing and training these networks resulted in ground breaking results by a DCNN, called 'AlexNet', on the public 'ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)' image set and brought deep learning to the mainstream [32]. ILSVRC is an annually held challenge focusing on detection and classification tasks [33]. For examples of the application of deep neural networks please refer to Section 3.8 as well as refer to the excellent reviews by Schmidhuber and LeCun et al., [30, 34]. We make use of AlexNet in our thesis in the Papers II and VII.

We should emphasize that the field of deep learning is not limited to deep neural networks and alternate non-neural deep architectures such as k-means clustering-based hierarchically organized structures [35] have also been suggested. However, the use of deep neural networks is the prevalent archetypal deep learning model and in this thesis we will use the term 'deep learning' to refer to the 'deep neural networks'.

3.2 Structure of a deep network

A typical neural network is composed of an input, an output and at least one intermediate layer, referred to as the hidden layer. The layers are composed of elements called neurons. The neurons of a layer are connected to neurons of the preceding and succeeding layers through weighted connections, also called network weights. Each neuron y_j computes a weighted sum of its inputs x_i (using weights w_{ij}) and a bias b_j and depending on the network architecture may apply a linear or non-linear transformation/activation f to this sum, that is,

$$y_j = f(\sum_i x_i w_{ij} + b_j). \tag{3.1}$$

The final performance of a network is measured through a 'loss' function. During network training, this loss function is minimized with respect to the set of weighted connections W (consisting of the aforementioned weights w_{ij}). For simplicity we will only consider the weights and ignore the bias in the notations. The training is typically based on a backpropagation regime and involves two iterative steps called the forward and the backward pass. Removal of the non-linear activations reduces the network to simple matrix multiplication operations.

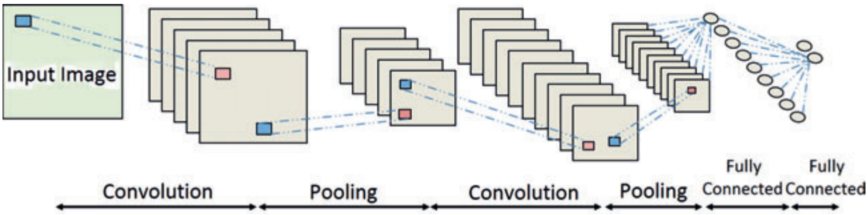


Figure 3.1. Representation of a DCNN comprising two convolution layer which compute multiple feature maps. The convolution layers are interleaved with two subsampling or pooling layers which perform a combination of dimensionality reduction via response pooling and impart a degree of translation invariance. Finally, two fully connected layers are included at the end which perform a weighted sum of the feature vectors obtained from the convolution layers.

3.2.1 Network depth

The network depth is typically counted as its number of layers excluding the input layer. Therefore, the standard shallow neural network with one input, one output and one hidden layer has a depth two (i.e., considered two layers deep). The width of a particular layer is proportional to the number of its neurons. Typically, a deeper and wider network may translate to a greater modeling capacity of the network (this observation is subjective and dependent on the *type* of the neural network and the problem to which it is applied, while keeping in mind that a deeper network can be subject to overfitting). Although the question of what constitutes a 'shallow' or 'deep' network is quite subjective, it is common to consider a network with multiple hidden layers as a deep one.

A generalized representation of a deep network (in particular a DCNN) is shown in Figure 3.1, it comprises a stack of two convolution layers (interleaved with pooling layers) followed by the fully-connected (or inner-product) layers. The convolution layers result in generation of multiple feature maps from the previous layer or the input image. The subsampling layers are used to reduce the dimensionality of the network by pooling the spatially localized features. The features generated by the last convolution or pooling layer in the network are fed to a fully connected layer which performs a weighted sum of these inputs and typically applies a non-linear transformation (i.e., activation) on the input.

3.2.2 Network Orientation

When describing a deep network, the terms 'up' and 'down' typically refer to directions oriented towards the output and the input layers, respectively. In other words, the input layer is considered present at the bottom and the output layer is considered to exist at the top of the network. Therefore, during network training using backpropagation, the data in the network flows upstream in a forward pass and the gradient updates flow downstream in a backward pass.

3.2.3 Layers, activations and variations in taxonomy

In theory, the activation functions are considered as part of the network layer and not as separate layers themselves. However, due to the explosion in the number of deep learning libraries and implementations in the last two to three years, it is common to find instances where, for the purpose of implementation, the activation functions are implemented as separate layers [36], although in practice they perform the same operations as before. Therefore, it is becoming more common to find discussions where the activation functions are referred to as 'layers'. Description of the more commonly used network components such as layers, activations and loss functions etc., used for building modern deep learning pipelines is provided in Section 3.3.

3.3 Components of a DCNN

3.3.1 Convolution layers

A convolution layer is one of most common types of network layers used for computer vision and image analysis applications (Please refer to Section 3.8 for a brief discussion of these applications). A convolution layer comprises a number of small two dimensional filters/feature detectors. Unlike the hand-crafted features used in typical machine learning paradigms, these filters used in the deep networks are learnt as part of the network training. Such a paradigm where the feature training is done along side the classifier training is also known as representation learning.

These learnt filters are convolved with the input image and the resulting feature responses are passed upstream to the next processing layer [28]. Neural networks employing cascading stacks of these convolution layers (typically interleaved with pooling layers) are referred to as DCNNs. A well known example is the AlexNet which presented the best recognition performance at ILSVRC 2012. The AlexNet employs five of these convolution layers [32]. Taking this concept further, the more recent trend is to build networks almost completely using convolution layers; the recently proposed 152 layers deep residual learning network, also referred to as ResNet, (best performance at ILSVRC 2015) is composed almost entirely of convolution layers [37].

Convolution layers as hierarchy of features

In classic machine learning-based image processing pipelines, a set of hand-crafted feature detectors generate responses which are subsequently processed through a classifier for assignment of class labels. Similarly, the convolution layers present early in a deep network (i.e., downstream) can be considered as the corresponding low level feature detectors. However, these feature detectors (more specifically the weights of the kernels used in these detectors) are learnt as part of the data-driven classifier training. Typically, DCNNs use a hierarchy of multiple successive convolution layers where the output from a convolution layer C_n is fed into the next convolution layer C_{n+1}. This results in both low level (downstream and elementary) and high level (upstream and composite) features. A high level features can be considered as a weighted combination of the underlying low level features. In this manner the low level features (typically representing edges and local textures) are combined together to generate high level features which may represent complex and more sophisticated patterns in the image. Recently, a number of research papers have focused on visualization of both low and high level features generated from neural networks [38, 39, 40] for a better understanding of these features. We also perform feature visualization in Papers II and VII.

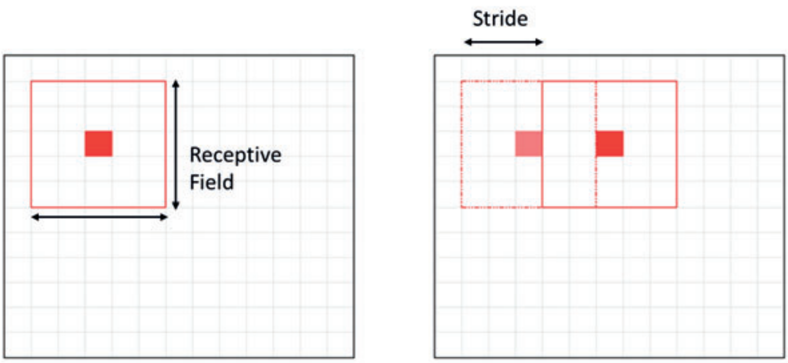


Figure 3.2. Representation of the receptive field and stride of a convolution layer. The receptive field spans a 5×5 neighborhood. For this example the stride is set as three and implies that the convolution filter is shifted by a distance three after each convolution.

Receptive fields and weight sharing

These convolution filters are relatively small in size (typical range between 3×3 to 11×11 pixels) and exploit the fact that image pixels within a small local neighborhood enjoy strong spatial correlations. The size of a convolution filter is called its 'receptive field' and is a user specified parameter. The size of the receptive field controls the number of neurons in the preceding layer (or pixels if the preceding layer is an input one) which are connected to each neuron in the current layer. Convolution layers employ 'shared weights', that is, the weights of the convolution filter are independent of where it is applied in an image implying that the weight updates to a convolution filter have to be consistent across the full image. Shared weights result in a reduction in the dimensionality of the network weights (i.e., optimization parameters). Therefore, convolution networks tend to have fewer weights than fully connected networks.

'Stride' is the other key parameter in a convolution layer. It specifies the horizontal or vertical shift in the filter position after performing a convolution. As an example, please refer to the Figure 3.2 where a 5×5 kernel is being convolved with an image, a stride of three pixels implies that after the convolution the kernel will be shifted three pixels to the right.

3.3.2 Fully connected layers

Dense connectivity pattern

As implied by the name, a fully connected layer (also called an inner product layer) has the property that each of its neurons is connected with all the elements of its preceding (downstream) layer, thus resulting in a very dense connecting pattern. This is in contrast with the convolution layers where an element is connected only with those preceding elements which lie in its receptive field. The high density connectivity pattern means that only a few fully connected layers can still account for a large subset of the network weights.

Position in a deep network
Typically, the fully connected layers comprise the final few layers of a network just before calculation of the network loss. It is also common for two adjacent fully connected layers to incorporate an intermediate non-linearity such as the Rectified linear unit (ReLU) activation function [41] (ReLU is discussed in more detail in Section 3.3.5). Lately, the utility of these layers have been called in question and it is becoming more common to replace the final fully connected layer with an support vector machine classifier. In [42], it was found that the use of fewer fully connected layers resulted in better generalization on the test set.

3.3.3 Pooling layers

Another commonly used layer is the pooling layer. It is typically positioned after a convolution layer and is specified by ‘size’, ‘stride’ and ‘type’ parameters. A pooling layer is used for subsampling the feature response from a preceding convolution layer and for propagating only the dominant feature response (depending on the type of pooling layer) to the succeeding layers. Pooling layers also introduce a degree of translation invariance in neural networks.

Size and stride of a pooling layer
The size parameter specifies the length of the neighborhood in both horizontal and vertical directions over which the pooling operation is performed. The stride parameter is similar to the one used in convolution layers and specifies the position of the next pooling neighborhood. Tweaking the stride and size parameters can result in both overlapping (pooling stride $<$ pooling size) and non-overlapping (pooling stride $\geq$ pooling size) pooling neighborhoods.

Types of pooling layers
The most common pooling types are the maximum (also called max pooling) and the average/mean pooling. For maximum pooling, only the dominant feature response in the pooling window is retained and propagated to the next layer. On the other hand, the average pooling layer computes and propagates the mean response of all the features in the pooling window. As an example, refer to the Figure 3.3 where the outputs from a max and a mean pooling layers are shown. The subsampling operation also results in a reduction in the number of network parameters and therefore makes the optimization more tractable.

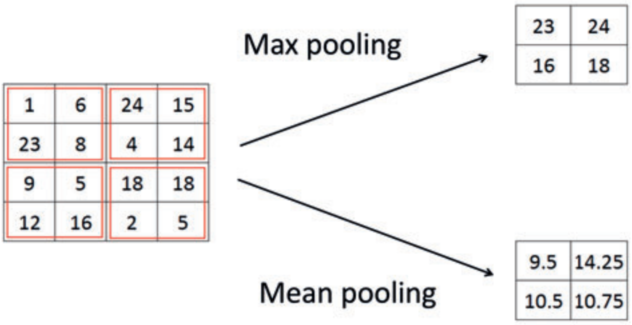


Figure 3.3. Example of max and mean pooling types. The example uses a pooling size of two pixels along the horizontal and the vertical. The stride is of size two pixels resulting in non-overlapping pooling neighborhoods.

3.3.4 Dropout layers

Dropout layers were introduced by Hinton et al., in 2012 [43]. Since their introduction, these layers have become a standard feature in most deep architectures. Instances of the layer can be positioned almost anywhere in the network. For example, in a subset of the experiments in [43], a dropout layer was placed with the input layer (in addition to their inclusion with the hidden layers), and this arrangement resulted in further reduction in the error for the recognition task. The dropout layer is always in support of another layer, such as a convolution or a fully connected layer. In this manner a dropout layer always associates with a corresponding functional layer.

Function of a dropout layer

Each dropout layer has an associated user-specified dropout probability. The dropout probability specifies the frequency with which elements of the corresponding ‘functional’ layer can be turned ‘on/off’ during a training iteration. For example, a probability of 0.5 implies that half of the neurons are randomly turned off during a given iteration. Consequently, these neurons do not take part in either the forward or the backward pass. Consider an neuron which is turned off during a training iteration k. This neuron will not take part in the current iteration, however its previous set of incoming and outgoing weights (i.e., weight value obtained/updated during iteration k-1) will still be retained and used in the next iteration k+1, given that the neuron has now been turned on. During test phase, all neurons take part in the forward pass and their weights are normalized by two.

Role as a regularizer

The dropout layer has a regularizing influence on the deep network and tends to decrease the variance and overfitting in a network. Turning off a number of layer neurons is equivalent to modifying the structure of a network layer. For a dropout probability 0.5, this behavior is equivalent to sampling from a set of 2^n different networks where n is the number of neurons in the corresponding functional layer. We experimented with the use of variable number of dropout layers in Paper III to reduce overfitting when training on synthetic datasets which were prone to overfitting.

3.3.5 Rectified linear unit (ReLU)

The ReLU is an example of an activation function which can be used in conjunction with other layers. Since introduced in [41], it has become the most frequently used activation function in deep learning and has become more popular than the hyperbolic tangent and the logistic sigmoid functions [44]. We use ReLU as an activation function in most of the architectures encountered in the Papers II, III and VII. The ReLU is formulated as follows,

$$f(x) = max(0,x), \tag{3.2}$$

where x is the input to the ReLU. It is similar to a half wave rectifier as it truncates all negative input to zero. When used in conjunction with a batch normalization layer only 50% of the ReLUs are activated at a given time thus resulting in sparse activations.

Prevention of vanishing gradients

A major benefit of ReLU over a sigmoid activation functions is that the gradient of the ReLU does not vanish on increasing the input (it remains constant at the value one as long as the input is positive) and thus training can still take place on the given unit. As an example consider the case where the input to a sigmoid is a large positive value but the output is bounded at the value one. Therefore, an increases in the input will result in only a marginal increases in the function output (consequently, a marginal value of the gradient) resulting in the aptly named vanishing gradients problem. The gradient of a ReLU is shown below,

$$\frac{\partial f}{\partial x} = \begin{cases} 1 & if x > 0 \\ 0 & otherwise. \end{cases} \tag{3.3}$$

Dead neurons

A potential problem with ReLU is that one can learn a set of weights which will cause the unit never to get activated again by any training example in the dataset resulting in the so called ‘dead neurons’. As a toy example, consider a bias which has been set too low during the training phase. To mitigate against this problem a number of modified forms of the ReLU functions such as the leaky ReLU, parametric rectified ReLU and randomized leaky ReLU have been proposed. For details, the reader is encouraged to take a look at a recent publication which performs a quantitative evaluation of these activation functions [45]. It should be kept in mind that most of these modified forms of the ReLU are fairly new and haven't been tested enough to form a conclusive opinion about their utility and generalization to different types of data and network architectures.

3.3.6 Batch normalization

The batch normalization (BN) layer was introduced in 2015 [46], and has over a very short time become popular especially for training very deep neural networks. This layer is typically introduced before a non-linearity such as the ReLU operation and performs a normalization, over the minibatch, for each set of feature responses being fed into the BN layer. The BN layer transforms the set of feature responses corresponding to each feature detector into a standard normal distribution. After the mandatory initial normalization it retains the option to learn and apply an inverse transform by rescaling and re-translating the normalized features. The parameters for inverse transformation are learnt as part of network training.

In theory, the normalization may seem like an insignificant operation especially when the data being fed into the network at the input layer might have already been normalized. However, in practice the batch normalization plays a very important role in learning by ensuring that only half of the feature responses are above the activation threshold of the subsequent non-linear activation. Therefore, after a BN operation, the contribution of a feature response to the subsequent non-linear activation is dependent on its relative value and not on its absolute value. Consider an example where all the feature responses being fed into a ReLU have large positive values, thus resulting in the activation of all ReLU units. On the other hand, all negative input values may result in a case where none of the non-linearities are activated. However, the inclusion of a BN layer transforms this entirely positive or negative range of values to a zero-mean unit-standard-deviation distribution thus ensuring that half of the feature responses always result in an update to the gradient thus potentially avoiding the 'dead neurons' condition where negative inputs to a ReLU can result in the situation where no updates to its weights can take place.

3.3.7 Loss functions

In a supervised regime for training deep networks for the classification task, the data is provided in the form of data and target pairs (x_i, t_i), where x_i is a given input data example and t_i is the corresponding class label. If we consider the neural network to be a function f composed of a number of iterative applications of linear operations and non-linear activations, then as a broad generalization the output of f for a particular input vector x_i can be written as,

$$y_i = f(x_i, W), \tag{3.4}$$

where W is the set of network weights which serve as the optimization parameters. Intuitively, one can see that the goal of network training is to minimize the difference between the output y_i and the target t_i over the set of images in our image set. The metric used for characterizing the goodness of the results is typically referred to as the loss function. There are many different types of loss functions in literature such as the Softmax, Hinge and Euclidean loss functions [47].

Softmax loss

Nowadays, one of the most commonly used loss function is the Softmax loss which treats the output y_i as unnormalized probability of predicting a particular class label. Softmax loss was used as the loss function in Papers II, III and VII. Here we add some more detail to our notation to represent the fact that the classification problem typically deals with classifying data into two or more categories. Therefore we can modify our notation to represent the output as y_{ij} and the target vector as t_{ij} where i is a data point and j is the class. As an example in a binary classification problem each data point x_i (taken from a training set or minibatch X) will produce two values y_{i1} and y_{i2} which represent the unnormalized probabilities for the two classes. Similarly the target vector for a particular data point will also contain two elements, t_{i1} and t_{i2}, where the element corresponding to the correct class t_{ij} is set at one and the other element is set at zero.

Given the modified notation the loss for a given data point i is then represented as,

$$L_i = -\log\left(\frac{e^{y_{ik}}}{\sum\limits_j e^{y_{ij}}}\right), \tag{3.5}$$

where y_{ik} is the output corresponding to the correct class k and is normalized by the values y_{ij} which form the complete output vector for the input example i. Since neural networks typically work with a minibatch of n examples at a time. The overall loss is computed by averaging the individual losses over the n examples, as below,

$$L = \frac{\sum\limits_{i=1}^{n} L_i}{n}, \tag{3.6}$$

in addition to the data-matching term L, it is also quite common to add a regularization term R weighted by a weight λ. Typically, this term is calculated over the set of network weights W indexed by an iterator p. The regularization term is given as,

$$R = \sum_p W_p^2, \tag{3.7}$$

in this case, the final loss L_{final} is given as,

$$L_{final} = L + R. \tag{3.8}$$

The derivatives of the loss function, i.e., Equation 3.8, are used for updating the weights of the layer preceding the final layer in the network. Similarly the weight updates take place sequentially through out the network as the gradient is propagated back through the network using computation of analytical gradients and the chain rule.

3.4 Optimization

Training of a neural network is an iterative optimization process. The computaton of the loss function L_{final} is dependent on two different quantities: (i) The set of network weights W; (ii) The set of input examples constituting the current minibatch X.

Out of these two inputs, the set of networks weights W is treated as the parameters of the optimization, that is, given the data X, the set of weights W are updated/selected such that the training data can be categorized into the correct constituent classes. In literature, a number of different optimizers have been proposed for minimizing the network loss [48]. Most of these optimizers are gradient-based optimizers designed on a local search paradigm. Some of the common optimizers available in the prevalent deep learning libraries are the Gradient descent (GD), Minibatch gradient descent (MBGD), Stochastic gradient descent (SGD), AdaDelta (based on an adaptive learning rate), Adaptive gradient, Adam and Nesterov's accelerated gradient [49, 50, 51, 52]. Despite the availability of this rather large set of optimizers for potential use in the deep network, the MBGD remains still remains one of the most widely used optimizer in practice. In particular, MBGD was used as the optimizer for the experiments in this particular thesis.

3.4.1 Differences between the GD, MBGD and SGD optimization methods

We emphasize that the main difference between GD, MBGD and SGD is the size of the training set used in each iteration of the optimizer. More specifically, GD computes gradients using the whole training set, the MBGD typically utilizes a small subset of the training set, called the minibatch, in each iteration where as for the SGD the minibatch size is set as only one training example per iteration. However in general practice, it has become common to refer to the MBGD as SGD even when the minibatch size is typically more than one. For example, the widely used deep learning library 'convolutional architecture for fast feature embedding (Caffe)' refer to the minibatch-based training regime as the SGD [36].

3.5 Well known network architectures

In the last five years a number of different network architectures have been proposed. Some of these have shown record breaking performance on the widely available and used image sets. Some of the significant examples include AlexNet which won the ILSVRC 2012 classification challenge [32]. The network in a network (NiN) was one of the earliest examples of a modular design of neural networks where the deep network is constructed using a repetitive arrangement of similar blocks or components [53]. Another well known modular design is the Inception network (also known as the GoogLeNet) which set the state of the art in the detection and recognition challenge at the ILSVRC 2014 while still using 12 times fewer parameters/weights, in the form of a 22-layer deep network, than what the AlexNet used in 2014 [54]. A number of models of varying lengths have been proposed by the Visual Geometry Group (VGG) at Oxford and out of these the 16 and 19 layers models reported results which were almost competitive with the state of the art (i.e., GoogLeNet) at ILSVRC 2014 challenge [55]. The residual networks (ResNets) by He et al., which won the ILSVRC 2015 classification challenge are an example of a radically different design for creating networks which can be exceedingly deep (networks of different depths including 152 and 1000 layers deep networks were evaluated) yet substantially lower number of parameters mostly due to the use of very small 3×3 convolution filters [37]. Finally the discussion would be remiss without mentioning the classic LeNet architecture which, although much older than the other aforementioned architectures, has served as a blueprint for many deeper architectures.

In this thesis the experiments for Paper II were limited to the AlexNet, NiN and the LeNet-5 architectures. In addition, the architecture used for Paper III was inpisred by LeNet. We used an AlexNet in Paper VII. We briefly discuss the three architectures (i.e., AlexNet, NiN and LeNet) in the forth coming subsections.

3.5.1 LeNet

The LeNet architecture can arguably be called the precursor to the modern neural networks [28, 29]. The LeNet architecture was a seminal effort towards the design principal of placing a number of convolution layers, interspersed by down sampling layers, which are then followed by a number of fully connected layers. LeCun et al., have proposed a number of similar architectures which are referenced with the prefix ‘LeNet’. The LeNet-5 comprises three convolution and two fully connected layers employing hyperbolic tangent non-linearities and a Euclidean loss function.

3.5.2 AlexNet

The AlexNet follows a classic convolution architecture which comprises five successive convolution layers of filters with sizes 11×11, 5×5, 3×3, 3×3 and 3×3, respectively, followed by three fully connected layers [32]. The five convolution layers (i.e., forming the convolution stack) are repeated in two parallel columns. The random nature of the initialization ensures that the filters in the two columns end up learning quite different features, for example, in the original article one of the convolution stacks ended up learning the edge detectors where as the other one learnt color patterns representing the low frequency background.

3.5.3 Network in Network

The NiN architecture follows a modular design [53]. This concept of modular design has also served as the basis of the Inception model adopted by the designers of the GoogLeNet [54]. The network consists of a number of identical micro networks which can be connected end-to-end to create networks of arbitrary depth. The NiN is inspired by the philosophy that if a micro network can result in acceptable performance then connecting and training a number of these micro networks in a serial formation should register even better performance. Although, the principal does not always hold in practice since adding a large number of additional micro-networks may result in problems typically associated with very deep networks such as overfitting and high optimization cost. In fact, Lin et al. [53], only used three of these micro networks, each composed of a convolution layer followed by a couple of fully connected layers, to build their composite classification solution.

3.6 Data augmentation

Depending on the task at hand, deep learning methods may require more training data than is available. In addition, limited training data may result in overfitting, evidenced by much higher accuracy on the training set as opposed to the validation and test sets. This need for more data can be mitigated by using data augmentation as a preprocessing step. Typically, data augmentation refers to generation of more data for training by suitable transformation of the existing data.

The applied transformations are application dependent. For example, in our spot detection framework (Papers II and III) the fluorescent spots lack a dominant direction, that is, a rotated spot (180 degrees rotation) would be considered as valid as a non-transformed one. On the other hand, in the typical computer vision datasets such rotations would not considered suitable since there is limited motivation to rotate images representing cars because of a very low probability of finding upside down cars in standard photography (unless one happens to visit a junkyard). Typical transformation for data augmentations involve flips, translations, rotations, shearing, rescaling, contrast enhancement, illumination changes for natural scenes. Some of these augmentations, applied for the image recognition task, have been discussed in [56]. The data augmentation step adds more diversity to the data and is useful provided the transformations are of the kind which are suitable for the data and task at hand.

When working with limited datasets where the value of each data point is at a premium, it becomes difficult to further exclude a subset of data from training by designating it as the test set. A work around for such cases is to perform *k*-fold data sampling and cross validation where the data is divided in *k* segments and *k*-1 of the segments are used for training and one segment for testing. In our experiments *k*-2 segments were used for training and out of the remaining segments, one was used for validation and one for testing. The train/test cycles are undertaken *k* times and each time a different segment is selected for testing. In this manner the final result is the mean of scores from the *k*-times trained and tested networks and thus is unlikely to be biased by a particular subset of data. After testing, one deployment strategy (i.e., when deploying on novel unseen data) could be to treat the *k* differently trained networks as parts of an ensemble and to average their predictions before assigning class labels.

Finally, approaches such as pretraining on a suitably large external image set or using an already pretrained network followed by fine-tuning on the internal training set have also been shown to improve performance in some cases [57]. Our approach, in Paper III, towards generation of semi-synthetic data by sampling from distributions generated from real data can also be considered as an alternative way of generating additional data for data augmentation purposes.

3.7 Hyperparameter tuning

The convergence of neural networks to an optimum solution is dependent on a number of factors such as the network design (including depth and breadth of the network as well as the choice of the activation functions), choice of the loss function, choice of optimizer, base learning rate as well as the schedule of changes to the learning rate. The progressive increase in the computation power (especially with the leveraging of the GPUs) has made it possible to search over a narrow range of some of these parameters. Recently, it has been claimed that hyper parameter tuning methods such as the Bayesian approach by Snoek et al., [58] could be used to further improve network performance. An alternate approach involves a grid search strategy. Typically, learning rate and the choice of the loss function are the more commonly optimized hyper parameters. In practice, it is much more common to take an ad hoc approach and optimize the network performance over a narrow range of these parameters rather than engage in extensive grid search or Bayesian hyperparameter tuning.

3.8 Applications of deep neural networks

In the last few years, deep neural networks have been applied to a wide range of problems. In particular the convolution neural networks have been extensively applied to object detection and recognition tasks where they have recorded state of the art performance [42, 32, 53, 37, 54, 59, 60]. One of the early applications of these networks was for dimensionality reduction when Hinton et al. employed a layer wise training paradigm to train a deep autoencoder [31]. These networks have also been used for super resolved image reconstruction [61]. Recently, a particular design of convolution neural networks, called the U-Net, has been used for segmentation of biomedical images and has resulted in excellent performance on a number of image segmentation challenges [62]. Moreover, human level performance (using only visual information) has been reported on computer games [63]. Finally, impressive results on tasks such as counting calories from images of food [64], pose estimation [65], estimation of optical flow [66], tracking [67] and speech recognition [68]. As listed above, one can see that deep neural networks have found applications in a number of different areas, including image analysis.

4. Compressed Sensing

4.1 Introduction

Compressed sensing is a signal processing technique which enables compressed measurement of sparse signals using very few samples. Subsequently, the original signal can be reconstructed at a high quality (exact reconstruction is also possible under suitable assumptions about noise and number of acquired samples) through an optimization framework [69, 70, 71].

During signal measurement, compressed sensing combines acquisition and compression in one integral step. This is in contrast with the standard acquisition and compression systems where large amounts of data are first acquired and then compressed to smaller sizes (e.g., compression of images after acquisition from a charge-coupled device (CCD) camera). The decompression or recovery step is similar to solving an under-determined linear system under a few assumptions about the original signal (i.e., signal should be sparse) and the signal acquisition system [71]. The quality of the reconstructed signals is affected by aspects such as noise and the number of acquired samples. Exact reconstruction may not possible with a noisy signal. Moreover, there are constraints on the minimum number of samples required to reconstruct the signal and given similar conditions the reconstruction quality increases with an increase in the number of samples. A compressed sensing-based reconstruction scheme is used in Paper I.

The compressed sensing framework comprises two components: (i) an encoder or compressed acquisition component; (ii) a decoder or decompression component.

4.2 Encoder

The measurement or sampling of a signal in a compressed sensing framework is different from the one in standard acquisition systems. Lets start with a simple and intuitive example of a CCD camera. Consider an image of a natural scene produced using a CCD camera. It comprises a two dimensional array of pixel positions, sometimes also referred as the image grid or the sampling lattice. Suppose there are n such pixels in the image. Each pixel captures some information about the imaged scene. Removal of some of this information (e.g., by setting some pixel values to zero) typically results in some loss of details (note that we use the term information in a general sense here and are not referring to information as defined in information theory) and depending on the scale of the information which has been removed, it may or may not be possible to reconstruct the original image again. For example, reconstruction of objects existing at very small scale may be difficult because information about such objects may be lost even by removal of very few pixels.

Consider the case where the number of retained or unaffected pixels m is much smaller than n,

$$m \ll n, \tag{4.1}$$

in such a case reconstruction seems quite challenging. On the other hand, compressed sensing claims that such reconstruction is possible under a few conditions and assumptions [72], which are discussed next in this section.

4.2.1 Type of samples

For compressed sensing, the type of these m measurements is quite different from the n pixels discussed earlier. Rather than just picking an m-sized subset of the original pixels, intuitively one can think of these measurements as the projections of the original data (considered as an n dimensional data point) onto a lower m dimensional space. In this manner, the compression can be considered as a form of dimensionality reduction.

More precisely, by labeling the vector of n original measurements/data as x such that $x \in \mathbb{R}^n$ and the vector of m transformed/retained measurements as y such that $y \in \mathbb{R}^m$ we can depict this transformation as,

$$y = Ax, \quad where\, A \in \mathbb{R}^{m \times n}. \tag{4.2}$$

Therefore each element of y is a weighted sum (weighted by the matrix A) of all the elements of x. That is, each element of y is capturing some information from all the elements of x, examples of a few such systems are available in Section 4.2.2. Therefore, these measurements y are in a sense more ‘global’ than the original measurements x. The matrix A is called the sensing matrix and specifies the operation of the acquisition system in the form of the basis vectors used for data compression (in Paper I the matrix A encodes the effect of the PSF of the image acquisition system). In real applications, where the measurements may include noise, the Equation 4.2 is modified as follows,

$$y = Ax + \varepsilon \tag{4.3}$$

where epsilon specifies the noise included in the measurements. Moreover, exact recovery of x may not be possible under noisy conditions (e.g., in Paper I, we note that reconstruction of super resolved microscopy images degrades with the increase in image noise).

4.2.2 Examples of acquisition systems

Equation 4.2 imposes certain constraints on the acquisition systems used for compressed sensing and not all systems can be considered suitable for acquisition under this paradigm.

Single pixel camera

An example of such an imaging systems is the single pixel camera [73], where a digital micromirror device (DMD) is used for acquisition. A DMD is composed of a number of very small mirrors. Each of the mirrors can be very quickly set to either point towards, or away from a single photodiode (i.e., a single pixel detector). In this way, light from certain parts of the scene can be projected onto the detector and light from certain parts projected away from the detector. Observe, that the light reaching the detector is the 'sum' of the reflected light from the mirrors facing the detector.

By connecting the DMD with a random number generator, the sequence of projecting and non-projecting mirrors can be changed very quickly. Each of these sequences correspond to a row in the sensing matrix A and results in acquisition of one compressed sample. The sequence of mirrors is randomly changed m times to acquire the required number of compressed samples. For this system, each row of the sensing matrix A is composed of binary valued elements, where the elements with the value 'one' represent the projecting mirrors and the elements with the value 'zero' represent the non-projecting mirrors.

Acquisition of m number of these single pixel measurements enable reconstruction of the original image. Such acquisition frameworks are some times called 'coded imaging systems', for more details refer to [74, 75].

4.2.3 Sensing matrix

The sensing matrix A comprises m rows and n columns. As seen in the case of a 'single pixel camera', the selection of this sensing matrix is closely coupled with the acquisition system and acquisition hardware thus curtailing the design freedom in selecting or constructing these matrices [76]. For the *Faster STORM* method discussed and evaluated in Paper I, the A is meant to capture the effect of the PSF on a point source of light and is therefore modeled on a Gaussian function.

For a general application, a simple yet effective way of building a sensing matrix is to randomly populate the matrix with +1 and -1 values and normalize by column lengths to get unit vectors [74].

In addition, the column vectors of A should satisfy a few additional conditions such as that they have low mutual coherence and obey the restricted isometry property (RIP) [77].

4.3 Decoder

The original signal x can be decoded from the observation/measurement y through a decoding or decompression step which involves non-linear optimization [71, 76]. Therefore, for compressed sensing the decompression step is computationally more expensive than the compression one. High quality reconstruction of the original data x is dependent on a number of factors such as the sparsity of the x, noise in the measurements and choice of optimization method.

Note that the effect of sparsity and noise on the signal reconstruction is very tightly coupled with the choice of the optimizer. In Paper I, we found that convex optimizers were more resilient to higher noise and low sparsity conditions than the greedy optimizers when applied to super resolved reconstruction of diffraction limited microscopy images, that is, the performance of greedy optimizers was found to notably degrade for noisy images with large number of fluorophores.

4.3.1 Sparsity

Equation 4.2 suggests that compression is a trivial task of projecting the data x onto the lower dimensional compression space through a inner product. However, the reconstruction poses more problems; as is apparent from the Equation 4.2 we are dealing with an under determined system implying that an almost infinite number of solutions for x could have resulted in the recorded measurements y. However, this large space of solutions can be reduced by imposing more restrictions on the signal x and consequently the recovered solution x^*. To better motivate this problem consider the toy example represented below,

$$5 = x_1 + x_2, \tag{4.4}$$

as one can see the solutions for this under determined linear system can span a two dimensional infinite plane. However, if we add the restriction that the solution needs to be non-negative, than the set of possible solutions is somewhat reduced. More over, an additional restriction that the solution need to be 1-sparse, that is, have at most one non-zero element reduces the system to only two solutions where either $(x_1 = 5, x_2 = 0)$ or $(x_1 = 0, x_2 = 5)$.

It turns out that for larger systems the requirement for sparse solutions can still result in high quality reconstructions of x (as long as the signal x was actually sparse in the first place). In addition the number of measurements m is related to the sparsity of the original data, that is, more sparse signals can be reconstructed with fewer measurements. The relationship between m and a k-sparse signal is represented as,

$$m \geq C \cdot \mu^2 \cdot k \cdot \log(n) \tag{4.5}$$

where C is a positive constant and μ is the mutual coherence of the sensing matrix [78].

4.3.2 Transform domain representation

In addition to the sparse signals, compressed sensing can be applied to non-sparse signals as long as those signals can be transformed to another domain where they have a sparse representation. As an example, consider image data which may not be itself sparse, but a Fourier transform of the image is sparse as some of the coefficients are enough to reconstruct the image. In general if the data x can be decomposed to a set of transformation basis Φ and set of coefficients α, that is,

$$x = \Phi\alpha, \tag{4.6}$$

then the system in Equation 4.2 can be expressed as,

$$y = A\Phi\alpha. \tag{4.7}$$

4.3.3 Solution of a sparse system

The reconstruction problem can be expressed in multiple alternative formulations, shown below,

$$min\ \frac{1}{2}||Ax-y||_2^2 \qquad subject\ to\ ||x||_1 \leq k \tag{4.8}$$

$$min\ ||x||_1 \qquad subject\ to\ ||Ax-y||_2 \leq \varepsilon \tag{4.9}$$

$$min\ \frac{1}{2}||Ax-y||_2^2 + \lambda||x||_1 \tag{4.10}$$

where k is a parameter controlling the sparsity of the solution and ε is the measure of noise in the data [76]. All three equations can result in the equivalent solutions for x given a careful selection of the parameters k, ε and the weighting term λ. The first two equations are referred to as the constrained formulations and the last one is the unconstrained formulation of the problem. The selection of the most suitable out of the three formulations is dependent on the application/problem at hand and the type of information available. For example, the spot detection and localization method *Faster STORM* evaluated in Paper I uses Equation 4.9 as the objective for minimization [21]. In case of transform domain processing all instances of Ax in equations 4.8 – 4.10 are replaced by $A\Phi\alpha$.

The three different formulations of the problem (i.e., Equations 4.8 – 4.10), can be solved by convex solvers and optimizers. Further, a large number of solver packages and libraries are available such as the CVX package from Boyd et al. [79], the *ℓ1-magic* by Candes et al. [80], as well as the *iterative hard thresholding* (IHT) optimizer based on a greedy optimization strategy [81]. A list of such solvers is available at [82]. The CVX and a normalized version of the IHT were employed by us in Paper I.

4.3.4 Comparison with compression schemes

Compressed sensing differs in a few important ways from standard compression algorithms. First, in compressed sensing the data acquisition and compression takes place simultaneously where as in standard compression techniques the data is first acquired and then compressed. Therefore, applications where acquisition time is of a premium, benefit from a compressed sensing approach, such as magnetic resonance imaging and computed tomography tasks [83, 84, 85].

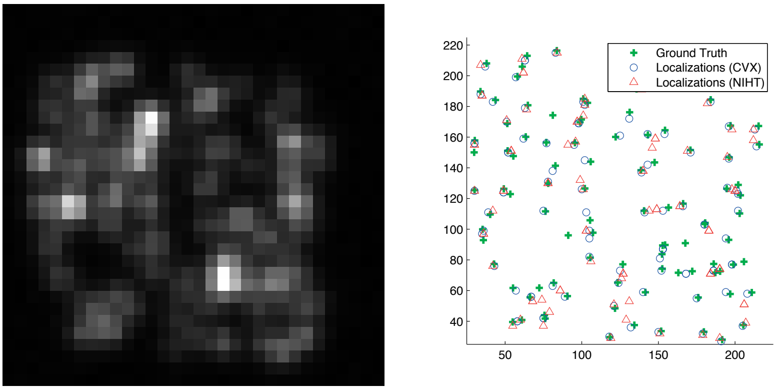


Figure 4.1. Synthetic image data (left) and localization positions (right). The synthetic image represents a molecule density of 100 molecules per 32×32 pixels image. The plot on the right displays the ground truth and the localized positions for the convex (CVX) and greedy (NIHT) solvers.

4.3.5 Application to super resolved microscopy

Compressed sensing has been applied to identification and super resolved localization of fluorescent spots [21]. In Paper I, we evaluate the suitability of this approach against variation in noise, fluorophore density, choice of optimizer as well as the estimate of the system PSF. Subset of results are shown in Figure 4.1.

The images were acquired using a standard fluorescent microscope and appeared as diffraction limited spots due to the effect of the PSF. The acquired image was treated as the observation y, the sensing matrix A encapsulates the effect of the PSF and the goal of the decoder is to recover the positions of the point sources of light (expressed as spots after convolution with the PSF). The position of these point sources was estimated to subpixel precision by using a localization grid of x which is much finer than the image grid. The data fulfilled the sparsity requirement since the ratio of foreground to background pixels in such applications is quite low. For details please refer to Paper I.

5. Overview of papers and summary of contributions

The chapter briefly summarizes the methods developed during the course of the Ph.D. studies as well as the contributions for each paper included in the thesis. Each section represents one of the included papers.

5.1 Evaluation of *Faster STORM*; a compressed sensing-based method for spot localization (Paper I)

5.1.1 Background

Faster STORM is a compressed sensing-based method for detection and localization of fluorescent spots [21]. This highly cited method is considered part of super resolution microscopy. Super resolution microscopy enables lateral resolutions of $\sim$20 nm against resolution of $\sim$250 nm from conventional fluorescence microscopy [86, 87]. Unlike techniques such as the 'stimulated emission depletion microscopy' (STED) which use specialized optical instruments to modify the effective PSF of the excitation beam to acquire super resolved images [88], *Faster STORM* is a post acquisition method applied to images obtained from standard fluorescence microscopes. Moreover, it enables identification of fluorophores even at very high activation densities (i.e., overlapping fluorophores).

5.1.2 Use of a compressed sensing approach

Faster STORM is based on a compressed sensing approach [69], where recovery of a super resolved image from an acquired diffraction limited image is treated as an under determined linear system. High quality reconstruction is possible under constraints on the sparsity of the solution as well as the structure of the transformation representing the acquisition process. An overview of compressed sensing is presented in Chapter 4. For *Faster STORM*, the solution to the reconstruction problem is recovered through the *CVX* convex optimization software [79].

5.1.3 Drawbacks of *Faster STORM*

The *Faster STORM*, though novel and reasonably effective, suffers from a number of drawbacks. The convex optimization process is quite slow and time consuming, typically requiring a few hours to process a medium sized image. Moreover, it requires an estimate of the PSF.

5.1.4 Acceleration of *Faster STORM* by parallel processing

We accelerated the method by modifying the code provided by the authors of *Faster STORM* (i.e., [21]) so that it can be executed over multiple CPU cores. The modification incorporated parallel *for* loops and changes to the program data structures. The parallelization allowed a code speedup which was proportional to the number of CPU cores in the system. The original method took 2041 seconds to process a 480×480 pixel sized image. After the acceleration, the modified code took 559 seconds over a 4-core CPU machine thus providing an almost 4-times speed boost.

5.1.5 Effect of variation in the PSF

As mentioned earlier, *Faster STORM* requires an estimate of the PSF. Therefore, an error in estimating the PSF can affect the recovery of the super resolved image. We quantified this effect on recall and localization precision by changing the standard deviation of the symmetric PSF from -30 to 30 percent with a 10 percent step size. We found that small variations in the PSF can have a large impact on the signal recovery.

5.1.6 Comparison with a greedy optimizer

We evaluated the performance of the method when the optimization in the decoding step is performed by a greedy optimizer rather than the convex *CVX* software. We selected the 'normalized iterative hard thresholding' (NIHT [89]) method as the optimizer for comparison with the CVX. The comparison was performed under varying conditions of noise and fluorophore density. Representative localizations are displayed in the Figure 5.1.

5.2 Application of deep learning for spot detection (Paper II)

5.2.1 Background

Detection of fluorescent biomarkers is an important part of bioimage informatics. A large number of such detection methods have been proposed. In general, these methods can be grouped in two categories. Using the taxonomy employed in [1], these groups/classes of detection methods can be categorized as supervised and unsupervised. Since the terms supervised and unsupervised

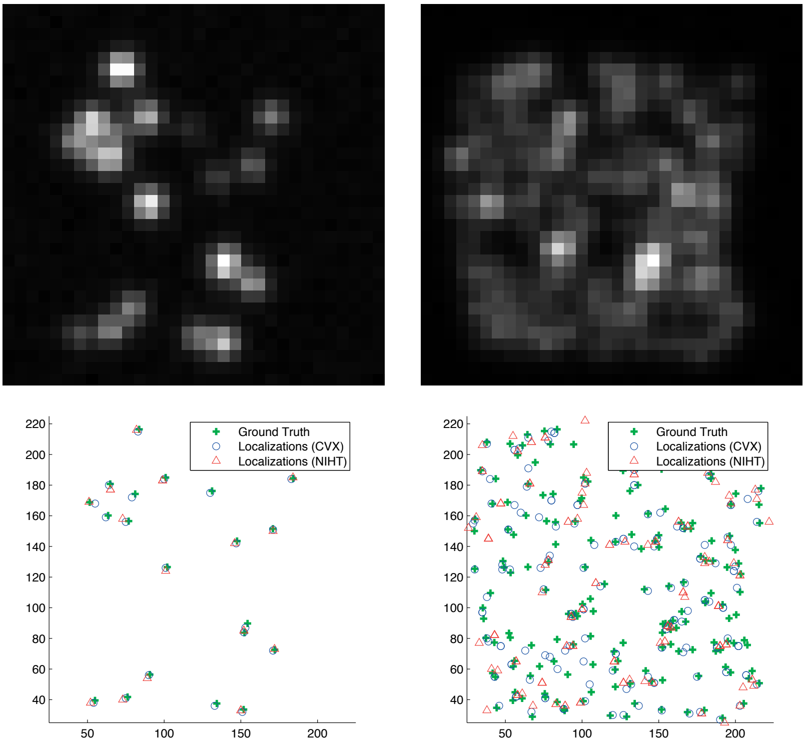


Figure 5.1. Synthetic image data (row 1) and localization positions (row 2) at different fluorophore densities. For each row the columns from left to right represent two different molecule densities: 20 and 180 molecules per 32×32 pixels image. The second row display the ground truth and the localized positions for CVX and NIHT solvers.

have a particular meaning in machine learning (which is different from the way these terms were used in [1]), we can alternatively refer to these groups as data-driven and model-based (i.e., non-data-driven). The data-driven methods typically calculate features on annotated positive and negative examples and train a classifier to minimize the classification error [11]. In model-based methods, a user defined detection model is used for finding and identifying the fluorescent spots [18]. Data-driven methods in general have been relatively infrequently used for spot detection.

Recently deep learning methods have found a number of applications in image analysis and computer vision [42, 32, 62, 64]. However, to the best of out knowledge these methods have not been applied to spot detection.

5.2.2 Quantitative comparison of spot detection methods on real data

We compared the spot detection performance of a number of popular deep architectures [32, 28, 53] against shallow machine learning methods. Moreover, two model-based (i.e., non-data-driven) detection methods were also compared, thus resulting in a comparison of over nine different spot detection methods.

All comparisons were performed on real data annotated with an in house annotation tool called the *SpotObserver* which was based on a 2AFC approach. Details of the annotation tool are provided in Paper III. Typically, for such methods, quantitative results are provided for synthetic data, and for real data, the results are either qualitative, or computed over small datasets. However, in this work we performed all comparisons on manually annotated real data comprising 792 data points.

5.2.3 Effect of pretraining on detection performance

Data-driven methods require annotated data for training. Typically, this annotation is a manual process which is costly and time consuming. One potential way of mitigating this need for large annotated datasets is by pretraining the deep networks on external datasets such as the CIFAR-10 [90], and then subsequently finetuning on the biomedical image data available. We evaluated the effect of pretraining on a modified LeNet type architecture [28], and compared against an instance of the same network trained without any pretraining. We concluded that for our spot detection task, pretraining does not provide any significant benefit. On the other hand, the pretrained network converged faster than its non-pretrained counterparts. A subset of the convolution filters learnt during network training are shown in Figure 5.2.

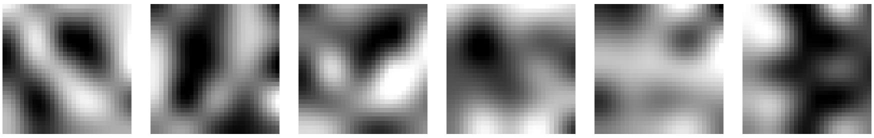


Figure 5.2. Subset of feature maps (resampled over larger grid) from the first convolution layer for the LeNet architecture.

5.2.4 Effect of decreasing the data size

We conducted experiments where the data-driven methods for spot detection was trained on progressively smaller training sets and found that some of the deep networks were surprisingly resilient to small amounts of training data.

5.3 Training spot detectors on synthetic, semi-synthetic and real image sets (Paper III)

5.3.1 Background

As mentioned earlier, a major part of the thesis has focused on spot detection methods and their evaluation, especially data-driven methods for spot detection including the deep methods which have become recently popular. It is typically perceived that data-driven methods require a large amount of training data and this data due to its very nature (i.e., biomedical images) requires expert annotation which is time consuming and costly. An alternative approach is to train the data-driven spot detectors on multiple synthetic and semi-synthetic datasets. To gauge the effectiveness of such an approach we trained the spot detectors on these synthetic datasets, evaluated the spot detection performance on annotated real data and compared this performance against the case where training took place on real data. In addition, we evaluated the detection performance on progressively smaller subsets of the training data to determine how much training data is really needed for the spot detection task. We developed an inhouse tool for effective manual annotation of spots in real images. Finally we evaluated the reliability of the manual annotations performed on real data.

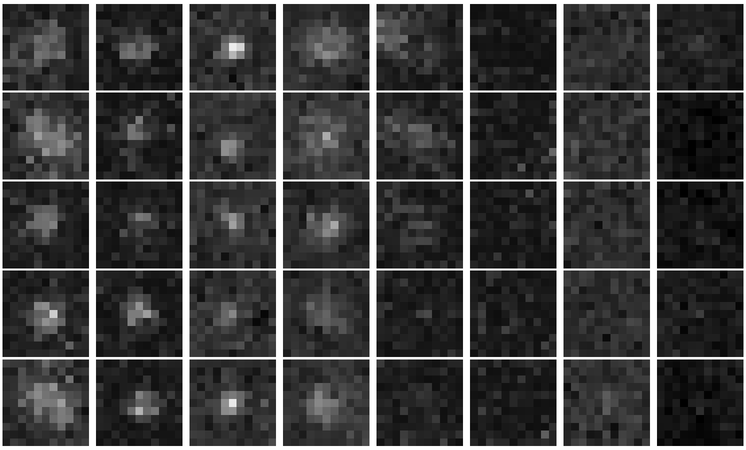


Figure 5.3. Spots and background from real, synthetic and semi-synthetic image sets. Columns 1–4 represent the positive examples from the real, *SMeagol* generated synthetic, off-the-shelf synthetic and the semi-synthetic image sets, respectively. Columns 5–8 represent the negative examples in the same order. Images were generated under similar conditions/simulation parameters values.

5.3.2 Datasets

We used two synthetic, one semi-synthetic and one real dataset for training. The testing was done only on real data. The two synthetic datasets were generated from two different software where one was a state of the art method for generating synthetic spots, called *SMeagol*, (but required extensive specification of simulation parameters) [91], and the other one was a more generic or off-the-shelf method requiring less parameter tuning (used for generating synthetic data in [21]). The semi-synthetic datasets was produced by sampling from probability distributions built from the real image set. The real dataset comprised multiple low and high fluorescence frames from an imaging sequence. Positive and negative examples from the four datasets are shown in the Figure 5.3.

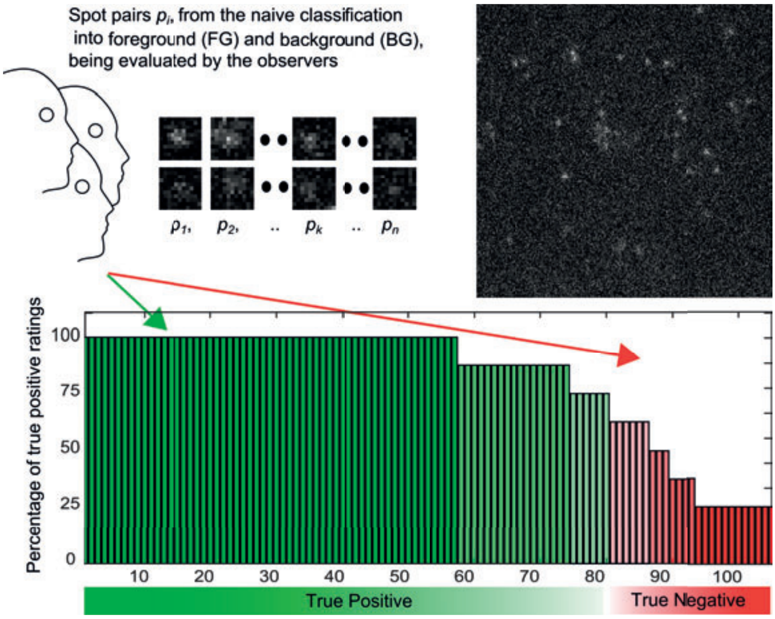

Spot pairs p_i, from the naive classification into foreground (FG) and background (BG), being evaluated by the observers
p_1, p_2, .. p_k .. p_n
Percentage of true positive ratings
100
75
50
25
0
10
20
30
40
50
60
70
80
90
100
True Positive
True Negative

Figure 5.4. Fluorescent spot-pairs $p_1, p_2, ...p_k, ...p_n$ being annotated as either true positives or true negatives using a two-alternative forced-choice approach and a user specified decision boundary.

5.3.3 Tool for annotation of real data

In order to quantitatively evaluate the effect of training on a particular dataset the test/evaluation should be performed on real data annotated by experts. Therefore, we created a tool, called *SpotObserver*, for assisting the experts in the annotation of these fluorescent spots. The tool is based on a 2AFC approach where the user is asked to decide which of the two shown examples looks more like a spot rather than the standard annotation approach where the user is asked whether a shown example is a spot or not. Such an approach has previously been found to be quite useful for annotation of biomedical data

[19]. A visual representation of the approach is shown in Figure 5.4. The real data was extensively annotated by as many as nine experts resulting in 1130 positive and negative training examples.

5.3.4 Evaluation of different types and sizes of training data

We evaluated the training of the data-driven spot detection methods on these varied datasets. All four datasets (synthetic, semi-synthetic and real) were augmented with transformations such as flips and rotations. We used a 5-fold validation scheme for evaluating the results. The spot detectors trained on this data comprised a linear discriminant analysis-based classifier, a boosted ensemble of decision trees and a DCNN. For the first two shallow methods a features set based on Haar-like features was used. The testing was performed exclusively on real data. We concluded that the use of either manually annotated real data or its carefully parameterized synthetic counterpart for detector training was always preferable to training on off-the-shelf data generators.

We identified the number of real or semi-synthetic training examples needed for acceptable spot detection performance. This is useful in the case where the required synthetic data generation expertise is not available. We progressively reduced the size of the real training set while recording the spot detection performance and found that for our task, the deep method appeared to be quite resilient to fewer training examples.

5.3.5 Analysis of annotation variability

We also analyzed the variability of the annotations performed by nine experts. The variability was compared against baseline obtained from random allocation of annotation labels and we found that the annotations from different raters had high mutual agreement, indicating a low variability. More over, we compared the annotations performed using our tool against that from a naive point and click method and found that our tool had better performance as quantified by the F-score.

5.4 Analysis of the effect of compaction oligonucleotides on fluorescent signals (Paper IV)

5.4.1 Background

Rolling circle amplification (RCA) is a technique for replication and generation of multiple copies of circular DNA or RNA. These copies are concatenated together to form a long strand called a rolling circle amplification product (RCP). An oligonucleotide is a short DNA or RNA molecule which has multiple biological applications including its use as a molecular detection probe. These oligonucleotides can be designed and synthesized to attach or *hybridize* to complementary sequences in an RCP strand. In other words, the oligonucleotide can act as a fluorescent probe for detection of particular DNA or RNA subsequences in an RCP. Typically, an RCP collapses and folds onto itself, however sometimes an RCP can split into multiple clusters which results in a decrease in the density of the fluorophores. Consequently these signals become harder to find and may be mistaken for multiple signals.

5.4.2 Effect of compaction oligonucleotide

A compaction oligonucleotide comprises of two molecular probes, one at each end. Each of these probes can hybridize to a nucleic acid sequence present at a different locations in the RCP strand. Intuitively, one can think of a compaction oligonucleotide as a double ended lasso which can attach to and bring together different parts of an RCP strand, thus resulting in a more compact RCP. In theory, such an RCP should appear brighter in fluorescence microscopy images and is less likely to split into clusters, thus making it more suitable for quantitative microscopy applications. In practice, such claims need to be validated by quantifying and comparing the behavior of compacted RCPs (i.e., treated with compaction oligonucleotide) against that of standard (i.e., non-compacted RCPs). More specifically, we look for evidence that the compacted RCPs are more intense and less likely to split into smaller clusters. An example of compacted and non-compacted/standard RCPs is shown in Figure 5.5

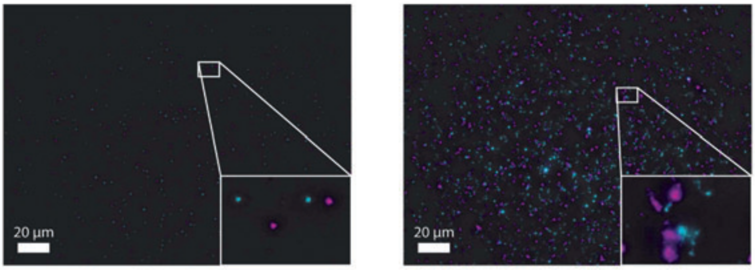


Figure 5.5. Compacted (left) and non-compacted (right) RCPs.

5.4.3 Measurement of signal intensity

First, a maximum and minimum intensity projection was calculated for a 16 level z-stack of input images. The background fluorescence was removed by subtracting the minimum intensity projection from the maximum intensity projection. The signals were identified by a thresholding step where the threshold was selected as the mode of the image histogram. Then the signal intensity was quantified through calculation of SNR for the identified signals. Results indicated that the compacted RCPs tend to have much higher SNRs then their non-compacted counterparts.

5.4.4 Measurement of signal integrity

For measuring signal integrity (i.e., tendency of a signal to break into clusters), we used signals with two different colors. The images were filtered with a Gaussian kernel and the background was removed by subtracting the median image intensity. The local maximum in each signal was designated as its center. Next we identified whether the closest neighbor to each signal was a signal of the same or different color. The analysis was performed under the hypothesis that if an RCP breaks into clusters, these clusters are more likely to be situated close to each other and consequently the ratio of same color neighbors should be higher than the ratio of different colored ones. Since the two colored signals occurred in different concentrations, we performed a simulation to predict the ratio of same color neighbors provided RCP splitting does not take place. The results indicated that for the compacted RCPs the ratio of same color neighbors was in accordance with the simulation, whereas for the non-compacted RCPs the observed ratio was much higher than the prediction, thus suggesting that compacted RCPs are less likely to split into clusters.

For signal colocalization analysis, we computed the intersection over union statistic for the pixels corresponding to the signals in the two color channels and discovered that as per expectation this measure was considerably higher for regular RCPs as compared to compacted ones.

5.5 Computational analysis of spatially resolved transcriptomics (Paper V)

5.5.1 Background

In cell biology, transcription and translation processes result in the conversion of DNA to RNA and RNA to proteins, respectively, also referred to as the expression of the cell genome. The collection of RNA molecules in a cell is called its transcriptome. Cells exhibit large transcriptional variations. The study of these variations is of considerable interest in biology. The need for high multiplexity (i.e., number of identified transcripts), high sensitivity (i.e., detection of even a few molecules) and high spatial resolution (i.e., image resolution and field of view) are some of the key challenges for the methods used in transcriptomics.

5.5.2 Description

In this survey paper, we focus on three different types of methods. First, we discuss the upstream methods for tagging and identifying the molecular targets in biological samples. Methods such as single-cell RNA-sequencing which allow study of the whole transcriptome albeit with loss of spatial resolution and localization. Laser-capture microdissection methods which retain spatial resolution but are limited by procedural difficulties. Fluorescent in situ hybridization (FISH) techniques which allow spatial resolution as well as multiplexity by using multicolored combinations of molecular probes. In most of these image-based techniques the molecular probes appear as bright fluorescent spots. Identification and localization of such signals is an important component of such a transcriptome analysis pipeline.

Next, we focus on, and identify, the midstream image analysis methods for identification and localization of the fluorescent spots. More specifically we targeted those image analysis mechanisms which were employed in the biology-focused upstream techniques. These may include methods for noise removal such as convolution with a smoothing filter (i.e., Gaussian filter) of suitable standard deviation, as well as wavelet filtering techniques. Moreover, techniques such as image deconvolution can be used for improving the SNR. Thresholding techniques such as the Otsu method have proved useful for detection of different types of signals [92]. In fact all signal detection schemes typically incorporate some form of implicit or explicit thresholding step (irrespective of whether the thresholding is on signal intensity or other measures such as size, shape or feature response of the spot under consideration). Methods such as the morphological top-hat filter, wavelet multiscale products and boosted detectors based on Haar-like features have also been utilized in a number of studies. PSF fitting methods such as DAOPHOT have also been found useful for localization of fluorescent spots [93].

Finally, the quantitative information acquired from images, especially regarding the spatial distribution of the fluorescently tagged molecular probes can be integrated and visualized by a number of downstream visualization and statistical analysis methods. Methods such as tools for visualization of information through multicolor semitransparent layers. Also included are techniques for comparison of observed spatial distribution patterns against randomly generated patterns.

5.6 Quantification of zebrafish tail deformation (Paper VI)

5.6.1 Background

Quantitative microscopy is a powerful tool for analysis of biological structures at cellular and subcellular level resolution. However, many processes and pathologies are better understood at the level of whole organisms. Zebrafish has been used as a model organism in many fields of biological research including neurobiology [3, 4]. Here, we focused on identifying and quantifying the effect of drug-induced neuronal damage. This effect is expressed as shape deformation of the zebrafish embryos due to inhibition of DNA repair. An easily observable component of this deformation is the bending of the zebrafish tail. We employ the curvature of the zebrafish tail as a feature for differentiating between affected and regular fish.

5.6.2 Confounding factors in measurement of tail curvature

The measurement of the zebrafish tail curvature is made difficult by two factors: (i) the zebrafish embryo are almost transparent which makes it difficult to distinguish them from the image background. Consequently, these tails cannot be easily segmented and their curvature measured; (ii) in order to economize the use of expensive chemical libraries in high throughput experiments, multiple zebrafish embryos are placed in the same multiwell plate, thereby causing the fish and their tails to overlap and occlude each other. Thus making it more difficult to measure the tail curvature.

5.6.3 Datasets

The image set comprised two replicates, with multiple images, acquired through a bright field microscope. Each image displayed a well from a multiwell plate, with an average of five zebrafish embryos per well. The wells were divided into negative-control and treatment cohorts. The fish in the treatment cohort were exposed to 100 nM and 200 nM concentrations of Camptothecin. Consequently, the affected fish displayed a visible curling of their tails.

5.6.4 Segmentation of zebrafish tails and measurement of its curvature

As a preprocessing step the inhomogeneous illumination was estimated and corrected through a convex hull approach followed by pixel-wise subtraction [94]. Next, we applied Gaussian smoothing to reduce image noise. Subsequently the fish were segmented using the Otsu thresholding method [92].

Morphological skeletons were generated for the foreground fish by using a branch free approach for skeleton generation. We designated this approach as refined medial axis (RMA). The RMA generation was augmented with an efficient seeding strategy to reduce the computational cost of the method. The use of RMA results in skeletonization of all parts of the fish except those where part of a fish is occluded by another fish. For such overlapping regions we proposed and implemented a *RMA fusion* step where disjoint parts of fish tail (i.e., separated due to occlusion) are fused/joined together to yield a complete fish tail. The steps in the image analysis pipeline are depicted in Figure 5.6.

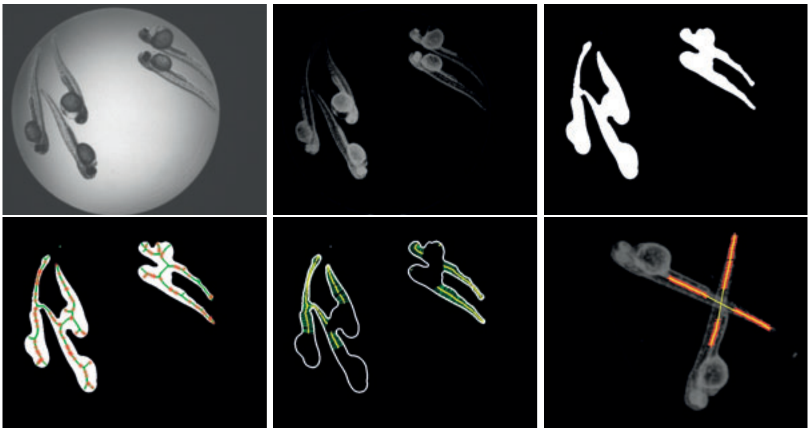


Figure 5.6. Steps for curvature extraction: (top left) an input image; (top center) after illumination correction; (top right) binary image after smoothing and thresholding; (bottom left) computed medial axes (highlighted in green) and seed-points (highlighted in red); (bottom center) refined medial axes (highlighted in yellow); (bottom right) medial axis fusion where the red lines represent tail-segments fused together to yield the complete tails (shown in yellow).

We calculated four different curvature-based features for each recovered tail, these were the tail curvature, normalized tail curvature, fan-beam width and normalized fan-beam width. These four features were then used for training classifiers to predict fish as either having straight or bent tails. Moreover, we observed substantial difference between the tail curvatures of treated and untreated fish.

5.7 Deep learning-based classification of zebrafish deformation (Paper VII)

5.7.1 Background

Zebrafish has been used as a model organism in many fields of biological research. Zebrafish embryos undergo significant shape deformation upon exposure to chemical agents, such as Camptothecin, that inhibit DNA repair. Earlier in Paper VI, we focused on quantifying and using the tail curvature of zebrafish embryos as a feature for differentiating between affected and regular fish samples in multi-fish micro-wells. The image analysis pipeline was based on established operations such as noise reduction, fish delineation, skeletonization etc.

In this study (i.e., Paper VII), we address the same problem through an alternative deep learning-based approach where a deep neural network is employed for simultaneous learning of the discriminative features as well as for classifier training. In this manner, the deep network can potentially select the most relevant features for the binary classification problem, for instance, shape of the yolk-ball or shape of the fish anterior instead of a user-specified feature such as the tail curvature.

5.7.2 Preproccesing and network training

We used data augmentation as the primary preprocessing step. The augmentation was based on rotations and horizontal inversions of the images in the original dataset, resulting in eight times more than before. We avoided affine transformations as they could potentially affect the shape of the zebrafish. We employed the well known AlexNet deep neural architecture as a classifier [32].

5.7.3 Ablation studies

A potential criticism of our approach could be that the deep architecture may have learned features from the well background rather than learning feature from the zebrafish, that is, the image foreground. To assess that the classifier has actually learnt features from the zebrafish, we performed ablation experiments where we, one after another, masked the fish in an image and then observed the change in the class probabilities obtained from the deep network, under the assumption that masking the fish should result in large variations in the resulting class probabilities. Results of the ablation experiment for one micro-well are shown in Figure 5.7 where the probability of having a micro-well with deformed fish is plotted against a micro-well where we have removed one fish at a time from a treated sample.

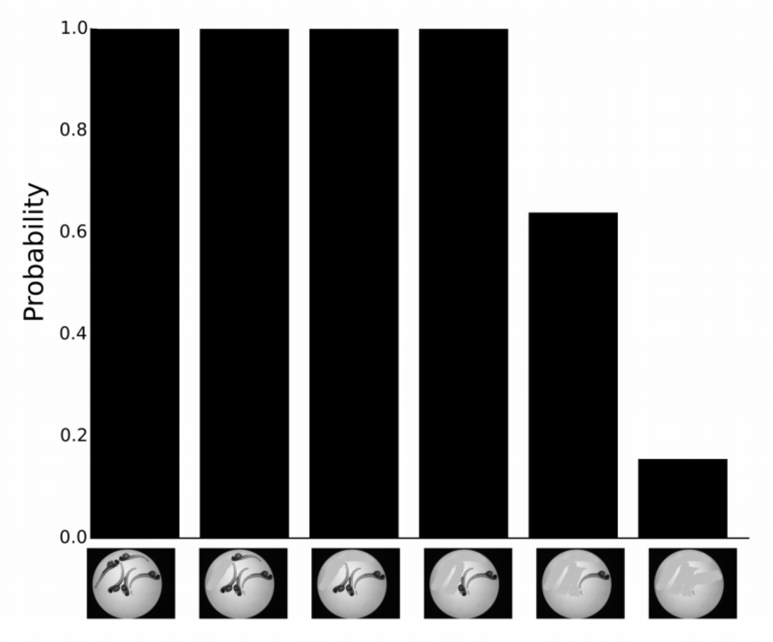


Figure 5.7. Probability of deformation as we remove one fish at a time from treated well.

6. Concluding Remarks and Future Perspectives

6.1 Concluding Remarks

This thesis has focused on both traditional image analysis methods as well as the modern deep learning methods for applications in microscopy.

From an application viewpoint, we have focused on quantifying and identifying the phenotypic deformation in microscopic zebrafish embryos in multi-fish micro-wells in Papers VI and VII. In addition, we have focused on the automated processing of fluorescently labeled subcellular structures. For this second application, we have contributed to multiple parts of the spot processing framework by focusing on spot detection in Papers II, III and IV, spot localization in Paper I and validation of the effectiveness of a compaction oligonucleotide along with a form of cluster analysis in Paper IV.

From a methods' perspective, we have performed, and thoroughly evaluated deep learning (on both zebrafish and fluorescently labeled subcellular elements) in Papers II, III and VII, against aspects such as types and amount of training data, effect of pretraining and ablation of the image foreground etc. In addition, we have focused and extended the use of compressed sensing for single molecule localization microscopy in high fluorophore density images in Paper I. We have designed and implemented traditional image analysis pipelines in Papers IV and VI. In addition, we have created and evaluated a 2AFC based tool for facilitating manual annotation of image data. Moreover, the comparison of the results from traditional image analysis techniques against those from more recent deep learning architectures has been a recurrent theme in a number of our experiments. Specific noteworthy conclusions from our experiments are discussed henceforth in this section.

When applying deep learning for spot detection in Papers II and III, we observed that the results were competitive with, and some times better than (i.e., in Paper III for reduced training data), the previous non-deep high performing methods. The results imply that similar to the developments in computer vision, the deep architectures are readily transferable to spot processing tasks in particular, and biomedical image analysis in general. Moreover, the results in Paper II, were obtained with what we refer to as 'off the shelf' networks, that is, variants of popular networks which have not been significantly modified for spot detection task. The use of the 'off the shelf' networks was motivated by the need to evaluate the effectiveness of these methods without incorporating modifications which require network modification expertise. This experimental design decision ensures that experts from the application domain (i.e., biologists) can be confident that these networks can be deployed without much tuning and still provide acceptable results. On the other hand, this restriction on modification was relaxed in Paper III and the network design modified by observing the performance on training and validation datasets (i.e., to estimate convergence and overfitting). Consequently, we observe that the network in Paper III performs relatively better than its Adaboost-based competitor method, for small sized training sets.

Our experiments on pretraining in Paper II, that is pretraining deep networks on an external dataset and subsequent finetuning on the reference dataset, did not appear to have a significant impact on the detection accuracy. However, pretraining does result in faster convergence of the network on the reference dataset. Moreover, the typical differences in the size and number of color channels of the pretraining- and the reference-datasets implies that one ends up resizing, rescaling (the external dataset) and pretraining on the external dataset itself rather than finetuning a pretrained model obtained from a model zoo. We contend that the usefulness of pretraining is a function of the tractability of the problem as well as the amount of data available for training. It seems that for our application of spot detection, the networks were tractable enough without resorting to pretraining. As part of the analysis, we also found that the design of filters/feature-detectors learnt from deep learning was markedly different from the Haar-like features which have previously been used in the traditional machine learning-based spot detectors.

We observed in Paper III that when training machine learning methods (both shallow and deep) using synthetic data, the degree of realism of the generated data makes a significant difference on the method performance, that is, the detectors performed better when trained on *SMeagol* (a tool for generating highly realistic data from carefully parameterized simulation settings[91]), as opposed to the case when training took place on an ‘off the self’ synthetic data generator even when using ‘similar’ parameters. Moreover, the ‘off the self’ synthetic datasets appear to cause significant overfitting. These results imply that many synthetic datasets may not realistically model the real data and therefore the time put into annotating a real dataset (at least for validation purposes) is well worth the effort. In this context, our development of a specialized 2AFC tool for facilitating manual annotation of real data (in Paper III), as well as its quantitative evaluation, gains more significance.

We observed that when applying deep learning to a new dataset, it is advantageous to start with relatively simple architectures such as AlexNet. For instance, in Paper VII, the eight layer AlexNet was able to rapidly converge to impressive solutions within a couple of thousand iterations. We think this pertains to a potential trade off between tractability and the ability of the network to model an intricate problem. Deeper networks such as the VGGs and ResNets may have a higher modeling potential but pose a bigger optimization challenge due to the presence of a larger set of parameters, sometimes making the problem intractable. For example, in Paper II the simple LeNet provided excellent results indicating that the choice of the network should be driven by the nature of the problem and the data.

When comparing the hand-crafted image analysis pipeline in VI (based on a user specified measure of neural degeneration, i.e., the curvature of the zebrafish tail) against the data-driven deep network in VII (based on automatically finding the most discriminative features), we observed that hand-crafted pipeline resulted in 2% higher accuracy. The results indicates that carefully designed and tuned traditional image analysis pipelines (such as the one designed in VI and comprising steps such as noise reduction, illumination correction, segmentation, skeleton generation and curvature estimation etc.) are still very competitive. However, the use of deep learning has significantly shortened the time required for generating competitive baseline results to weeks and even days. In addition, we observed in Paper VII that the most discriminative feature selected by the deep network (i.e., the yolk-ball) may be different from the one which may appear as the most discriminative to humans (i.e., the tail), and therefore it is quite possible that the best learnt feature may appear quite unintuitive to humans. On the other hand, the learnt features may guide us to new biological insights, such as morphological changes that have not previously been brought to our attention. We feel that the type of ablation studies performed in Paper VII are critical to determining whether the deep networks are learning the object of interest or are they being biased by the background.

Data-driven experiments can benefit from a prior estimate of the amount of data required for training the machine learning methods. However, to the best of our knowledge such ‘rule of the thumb’ estimates are few and far between. We have tried to experimentally address this requirement for the spot detection problem in our Papers II and III, thus coming to the rather surprising conclusion that unlike common perception, deep networks did not require large amounts of training data (although they may benefit from larger datasets). This behavior was encountered for two different real datasets used in the Papers II and III where progressive reduction in the size of the training set had marginal impact on the detection performance, in Paper III, a deep network appeared to be the most stable method against the reduction in training data. We hypothesize that this behavior is due to the deep network learning a set of discriminative features which were only common to most examples of one particular data class (i.e., positive or negative class) and allowed for separation of the two constituent classes. However, these results may not be transferable to other types of image data where the inherent complexity of the image as well as the size of the set of classification labels may necessitate provision of more data.

In general, we contend that for spot detection as well as other applications, the use of machine learning often removes the uncertain process of manual selection of appropriate thresholds as well their adaptation to different imaging conditions.

We have evaluated and extended a high fluorophore density-based single molecule localization method, called *Faster STORM*, in Paper I. Such high density methods are useful because they reduce the number of frames needed to be acquired for reconstructing a single high resolution image. We found *Faster STORM* to be very sensitive to the PSF estimate thus indicating that the method may not be suited for 3D samples where some of the fluorophores may be out of focus.

Furthermore, in paper V, we reviewed the recent developments in image-assisted spatially resolved transcriptomics. The review constituted discussion of the upstream biological experiments, the midstream spot detection techniques as well as the downstream visualization and statistical analysis tools. It was interesting to note that many recent papers still employ very basic image analysis techniques such as the use of simple thresholding techniques for identification of fluorescent signals.

6.2 Future Perspectives

Current trend in computer vision and image analysis points toward an undergoing transition from hand-crafted features and pipelines to data-driven deep learning approaches. We emphasize that this does *not* suggest that the hand-crafted solutions will be replaced by data-driven ones; rather it is more likely that we will see hybrid solutions which employ deep networks for certain sub-tasks and hand-designed traditional image analysis pipelines for other ones. Investigation of such hybrid solutions for the zebrafish classification problem discussed in Papers VI and VII would be of interest. We can already find applications in literature where the features extracted from the last few fully connected layers of the deep networks are used in conjunction with support vector machines to provide better predictions [42]. In this spirit of integration, it would be advantageous to enable current image analysis packages such as CellProfiler [95] and ImageJ [96] to export data and intermediate results in formats which can be directly processed by the contemporary deep learning libraries.

Furthermore, non-traditional data augmentation strategies such as generating scale space representations of images and then training on these representations, to assist the deep method to perform recognition at multiple scales may also prove to be useful.

The developments in the complementary field of computer vision have benefited from competitions on open datasets and corresponding leader boards to track state of the art progress. We think that biomedical image analysis will also benefit from a similar approach of having ‘living datasets’ which remain active, relevant and where the leader boards are regularly updated with new results (open datasets for biomedical image data already exist but typically lose their relevance after the competition for which they were created).

Most of these deep methods can be computationally intensive, thus making it hard to train and deploy large ensembles as well perform extensive parameter tuning. Next generation GPUs should reduce this problem and therefore it would be of interest to observe how much of a performance boost can be obtained through ensembles of deep networks.

Labeling of biomedical datasets may require support from domain experts which may be difficult to obtain due to cost and time constraints and disagreement between experts. A work around may be to use unsupervised methods such as stacked autoencoders or clustering techniques on unlabeled data and observe whether a meaningful separation of data into constituent classes is possible in this manner. It would be interesting to see this effect on both the fluorescent spot and zebrafish datasets, especially since both types of models exist at different scales of biological complexity. A study of the effect of different data augmentation strategies on biological datasets would be of interest.

For the spot detection applications, increased focus on machine learning-based methods for detection, separation and classification of clusters of overlapping spots would be practically very useful as another technique for dealing with high fluorophore density images.

Acknowledgments

This thesis is the culmination of four years of hard work and effort. I would like to take this opportunity to thank all the people who have been a fantastic source of encouragement, support and advice over these four years.

- My primary supervisor Carolina Wählby, for providing me with this wonderful opportunity to advance my education and scientific competence. I feel that every meeting with you has been a learning experience for me. Your insights, competence, patience and holistic view of science encompassing both biology and image analysis has always been a source of wonder for me. I have always appreciated the effort you put into providing rapid feedback on manuscripts.

- My co-supervisor Vladimir Ćurić, for your support, your almost instantaneous feedback on manuscripts, your availability for discussions at all times of the day and all days of the week. I appreciate the effort you put in when you, on multiple occasions, came to the university on the weekends so that you could provide much needed feedback on my work. You are a rare combination of knowledge and wisdom.

- My co-supervisors at the beginning of my degree, Alexandra Pacureanu and Ewert Bengtsson, thank you for your support, assistance and advice.

- Lena Nordström, for all the cooperation and support, and for solving problems even before they become apparent.

- Ingela Nyström, for being a very supportive head of division and for always keeping an open door for discussions.

- My colleague and friend Sajith Kecheril Sadanandan, for the long and fruitful discussions on research topics and for being a great office mate.

- Pontus Olsson, special thanks to you for your valuable advice on, and for answering my questions regarding, the thesis production process.

- My research collaborators Carl-Magnus Clausson, Johan Elf, Martin Lindén, Marco Mignardi and Xiaoyan Qian for your support and collaboration.

- Members of the informal machine learning fika group, Fredrik Wahlberg, Tomas Wilkinson and Kalyan Ram for the interesting discussions.

- I would specially like to thank my colleagues Joakim Lindblad and Petter Ranefall, for always being available for discussion on any research topic and for in general always being very supportive and helpful. It is hard to find more supportive colleagues than you two.

- Anders Hast, Ida-Maria Sintorn, Robin Strand, Gunilla Borgefors, Natasha Sladoje, Azadeh Fakhrzadeh, Fei Liu and Christophe Avenel, for being wonderful colleagues.

- Astrid Raidl and the late Olle Eriksson, for keeping the machines running.

- All Ph.D. students with whom I shared the office room 2144, aka the big room.

- All my colleagues at the VI2 division.

- My friends Salman Toor and Khurram Maqbool, for your cheerful disposition and great discussions.

- My wife Javeria, for your support and understanding, this thesis would not have been possible without you. You remind me of what is important in my life and how fortunate I truly am.

- My sons Junaid and Ibrahim, you guys are the best thing to have ever happened to me!!!

- My brother Javed and sister Amina, for being the best elder siblings one could have wished for. Both of you have been so supportive and have shouldered so many responsibilities, especially since last year. I owe you guys a lot.

- My parents Huma and Muhammad Ishaq, for your support, love and guidance at each step of my life. For encouraging me to follow this path and for prioritizing my needs above yours. All my strengths are due to you, all my weaknesses my own.

Svensk sammanfattning

Mikroskop är instrument för observation av mycket små föremål som inte kan observeras med blotta ögat. En viktig tillämpning av mikroskopi är att visuellt studera mycket små biologiska strukturer. Många biologiska strukturer är hierarkiskt organiserade, dvs atomer kombineras för att bilda molekyler, molekyler bildar subcellulära strukturer, vilka i sin tur bildar celler. Celler utgör vävnader vilka i sin tur bildar organ, och en kombination av organ bildar en organism. Biomedicinsk forskning innebär studier av både struktur och funktion av dessa hierarkiskt ordnade objekt, och med dagens nya inmärkningsmetoder och mikroskopitekniker kan vi nu visualisera enskilda molekyler. Kvantitativ och automatisk analys av resulterande data kräver digital bildanalys. Forskningsområdet är avgörande för utveckling av nya läkemedel, för att förstå fundamentala biologiska förlopp genom att redigera, aktivera eller inaktivera olika gener, hämma eller förstärka subcellulära processer etc. Framsteg inom mikroskopi har gjort det möjligt att observera allt mindre objekt. Numera är det möjligt att nå en bilduppläsning som är så hög att det går att särskilja objekt som bara ligger tjugo nanometer från varandra. Utvecklingen inom mikroskopi har åtföljts av annan teknisk utveckling som har gjort det möjligt för att genomföra ett stort antal automatiserade experiment på kort tid. Till exempel är 'high-throughput screening' en automatiserad teknik där biologiska prover exponeras för hundratals olika potentiella läkemedel, och med hjälp av digitala bilder blir det möjligt att analysera effekten på cellnivå. Dessa storskaliga experiment genererar ofta en stor mängd bilddata som inte på ett effektivt och noggrant sätt kan analyseras av en människa inom en given tidsram. Under de senaste tjugo åren har det därför utvecklats ett antal datoriserade metoder och verktyg för automatiserad analys av bilddata. Många av dessa metoder har i första hand utvecklats för bilder av icke-biologiskt ursprung, såsom fotografiska bilder av världen omkring oss. Forskningsområdet kallas datoriserad bildanalys.

I denna avhandling har vi utvecklat metoder och algoritmer för datoriserad bildanalys och tillämpat dem på två typer av biologiska data. Den första typen av biologisk data kommer från subcellulära strukturer som märkts med små fluorescerande molekyler, eller biomarkörer. Dessa fluorescerande molekyler fungerar som ljusfyrar och syns som små ljusa signaler i ett fluorescensmikroskop och gör det möjligt att avbilda, identifiera och lokalisera subcellulära strukturer, och kvantifiera förändringar som respons på behandling. Den andra typen av biologisk tillämpning som vi har arbetat med omfattar mikroskopbilder av zebrafiskembryon. Zebrafiskembryon används som modellorganismer, dvs. icke-humana arter som kan ha vissa fysiologiska egenskaper som liknar de hos människor. Dessa modellorganismer kan därför användas för grundforskning och för att testa läkemedel och terapier innan testning på försökspersoner.

I denna avhandling har vi vidareutvecklat och förfinat ett antal datoriserade bildanalysmetoder för att identifiera och lokalisera fluorescenssignaler samt vidareutvecklat traditionella bildanalystekniker för mätning av morfologiska egenskaper hos zebrafiskar. Dessutom har vi utforskat nya metoder från så kallad 'deep learning', som visat sig vara mycket framgångsrika när det gäller att lösa ett antal bildanalysproblem. I avhandlingen visar vi att det går att använda 'deep learning' för att hitta fluorescenssignaler och mäta morfologiska förändringar på zebrafiskembryon som behandlats med olika läkemedel. Vi har jämfört och utvärderat de olika metoderna och visar att 'deep learning' ofta ger lika bra eller bättre resultat tidigare använda metoder.

References

[1] I. Smal, M. Loog, W. Niessen, and E. Meijering, “Quantitative comparison of spot detection methods in fluorescence microscopy,” *IEEE Transactions on Medical Imaging*, vol. 29, no. 2, pp. 282–301, 2010.

[2] B. Rieger, R. Nieuwenhuizen, and S. Stallinga, “Image processing and analysis for single-molecule localization microscopy: Computation for nanoscale imaging,” *IEEE Signal Processing Magazine*, vol. 32, no. 1, pp. 49–57, 2015.

[3] G. J. Lieschke and P. D. Currie, “Animal models of human disease: Zebrafish swim into view,” *Nature Reviews Genetics*, vol. 8, no. 5, pp. 353–367, 2007.

[4] H. Feitsma and E. Cuppen, “Zebrafish as a cancer model,” *Molecular Cancer Research*, vol. 6, no. 5, pp. 685–694, 2008.

[5] R. J. Ober, A. Tahmasbi, S. Ram, Z. Lin, and E. S. Ward, “Quantitative aspects of single molecule microscopy.,” *IEEE Signal Processing Magazine*, vol. 32, no. 1, pp. 58–69, 2015.

[6] P. Ruusuvuori et al., “Evaluation of methods for detection of fluorescence labeled subcellular objects in microscope images,” *BMC Bioinformatics*, vol. 11, no. 1, p. 248, 2010.

[7] A. Small and S. Stahlheber, “Fluorophore localization algorithms for super-resolution microscopy,” *Nature Methods*, vol. 11, no. 3, pp. 267–279, 2014.

[8] S. Stallinga and B. Rieger, “Accuracy of the Gaussian point spread function model in 2D localization microscopy,” *Optics Express*, vol. 18, no. 24, pp. 24461–24476, 2010.

[9] M. J. Rust, M. Bates, and X. Zhuang, “Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM),” *Nature Methods*, vol. 3, no. 10, pp. 793–796, 2006.

[10] S. T. Hess, T. P. Girirajan, and M. D. Mason, “Ultra-high resolution imaging by fluorescence photoactivation localization microscopy,” *Biophysical Journal*, vol. 91, no. 11, pp. 4258–4272, 2006.

[11] S. Jiang, X. Zhou, T. Kirchhausen, and S. T. Wong, “Detection of molecular particles in live cells via machine learning,” *Cytometry Part A*, vol. 71, no. 8, pp. 563–575, 2007.

[12] A. Basset, J. Boulanger, J. Salamero, P. Bouthemy, and C. Kervrann, “Adaptive spot detection with optimal scale selection in fluorescence microscopy images,” *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4512–4527, 2015.

[13] A. Jaiswal, W. J. Godinez, R. Eils, M. J. Lehmann, and K. Rohr, “Tracking virus particles in fluorescence microscopy images using multi-scale detection and multi-frame association,” *IEEE Transactions on Image Processing*, vol. 24, no. 11, pp. 4122–4136, 2015.

[14] D. Bright and E. Steel, “Two-dimensional top hat filter for extracting spots and spheres from digital images,” *Journal of Microscopy*, vol. 146, no. 2, pp. 191–200, 1987.

[15] A. Allalou, A. Pinidiyaarachchi, and C. Wählby, “Robust signal detection in 3D fluorescence microscopy,” *Cytometry Part A*, vol. 77, no. 1, pp. 86–96, 2010.

[16] S. H. Rezatofighi, R. Hartley, and W. E. Hughes, “A new approach for spot detection in total internal reflection fluorescence microscopy,” in *Proceedings of the IEEE International Symposium on Biomedical Imaging*, pp. 860–863, IEEE, 2012.

[17] B. Zhang, M. Fadili, J.-L. Starck, and J.-C. Olivo-Marin, “Multiscale variance-stabilizing transform for mixed-Poisson-Gaussian processes and its applications in bioimaging,” in *Proceedings of the IEEE International Conference on Image Processing*, vol. 6, pp. 233–236, IEEE, 2007.

[18] J.-C. Olivo-Marin, “Extraction of spots in biological images using multiscale products,” *Pattern Recognition*, vol. 35, no. 9, pp. 1989–1996, 2002.

[19] T. R. Jones, A. E. Carpenter, M. R. Lamprecht, J. Moffat, S. J. Silver, J. K. Grenier, A. B. Castoreno, U. S. Eggert, D. E. Root, and P. Golland, “Scoring diverse cellular morphologies in image-based screens with iterative feedback and machine learning,” *Proceedings of the National Academy of Sciences*, vol. 106, no. 6, pp. 1826–1831, 2009.

[20] R. Parthasarathy, “Rapid, accurate particle tracking by calculation of radial symmetry centers,” *Nature Methods*, vol. 9, no. 7, pp. 724–726, 2012.

[21] L. Zhu, W. Zhang, D. Elnatan, and B. Huang, “Faster STORM using compressed sensing,” *Nature Methods*, vol. 9, no. 7, pp. 721–723, 2012.

[22] H. Deschout, F. C. Zanacchi, M. Mlodzianoski, A. Diaspro, J. Bewersdorf, S. T. Hess, and K. Braeckmans, “Precisely and accurately localizing single emitters in fluorescence microscopy,” *Nature methods*, vol. 11, no. 3, pp. 253–266, 2014.

[23] B. Rieger and S. Stallinga, “The lateral and axial localization uncertainty in super-resolution light microscopy,” *ChemPhysChem*, vol. 15, no. 4, pp. 664–670, 2014.

[24] F. Huang, S. L. Schwartz, J. M. Byars, and K. A. Lidke, “Simultaneous multiple-emitter fitting for single molecule super-resolution imaging,” *Biomedical Optics Express*, vol. 2, no. 5, pp. 1377–1393, 2011.

[25] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005.

[26] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain.,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[27] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, p. 9, 1986.

[28] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation applied to handwritten zip code recognition,” *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[29] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[30] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015. Published online 2014; based on TR arXiv:1404.7828 [cs.NE].

[31] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data

with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[33] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[34] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[35] A. Coates and A. Y. Ng, "Learning feature representations with k-means," in *Neural networks: Tricks of the trade*, pp. 561–580, Springer, 2012.

[36] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[37] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[38] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proceedings of the 13th European Conference on Computer Vision*, pp. 818–833, Springer, 2014.

[39] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, "Understanding neural networks through deep visualization," *arXiv preprint arXiv:1506.06579*, 2015.

[40] L. M. Zintgraf, T. S. Cohen, and M. Welling, "A new method to visualize deep neural networks," *arXiv preprint arXiv:1603.02518*, 2016.

[41] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 315–323, 2011.

[42] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, 2016.

[43] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[44] A. Karpathy, "Commonly used activation functions." `http://cs231n.github.io/neural-networks-1/#actfun`. Accessed: 2016-03-20.

[45] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[46] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[47] A. Karpathy, "Loss function." `http://cs231n.github.io/linear-classify/#loss`. Accessed: 2016-03-20.

[48] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng, "On optimization methods for deep learning," in *Proceedings of the 28th*

International Conference on Machine Learning, pp. 265–272, 2011.

[49] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[50] Y. Nesterov, “A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$,”

[51] M. D. Zeiler, “Adadelta: An adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.

[52] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[53] M. Lin, Q. Chen, and S. Yan, “Network in network,” *arXiv preprint arXiv:1312.4400*, 2013.

[54] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

[55] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, “Return of the devil in the details: Delving deep into convolutional nets,” in *British Machine Vision Conference*, 2014.

[56] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, “Deep image: Scaling up image recognition,” *arXiv preprint arXiv:1501.02876*, vol. 22, p. 388, 2015.

[57] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, “Why does unsupervised pre-training help deep learning?,” *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.

[58] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in *Advances in Neural Information Processing Systems*, pp. 2951–2959, 2012.

[59] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[60] C. Szegedy, A. Toshev, and D. Erhan, “Deep neural networks for object detection,” in *Advances in Neural Information Processing Systems*, pp. 2553–2561, 2013.

[61] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Proceedings of the European Conference on Computer Vision*, pp. 184–199, Springer, 2014.

[62] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, 2015.

[63] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[64] A. Meyers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. P. Murphy, “Im2calories: Towards an automated mobile vision food diary,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1233–1241, 2015.

[65] T. Pfister, J. Charles, and A. Zisserman, “Flowing convnets for human pose estimation in videos,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1913–1921, 2015.

[66] J. Walker, A. Gupta, and M. Hebert, “Dense optical flow prediction from a static image,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2443–2451, 2015.

[67] L. Wang, W. Ouyang, X. Wang, and H. Lu, “Visual tracking with fully convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3119–3127, 2015.

[68] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[69] E. J. Candès, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[70] D. L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[71] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[72] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[73] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. E. Kelly, R. G. Baraniuk, *et al.*, “Single-pixel imaging via compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[74] J. Romberg, “Imaging via compressive sampling [introduction to compressive sampling and recovery via convex programming],” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 14–20, 2008.

[75] R. Robucci, J. D. Gray, L. K. Chiu, J. Romberg, and P. Hasler, “Compressive sensing on a CMOS separable-transform image sensor,” *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1089–1101, 2010.

[76] G. Kutyniok, “Theory and applications of compressed sensing,” *GAMM-Mitteilungen*, vol. 36, no. 1, pp. 79–101, 2013.

[77] E. J. Candès and T. Tao, “Decoding by linear programming,” *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[78] E. Candès and J. Romberg, “Sparsity and incoherence in compressive sampling,” *Inverse Problems*, vol. 23, no. 3, pp. 969–985, 2007.

[79] M. Grant, S. Boyd, and Y. Ye, “CVX: Matlab software for disciplined convex programming,” 2008.

[80] E. Candès and J. Romberg, “ℓ_1-MAGIC : Recovery of sparse signals via convex programming.” `URL:www.acm.caltech.edu/l1magic/downloads/l1magic.pdf`, 2005.

[81] T. Blumensath and M. E. Davies, “Iterative hard thresholding for compressed sensing,” *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[82] I. Carron, “Sparse signal recovery solvers.”

https://sites.google.com/site/igorcarron2/cs#reconstruction. Accessed: 2016-03-20.

[83] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly, "Compressed sensing MRI," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 72–82, 2008.

[84] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[85] K. Choi, J. Wang, L. Zhu, T.-S. Suh, S. Boyd, and L. Xing, "Compressed sensing based cone-beam computed tomography reconstruction with a first-order method," *Medical Physics*, vol. 37, no. 9, pp. 5113–5125, 2010.

[86] K. R. Chi, "Microscopy: Ever-increasing resolution," *Nature*, vol. 462, no. 7273, pp. 675–678, 2009.

[87] L. Schermelleh, R. Heintzmann, and H. Leonhardt, "A guide to super-resolution fluorescence microscopy," *The Journal of Cell Biology*, vol. 190, no. 2, pp. 165–175, 2010.

[88] S. W. Hell and J. Wichmann, "Breaking the diffraction resolution limit by stimulated emission: Stimulated-emission-depletion fluorescence microscopy," *Optics Letters*, vol. 19, no. 11, pp. 780–782, 1994.

[89] T. Blumensath and M. E. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298–309, 2010.

[90] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," 2009.

[91] M. Lindén, V. Ćurić, A. Boucharin, D. Fange, and J. Elf, "Simulated single molecule microscopy with smeagol," *Bioinformatics*, 2016.

[92] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[93] P. B. Stetson, "Daophot: A computer program for crowded-field stellar photometry," *Publications of the Astronomical Society of the Pacific*, pp. 191–222, 1987.

[94] C. Wählby, L. Kamentsky, Z. H. Liu, T. Riklin-Raviv, A. L. Conery, E. J. O'Rourke, K. L. Sokolnicki, O. Visvikis, V. Ljosa, J. E. Irazoqui, *et al.*, "An image analysis toolbox for high-throughput C. elegans assays," *Nature Methods*, vol. 9, no. 7, pp. 714–716, 2012.

[95] A. E. Carpenter, T. R. Jones, M. R. Lamprecht, C. Clarke, I. H. Kang, O. Friman, D. A. Guertin, J. H. Chang, R. A. Lindquist, J. Moffat, *et al.*, "Cellprofiler: Image analysis software for identifying and quantifying cell phenotypes," *Genome Biology*, vol. 7, no. 10, p. R100, 2006.

[96] C. A. Schneider, W. S. Rasband, K. W. Eliceiri, *et al.*, "NIH Image to ImageJ: 25 years of image analysis," *Nature Methods*, vol. 9, no. 7, pp. 671–675, 2012.

Distribution: publications.uu.se
urn:nbn:se:uu:diva-283846

ACTA
UNIVERSITATIS
UPSALIENSIS
UPPSALA
2016