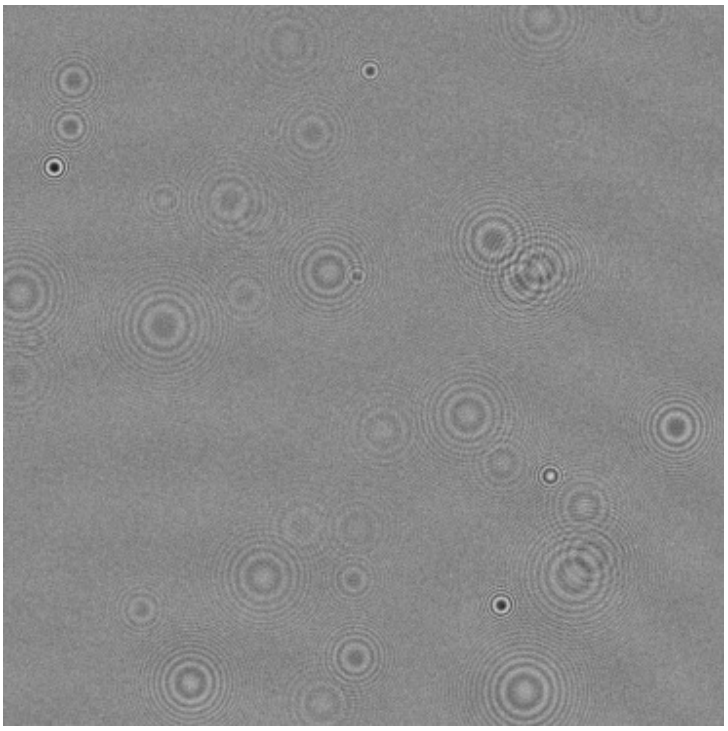


Karakterisering och spårning av nanopartiklar med djupinlärning

Användning av djupinlärnings-modellen YOLO för att karakterisera och bestämma positioner för nanopartiklar

Kandidatarbete för fysikinstitutionen

Arash Darakhsh Edvin Johansson Simon Nilsson
Sanna Persson Rickard Ström

Fysikinstitutionen

CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2021

Kandidatarbete 2021

Charaterization and Tracking of Nanoparticles with Deep Learning

Applying the deep learning model YOLO to characterize and detect positions for nanoparticles

Karakterisering och spårning av nanopartiklar med djupinlärning

Användning av djupinlärnings-modellen YOLO för att karakterisera och bestämma positioner för nanopartiklar

Arash Darakhsh
Edvin Johansson
Simon Nilsson
Sanna Persson
Rickard Ström

CHALMERS

Fysikinstitutionen
Chalmers tekniska högskola
Göteborg, Sverige 2021

Karakterisering och spårning av nanopartiklar med djupinlärning
Användning av djupinlärnings-modellen YOLO för att karakterisera och bestämma positioner för nanopartiklar

Handledare: Benjamin Midtvedt, Daniel Midtvedt
Examinator: Lena Falk

Kandidatarbete 2021
Fysikinstitutionen
Chalmers tekniska högskola

Omslagsbild: Datorsimulerad digital holografisk mikroskopibild av miepartiklar utifrån off-axis konfiguration framställd med pythonbiblioteket *Deeptrack 2.0*.

Abstract

Our work utilizes the deep learning model YOLOv3 to develop an open source particle tracking network. The model is significantly better than traditional algorithmic methods at particle tracking but is slightly worse at the task of characterization compared to traditional algorithms. The model has also proven to be similar in performance to previous deep learning methods. These results are obtained without the need for the user to micromanage the process. The network also has further room for improvements by iteratively training the model on larger datasets making future development possible. The model was trained on simulated holographic microscopic images generated with the Python library DeepTrack. The library was used to simulate images from an off axis configuration. Parameters for the particles' x-, y-, z-positions, refractive index, radius and number of particles per image were ascribed random values in appropriate intervals to simulate an experimental setup. After training on simulated data the YOLOv3 network was tested on both simulated and experimental images. The tests on simulated data consisted of 5000 images with 448x448 pixels with 0-10 particles per image. A recall of 0.96 and precision of 0.87 were obtained yielding an F1-score of 0.91. The model was further compared to traditional algorithms for particle tracking. The clear advantages of deep learning are motivated by better handling of multiple particles, overlapping particles and different noise conditions. Our model also shows the potential of estimating the refractive index and radius of particles with an end-to-end method. Our code and model weights are available at `https://github.com/Deep-learning-for-particle-tracking/YOLOv3-for-Particle-Tracking`.

Sammandrag

Föreliggande arbete använder maskininlärningsmetoden YOLOv3 för att utveckla en metod för spårning och karakterisering av partiklar som görs tillgängligt för allmänheten. Modellen presterar bättre än traditionella algoritmiska metoder för partikeligenkänning och lokalisering av partiklar men något sämre för karakterisering och bestämmande av partiklars egenskaper jämfört med traditionella algoritmer. Modellen har dessutom visat resultat i linje med tidigare djupinlärningsmetoder. Dessa resultat ges också utan samma krav på användaren för detaljstyrning av processen. Modellen har vidare möjlighet att förbättras över tid genom att iterativt tränas på större mängder data vilket ger möjligheter för framtida utveckling. Träningen av modellen utfördes med simulerade holografiska mikroskopbilder som genererades med python-biblioteket DeepTrack. Från biblioteket användes funktioner för att simulera bilder från en off-axis-konfiguration. Parametrar för partiklars x-, y-, z-positioner, brytningsindex, radie samt antal partiklar per bild tillskrevs slumpmässiga värden i lämpliga intervall representativa för en experimentuppställning. Modellen testades efter träningen på både simulerade och experimentella bilder. Tester på den simulerade datan bestod av 5000 bilder med 448x448 pixlar med 0-10 partiklar i varje bild. Modellens resultat visar på en recall på 0,96 samt en precision på 0,87 vilket ger ett F1-score på 0,91. Modellen jämfördes även med traditionella algoritmer för partikelspårning. Tydliga fördelar med djupinlärning motiveras genom bättre hantering av flera partiklar, överlappande partiklar samt olika brusförhållanden. Vår modell visar också på möjligheten att förutsäga partiklars brytningsindex och radie med en end-to-end-metod. Kod och modellvikter finns tillgängliga här: `https://github.com/Deep-learning-for-particle-tracking/YOLOv3-for-Particle-Tracking`.

Förord

Ett stort tack till både Daniel och Benjamin för er vägledning under arbetets gång. Vi är väldigt tacksamma för er hjälp med de frågor vi haft och ert kontinuerliga stöd under hela projektet.

Vi vill även tacka utvecklarna av DeepTrack för ett väldigt användbart simuleringsverktyg som underlättat vår arbetsprocess.

Arash.D, Edvin.J, Simon.N, Sanna.P, Rickard.S

Innehåll

1

Introduktion

Under 1948 undersökte Dennis Gabor lösningsmetoder för att förbättra upplösningen av elektronmikroskopi, men istället fann han en alternativ experimentell metod vilket ledde till grunden för ett ämnesområde som nu är känt som holografi [1]. Huvudprincipen bakom holografi bygger på interferens av ljusvågor där interferensmönstret som erhålls vid holografimätningar innehåller information om mätobjektets form och tjocklek [2]. Holografimätningar utförs genom att stråla ett objekt med ljus (nu för tiden oftast laserljus) som ger upphov till en diffraktionsstråle som sedan kombineras med en referensstråle på en detektor där den skapade bilden kallas ett hologram. De tidigaste analoga hologrammen använde fotoelektriska filmer som detektorer vilket erbjöd hög upplösning men var väldigt opraktiska att använda [2]. Problemet med fotoelektriska filmer är att de kräver mörka rum vid mätningar, lång exponeringstid, kemisk behandling av hologrammet samt att de måste bytas ut manuellt för varje ny mätning.

Betydande framsteg gjordes inom holografi med skapandet av lasern och utvecklingen av off-axis konfigurationen. Lasern medförde en koherent ljuskälla vilket gav upphov till stabila interferensmönster och off-axis konfigurationen [3]–[5], där objektstrålen och referensstrålen infaller mot detektorn med en viss vinkel, underlättade filtreringen av oönskade interferenser.

Digital holografi (DH) introducerades först under 1994 då Schnars och Jueptner använde en CCD-kamera (Charge Coupled Device) som detektor [6]. Fördelen med DH är att mätningarna är betydligt mer praktiska, snabbare att utföra och utifrån hologrammet kan sedan en komplex numerisk matris räknas fram vilket tillåter separering av amplituden och fasen av vågfronten. Fasen innehåller information om mätobjektets optiska tjocklek. Det finns flera olika experimentella konfigurationer av DH, exempelvis inline, off-axis och phase shift. Om dessa konfigurationer används inom applikationen av mikroskopi [7] kallas det för digital holografisk mikroskopi (DHM).

Exempel på tillämpningsområden av DHM är inom tomografi, biomedicin och mätning av partiklars rörelse i medium. Tomografi används för att avbilda objekt i skikt, mest känd från datortomografi (*eng.* computer tomography) som används för att avbilda den mänskliga hjärnan [8]. DHMs applikationer inom biomedicin används för att avbilda cancerceller [9], blodceller [10], vävnader och levande celler [11],[12]. Studier av partiklars rörelse i medium kan användas för att bestämma deras radie, position och brytningsindex [13]. Med information om dess radie och brytningsindex går det att identifiera partikeltypen.

För att identifiera partiklar finns flera regelbaserade algoritmer för partikeldetektion som kan erhålla hög noggrannhet i positionsbestämning och karakterisering. Dessa är dock

begränsade till låga brusmiljöer och låga partikeldensiteter eftersom problem vanligtvis uppstår vid överlappande partiklar. Dessutom blir dessa algoritmer väldigt tidskrävande när antal partiklar i bilden ökar vilket förhindrar en realtids-implementation där detektionen körs mellan varje bildtagning. Istället är identifieringen indelad i två delprocesser: bildtagning med DHM och sedan analys av datan med detektionsalgoritmen.

Applikationen av maskininlärning för fysikaliska modeller har under de senaste åren ökat i användning med framväxten av djupinlärning (*eng.* deep learning). Djupinlärning är ett område inom maskininlärning där neurala nätverk iterativt tränar sig själva (inlärning) utifrån data. Beteckningen 'djup' syftar på ett neuralt nätverk som består av flera sammanbundna lager. Problemområden som självkörande fordon [14], mönsterigenkänning av handskrivna symboler [15], röstigenkänning [16] samt objektdetektering i bilder [17]–[19] är exempel på problem som kan lösas och optimeras med djupinlärningsmodeller.

Tidiga detektionsmetoder inom djupinlärning, som exempelvis R-CNN [20] delar in bilden i mindre delar kallade regionsförslag (*eng.* region proposal) och nätverket förutsäger sedan klass och en detektionslåda (*eng.* bounding box) utifrån dessa regionsförslag. Med denna metod erhålls en hög noggrannhet, men nackdelen med R-CNN är dess komplexa nätverksstruktur vilket medför lång körningstid för modellen och en mer komplex träningsprocess.

Komplexitetsproblemet med R-CNN löses med en metod som gör hela detektionen i ett steg. Redmon et al. [21] publicerade under 2015 deras första version av en objekdetektionsmodell där det neurala nätverket genomför identifieringen av detektionslådor och klassificeringen av objekten i ett steg. Metoden skannar bilden endast en gång varför namnet *You only look once* (YOLO).

1.1 Syfte

Syftet med föreliggande arbete är att med hjälp av maskininlärningsmetoden YOLO kunna utveckla ett helt automatiserat djupinlärningsnätverk som sedan ska göras tillgängligt för allmänheten (*eng.* open source). Med tanke på den uppsjö tillämpningsområden som finns för partikeligenkänning, såsom studier av rörelsen för molekyler i vätska och proteiner i levande celler, är det viktigt att erhålla de bästa möjliga mätmetoder tillgängliga. De absolut vanligaste metoderna idag bygger på traditionella algoritmiska lösningar för partikeldetektering som funnits i över 20 år. Dessa algoritmiska detekteringsmetoder kräver att användaren bestämmer de parametrar som ska följas för varje ny bildbehandling av partiklar. Vidare problem uppstår när signal-brusförhållandena är dåliga och vid bristande belysning. Användaren måste därmed lägga mer tid på att manuellt justera algoritmen för att erhålla acceptabla resultat. Syftet med maskininlärning för partikeligenkänning är att kunna erhålla samma kvalitet av resultat utan att användaren behöver detaljstyra processen. Vidare kan djupinlärningsmetoder ha möjlighet att bearbeta stora mängder data för att iterativt förbättras över tid och även under dåliga förhållanden uppnå goda resultat.

1.2 Avgränsningar

Modellen som utvecklades är som nämnt tidigare baserad på YOLO-modellen. Detta gör att ingen egen modell behövdes utvecklas, utan YOLOv3-modellen [22] anpassades istället för att kunna tillämpas på spårning och karakterisering av nanopartiklar. Första målet med projektet var att anpassa YOLO-modellen till att fungera på simulerad data för holografisk

mikroskopi. Till en början bestämde modellen x-, y- och z-position och utvecklades sedan, när modellen fungerade väl, för att även bestämma radie och brytningsindex. Att förbättra modellens noggrannhet samt att träna modellen för att kunna användas på experimentell data under olika experimentella förhållanden var av mindre prioritet i projektet. Det främsta syftet var istället att utveckla en modell som är en prototyp för framtida utveckling inom området inriktat mot specifika tillämpningar.

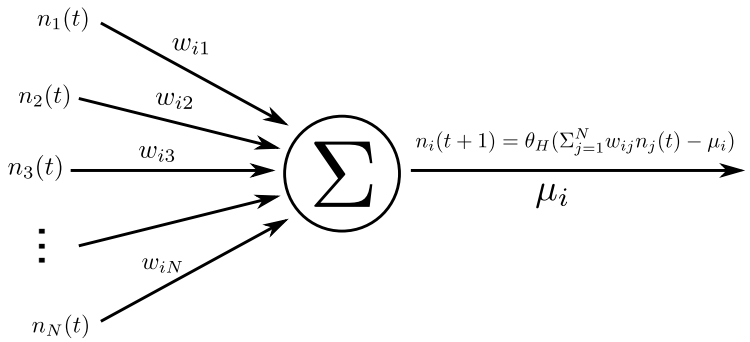
2

Bakgrund

2.1 Neurala nätverk och detektion

B. Mehlig beskriver i 'Machine Learning with Neural Networks' [23] digitala neurala nätverk som i grova drag inspirerade av nätverk av nervceller i däggdjurets hjärna. Jämfört med dess biologiska motsvarighet har dock de digitala neurala nätverken en enklare algoritm och är lättare att förstå. Emellertid är grundprinciperna mycket lika de för nervceller och igenkänning av komplexa strukturer och mönster kan generaliseras på ett sätt som aldrig skulle kunna uppnås med traditionella algoritmer.

McCulloch-Pitts-neuronen beskrivs enligt Mehlig som den digitala motsvarigheten till nervcellen och verkar genom att ta in N st värden (n_1, n_2, $n_3 \dots$) beroende av N st vikter (w_{i1}, w_{i2}, $w_{i3} \dots$) för att därefter ge ett utvärde som i det enklaste fallet svarar mot antingen noll eller ett. I bilden nedan visas en sådan neuron vars utvärde summeras över samtliga N värden multiplicerade med de N vikterna och en så kallad bias subtraheras. Därpå beskrivs vidare en så kallad aktiveringsfunktion som i det här fallet är en stegfunktion för att tvinga värdet till ett alternativt noll.



Figur 2.1: En McCulloch-Pitts-neuron som tar in N st värden beroende av N st vikter och ger ett utvärde som i det här svarar mot antingen noll eller ett. Egen bild inspirerad av [23].

Ett lager i ett neuralt nätverk kan sedan konstrueras av flera McCulloch-Pittsneuroner som beskrivet i Machine Learning with Neural Networks. I en vanligt typ av nätverk används exempelvis så kallade fullständigt sammankopplade lager (*eng.* fully connected layer) där inga kopplingar finns mellan neuronerna i samma lager men samtliga neuroner har kopplingar till samtliga neuroner i det föregående (det vänstra) lagret. Kopplingarna

består likt innan av vikter och bias som kan variera för varje koppling beskriver författaren.

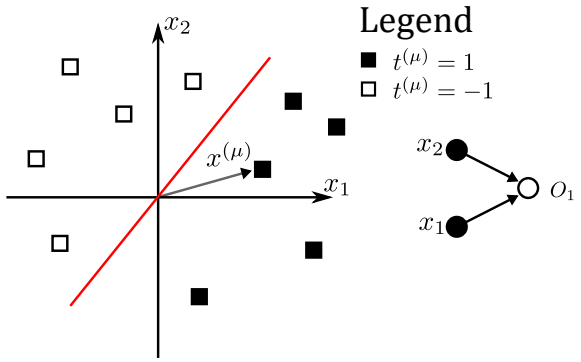
En serie av tre eller flera lager kallas ofta för djupinlärningsnätverk. Det/de lager som då ligger mellan lagret för invärden (första) och lagret för utvärden (sista) kallas dolda lager [23].

Beräkningen för ett grundläggande exempel presenteras i Machine Learning with Neural Networks med två invärden och ett utvärde som fås av

$$O = sgn(\overrightarrow{w} \cdot \overrightarrow{x} - \theta) \tag{2.1}$$

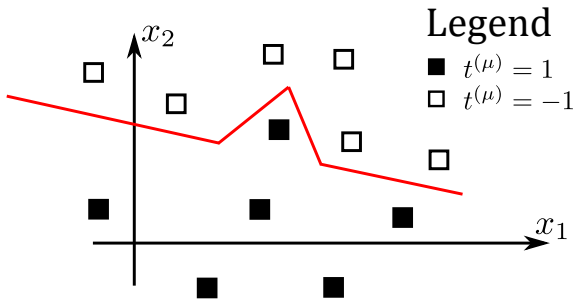
där $\overrightarrow{w}$ är vektorn för vikterna, $\overrightarrow{x}$ är indatan och θ är neuronens bias.

Nätverket ovan är tillräckligt för klassificeringen nedan eftersom datan är linjärt separerbar [23].



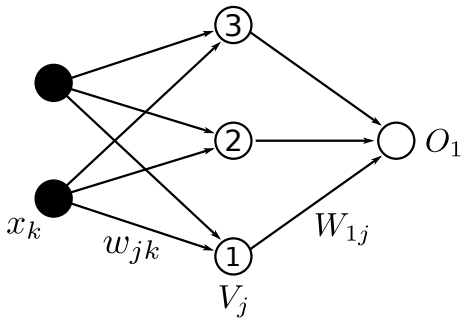
Figur 2.2: Ett exempel på ett nätverk med två invärden och ett utvärde där datan som visas till vänster representeras av rutor och är linjärt separerbar. Egen bild inspirerad av [23].

Om en datamängd emellertid inte är separerbar likt figur 2.3 nedan kan fler lager införas mellan invärden och utvärden menar Mehlig. Vidare beskrivs att två vektorer för vikterna då måste införas mellan samtliga lager och de kan i allmänhet skrivas som en M x N-matris för N neuroner i det vänstra lagret och M neuroner i det högra.



Figur 2.3: Ett exempel på en datamängd för ett nätverk som representeras av rutor och som inte är linjärt separerbar. Egen bild inspirerad av [23].

Ett exempel på ett sådant nätverk beskrivs vidare kunna se ut som figur 2.4.



Figur 2.4: Ett exempel på ett nätverk med två invärden, ett dolt lager med tre neuroner och ett utvärde. Egen bild inspirerad av [23].

För de tidiga nätverken definierades enligt Mehlig den så kallade kostnadsfunktionen (*eng.* loss function), MSE (Mean squared error), enligt dubbelsumman

$$H = \frac{1}{2}\sum_{\mu,n}(t_i^{(\mu)} - O_i^{(\mu)})^2, \tag{2.2}$$

där t_i är det i:te elementet i vektorn för rätt svar som nätverket tränas på och på den μ:te vektorn i dataserien och O_i är nätverkets i:te förutsägelse. Numera används enligt Mehlig flera andra olika typer av kostnadsfunktioner för olika användningsområden. Eftersom kostnadsfunktionen ovan bör minimeras enligt författaren för att minimera skillnaden mellan t_i och O_i bör uppdateringen av vikterna mellan två lager i exemplet ske enligt

$$\delta w'_{m,n} = -\alpha\frac{\partial H}{\partial w_{m,n}}, \tag{2.3}$$

där α är nätverkets learning rate vilket är en liten konstant. Det ger ett litet steg i den negativa riktningen av gradienten vilket vidare beskrivs motsvara en stegvis rörelse i riktningen för minskad kostnadsfunktion [23].

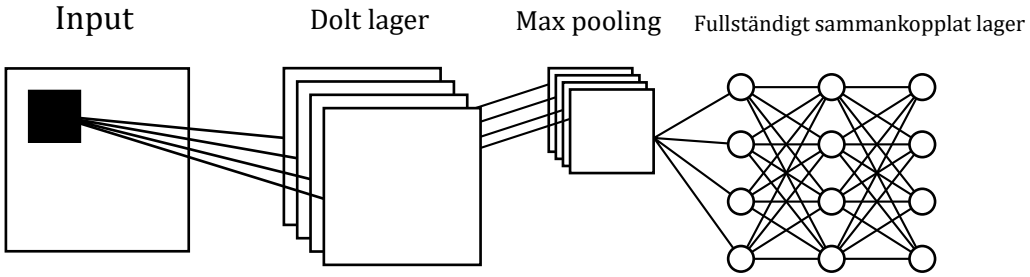
Att använda fler än ett dolt lager (*eng.* hidden layer) är inte nödvändigt men har ofta visat sig förbättra prestationen på nätverk för olika typer av uppgifter [23].

Ett faltningsnätverk (*eng.* Convolutional Neural Network (CNN)) har enligt Mehlig på senare tid visat sig effektivt vid bildigenkänning. En viktig anledning som lyfts upp till nätverkets framgång är att de kan innehålla färre neuroner. Enligt författaren är fördelen med detta dels att prestandan ökar för mindre krav på process-arbete och dels att det regulariserar nätverket vilket minskar risken för överanpassning.

Mehlig beskriver vidare att ett CNN kan ha flera olika hierarkier av lager, så kallade faltningslager. Tanken bakom arkitekturen är att ytterligare lager kan lära sig mer abstrakta egenskaper. Exempelvis kan ett lager vara skapat för att detektera en bilds lokala egenskaper, såsom kanter och enklare former. Dessa enklare egenskaper kallas ofta från engelskan för features [23].

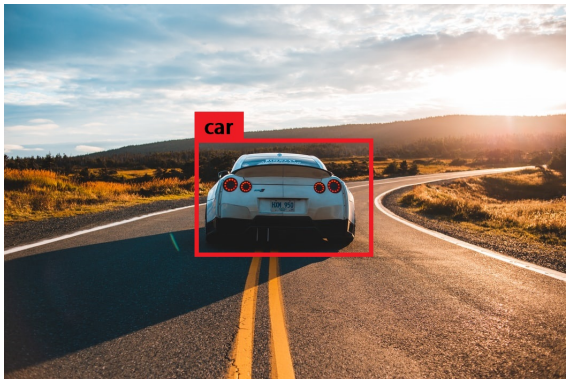
Utöver dess features innehåller, som fortsatt beskrivet av Mehlig, ofta faltningsnätverk andra sorters lager. Författaren menar vidare att det ofta förekommande max pooling-

lagret används för att förenkla outputs från de dolda lagren (*eng.* Hidden/Convolutional Layer(s)) . Efter max pooling-lagret beskrivs också att flera fullständigt sammankopplade lager kan förekomma. Ett exempel på ett faltningsnätverk med input-lager, dolda lager, max pooling-lager följt av det fullständigt sammankopplade lagret visas i figur 2.5 nedan.



Figur 2.5: Ett exempel på ett faltningsnätverk med input-lager, dolda lager, max pooling-lager följt av det fullständigt sammankopplade lagret. Egen bild inspirerad av [23].

Djupinlärning med faltningsnätverk har enligt Mehlig ökat i popularitet inte minst för bildigenkänning och objektdetektering. I figur 2.6 nedan visas en bildruta från en videoinspelning av en bil. Ett faltningsnätverk som tränats på allmänt tillgängliga Pascal VOC-träningsdatan detekterar genast bilen och ger en så kallad detektionslåda (*eng.* bounding box) för objektet. Därefter klassificeras även det detekterade objektet till att vara en bil [23].

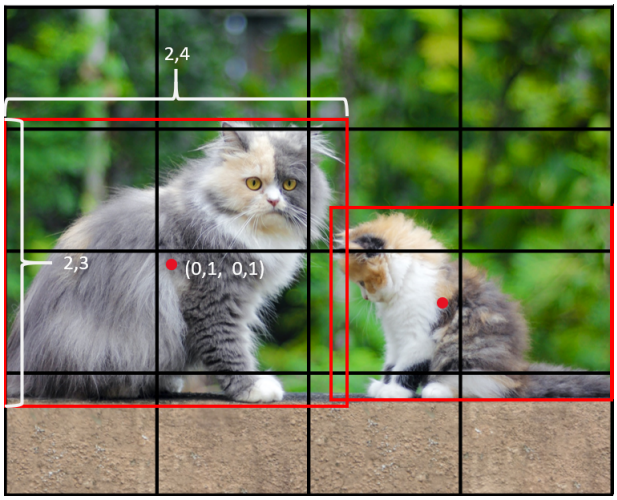


Figur 2.6: En bildruta tagen från en videoinspelning av en bil där ett faltningsnätverk som tränats på Pascal VOC-träningsdatan detekterar bilen och tillskriver den detektionslådor. Direkt efter klassificeras objektet till att vara en bil. Foto av Erik Mclean från Unsplash.

2.2 Arkitektur YOLO

Vid publiceringen av den första YOLO-modellen [21] presenterade författarna ett nytt angreppssätt som kunde utföra objektdetektering på hela bilden parallellt vilket radikalt minskade tiden för inferens (modellförutsägelser) jämfört med tidigare modeller som exempelvis [24]. De modellerade objektdetektering som ett regressionsproblem med målet att minimera kvadratiska medelfelet (*eng.* mean squared error, härefter används MSE) i lokaliseringskoordinaterna och klassbestämningen av objekten i bilderna [21]. Vidare presenterade Redmon et al. en metod där de delade in bilden i ett rutnät och lät varje ruta vara ansvarig för att bestämma position och klass för de objekt som har mittpunkt i rutan.

Det innebar att enbart en ruta var ansvarig för varje objekt och att alla objekt i bilden kunde bestämmas parallellt [21]. I figur 2.7 visas hur bilden delas in i ett rutnät och att varje ruta detekterar objekt som har mittpunkt i den, samt ger sitt output relativt till rutans storlek. Enligt konvention inom objektdetektering är origo i det övre vänstra hörnet. Mittpunkten ges relativt till origo och detektionslådans bredd och höjd ges i enheten antal rutor.



Figur 2.7: Visualisering av objektdetektering med YOLO algoritmen. De röda punkterna indikerar vilken ruta som är ansvarig för lokaliseringen och lådorna är de optimala detektionskoordinaterna relativt till rutstorleken. Foto av Nihal Karkala från Unsplash.

Den första YOLO-modellen [21] introducerade också ett antal problem. Framförallt hade modellen svårt att detektera mindre objekt och enbart ett objekt kunde detekteras i varje ruta [22]. Det innebar att i fallet då en person står mitt framför en bil skulle endast ett av objekten kunna detekteras [25]. De uppföljande publiceringarna av Redmon et al. försöker åtgärda dessa problem i YOLOv2 [25] och YOLOv3 [22] genom att introducera modifieringar till den ursprungliga arkitekturen. Detta arbete inriktar sig på en tillämpning av YOLOv3 och kommer därför enbart att beskriva detaljerna i den.

I YOLOv3 används först ett faltningsnätverk, Darknet-53, för att utvinna egenskaper (*eng.* extract features) i bilden. Detta nätverk tränades först för bildklassificering på det kända ImageNet [26] och för att utföra detektion tränades sedan också 53 ytterligare faltningslager [22]. För att förbättra detektionen av mindre objekt i bilderna utfördes detekteringen på tre olika rutnätsskalor där ett finare rutnät ska hjälpa modellen att detektera mindre objekt i bilderna [22].

Den andra begränsningen av den ursprungliga YOLO-modellen avseende överlappande objekt förbättrades genom introduktionen av ankare (*eng.* anchor boxes). Varje cell som bilden har, se figur 2.7, associeras med flera ankare vilka ger modellen en utgångspunkt för möjliga detektionslådor och nätverket ger sedan ett värde för hur mycket en viss ankare borde modifieras för att innesluta ett objekt [25]. Varje ankare definieras av en bredd och en höjd som bestäms med en metod som kallas K-means-clustering [22]. YOLOv3-modellen [22] förutsäger sedan en offset till varje ankare istället för att förutsäga objektets bredd och höjd direkt vilket medför att samtliga förutsägelser utförs på en liknande skala oavsett objektets storlek. Varje ankare ska representera en delmängd av datan som mo-

dellen tränas på. Exempelvis kan en vertikalt stående rektangel vara en lämplig ankare som representerar människor och stolpar [25]. I YOLOv3 ges output på tre olika skalor med olika rutnätsstorlekar och de använder totalt nio ankare. Varje skala tilldelas tre ankare [22]. För att vägleda modellen använder den finaste skalan de minsta ankarna och det grövsta rutnätet de tre största ankarna [22]. Modellens output konverteras sedan till detektionslådor med sambandet

$$\begin{aligned} b_x &= \sigma\left(t_x\right) + c_x \\ b_y &= \sigma\left(t_y\right) + c_y \\ b_w &= p_w e^{t_w} \\ b_h &= p_h e^{t_h} \end{aligned} \tag{2.4}$$

där t_* är modellens output, c_x och c_y är vilken cell som objektet har mittpunkt i, p_w och p_h är ankarens bredd och höjd samt

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.5}$$

De slutgiltiga detektionskoordinaterna ges av $\{b_x, b_y, b_w, b_h\}$ där b_x samt b_y är objektets mittpunkt, b_w är objektets bredd och b_h är objektets höjd [22]. Förutom detektionskoordinater bestämmer varje modellen även objektets klass samt sannolikheten för att det finns ett objekt på positionen. Det sanna värdet för sannolikheten för ett objekt är noll om en viss position och ankare i bilden inte tilldelats ett objekt, alltså inget objekt har mittpunkt i rutan, och annars ges den av

$$\frac{\hat{b} \cap b}{\hat{b} \cup b} \tag{2.6}$$

vilket är snittet över unionen (*eng.* intersection over union (IOU)) mellan den korrekta detektionslådan, $\hat{b}$ och modellens konverterade output, b enligt (2.4) [25].

Modellen tränades för att minimera en kostnadsfunktion som beräknar felet i detektionskoordinater, klassbestämning och sannolikheten för ett objekt [21]. Vid publiceringen av YOLOv3 uppvisade de resultat i linje med de bästa modellerna inom objektdetektering på MS COCO [27] samt ett signifikant högre FPS-värde vid inferens.

2.3 Mått på nätverkets noggrannhet

Det finns flera olika sätt att mäta ett nätverks noggrannhet. Ett sätt är att betrakta modellens förutsägelse och bestämma om det faktiskt finns ett objekt där. Om snittet över unionen överstiger en viss gräns (exempelvis 50 %) mellan den förutsagda detektionslådan och objektets sanna detektionslåda görs också en förutsägelse om vilken klass objektet tillhör. Om förutsägelsen är korrekt kallas den True Positive (TP). Om förutsägelsen inte motsvarar ett objekt betecknas den False Positive (FP). På samma sätt definieras True Negative (TN) om nätverket inte gör en felaktig förutsägelse samt False Negative (FN) om nätverket inte gör en förutsägelse men borde gjort det.

Ett sätt att sedan mäta nätverkets noggrannhet är att betrakta hur stor andel objekt som var korrekt förutsagda. Detta måttet kallas precision och definieras genom

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.7}$$

som är ett bra mått på själva förutsägelsen, men inte om man vill veta om nätverket förutsäger samtliga objekt i bilden. Ett mått på detta är

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.8}$$

som ger en uppskattning på hur stor andel av de sanna objekten som nätverket förutsäger. Det finns även ett väletablerat mått som kombinerar dessa två, nämligen

$$F_1 = 2\,\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{2.9}$$

som kallas F1-score.

2.4 Holografisk mikroskopi

Holografisk mikroskopi skiljer sig från traditionell mikroskopi eftersom man inte mäter en projektion av föremålet man studerar. Istället sparas bilden som ett hologram vilket innebär att en del av de optiska egenskaperna hos provet bevaras. I praktiken använder man nästan alltid en annan ljuskälla som referens och låter ljuset som passerar genom provet och referensstrålen interferera. Strålen som leds genom provet kallas objektvåg och referensstrålen som referensvåg. Utifrån detta interferensmönster kan sedan en bild av provet rekonstrueras. Denna bild innehåller då information om både amplitud och fas hos objektvågen [7]. Nu för tiden används nästan uteslutande digitala detektorer för att mäta de interfererade vågorna och tekniken benämns då mer specifikt som *digital* holografisk mikroskopi. Vanligt är att en CCD-kamera används för att digitalt mäta upp interferensmönstret.

Det finns två ledande tekniker som används vid digital holografisk mikroskopi: off-axis DH och phase-shifting DH [7]. Förutom dessa är även in-line DH vanligt eftersom det är en väldigt enkel uppställning där referensvågen och objektsvågen infaller parallellt. In-line är även möjligt utan en referensstråle, och då används ljuset som inte bryts i provet som referens. Detta ställer dock ett krav på att provet inte får vara för stort så att referensstrålen inte störs för mycket.

Både off-axis DH och phase-shifting DH använder sig i princip uteslutande av laserljus, eftersom koherent ljus behövs för ett stabilt interferensmönster. Teknikerna är väldigt lika varandra och skillnaden mellan dem ligger i hur objekt- och referenssvågen interfererar. I off-axis DH så leds referensvågen in mot objektvågen med en vinkelskillnad. Detta ger upphov till ett skarpt interferensmönster och från fouriertransformen av denna kan man filtrera ut både noll:te ordningens diffraktionsspektrum och tvillingbilden som uppstår (dessa förklaras mer utförligt i kommande avsnitt). Den stora fördelen med en off-axis uppställning är just enkelheten att filtrera bort de oönskade termerna. Med en in-line uppställning så kommer den faktiska objektbilden att superponeras med noll:te ordningens diffraktionsspektrum och tvillingbilderna, vilket gör termerna betydligt svårare att filtrera ut [2].

Slutligen kan, med hjälp av en invers fouriertransform av den filtrerade bilden, både fas och amplitud fås [7]. I phase-shifting DH leds referensvågen först genom en fasmodulator och flera olika hologram mäts vid varierande fasskillnad. Fördelen med off-axis holografi är att informationen kan fås från endast ett hologram och den är därför snabbare än andra alternativ [2].

2.4.1 Hologram

I holografisk mikroskopi så använder man sig av en objektvåg E_O som passerar genom provet man vill studera och en referensvåg E_R. När dessa interfererar så blir intensiteten

$$I = |E_R + E_O|^2 = |E_R|^2 + |E_0|^2 + E_R^* E_O + E_R E_O^*. \tag{2.10}$$

Det är denna intensitet som nu för tiden vanligast mäts upp av en CCD kamera, men tidigare gjordes det analogt med fotomaterial. För att sedan rekonstruera en bild av provet så låter man en annan referensvåg E_R' transmittera genom hologrammet. Förr gjordes detta praktiskt genom att förbereda fotomaterialet så att transmissionsamplituden var proportionell med den uppmätta intensiteten, men i DHM så görs detta digitalt [2]. Vi får då:

$$\begin{aligned} E &= E_R' I = E_R' |E_R + E_O|^2 \\ &= E_R' |E_R|^2 + E_R' |E_0|^2 + E_R' E_R^* E_O + E_R' E_R E_O^*. \end{aligned} \tag{2.11}$$

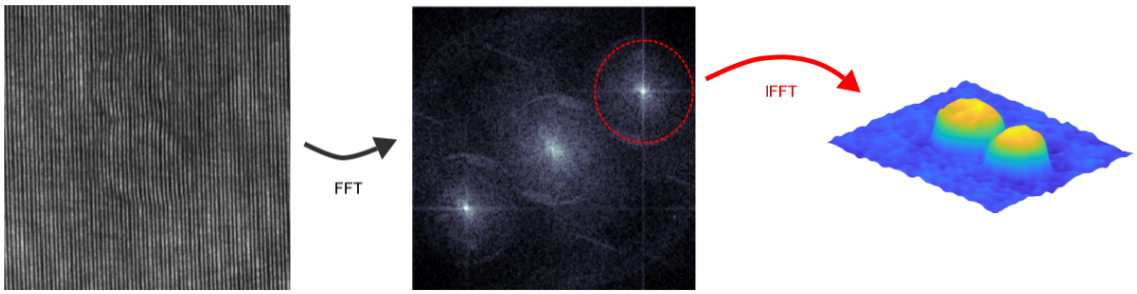
De första två termerna kallas för noll:te ordningens termer och representerar alltså de delar av objekt- respektive referensvåg som inte leder till diffraktion av den nya referensvågen. De två sista kallas för första ordningens tvillingtermer och representerar alltså de termer som leder till diffraktion. Notera att om referensvågorna är plana vågor så är tvillingtermerna proportionella till objektvågens komplexa fält. För att få en bild av provet behöver man alltså filtrera bort noll:te ordningens termer och en av tvillingbilderna som uppstår. Beroende på vilken konfiguration som används så finns det olika sätt att göra detta, och detta beskrivs i nästa avsnitt.

2.4.2 Bildrekonstruktion

I exempelvis en off-axis konfiguration är det betydligt enklare att filtrera bort de oönskade termerna eftersom noll:te ordningens termer hamnar i mitten av bildplanet, medan tvillingtermerna hamnar på var sin sida om centrum. Med en in-line konfiguration blir detta betydligt svårare eftersom samtliga termer överlappar i mitten av bildplanet [2].

Om mätningen görs digitalt så kan intensiteten som mäts upp av kameran, d.v.s. ekvation (2.10) användas vid rekonstruktionen, och termerna har då motsvarande namn som i ekvation (2.11).

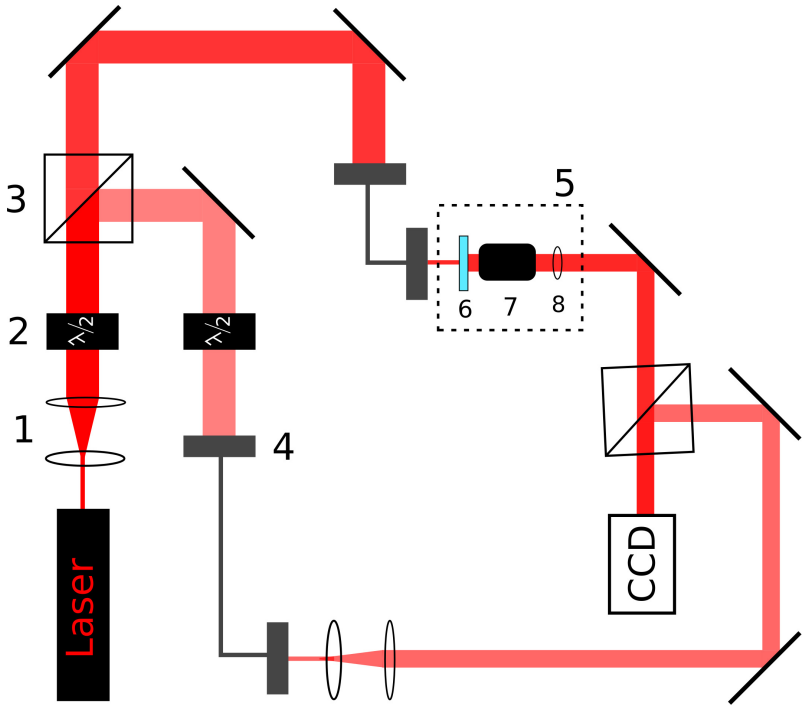
I en off-axis konfiguration infaller objektvågen och referensvågen med en vinkelskillnad, och vi antar nu att referensvågen är en plan våg. Efter en fouriertransform av bilden kommer då de två tvillingtermerna (se ekvation (2.11)) att hamna på var sin sida av centrum (se bild 2.8), medan noll:te ordningens term hamnar i centrum. Genom ett enkelt filter kan man då få ut en av tvillingtermerna, som är proportionell med objektvågens fält. Slutligen fås objektvågens fält genom en invers fouriertransform av den filtrerade bilden [2]. Hela arbetsgången illustreras i figur 2.8.



Figur 2.8: Rekonstruktion av hologram vid off-axis DHM. Notera att bilderna till vänster och i mitten inte motsvarar samma bild, utan endast används i illustrativt syfte. Egentligen skulle tvillingbilderna (en av dem markerad med rött) ligga på den horisontella axeln, precis som vågmönstrena i den vänstra bilden. Det resulterade komplexa fältet (höger) illustreras sedan som ett fasdiagram. Bilder av Daniel Midtvedt (handledare).

2.4.3 Experimentuppställning

Uppställningen som användes vid framtagningen av de experimentella bilderna som används för att testa modellen är en off-axis konfiguration, se figur 2.9 för schematisk ritning av experimentuppställningen. Lasern som används är en standard HeNe-laser med våglängd $\lambda = 633\,\text{nm}$. Laserstrålen passerar en beam expander (*sv.* strålexpanderare) följt av en halvvågsplatta innan den delas upp i en polarisationsberoende stråldelare till en objektstråle och en referensstråle där större delen av intensiteten tilldelas objektstrålen. Objektstrålen reflekteras på två speglar innan den med hjälp av en optisk fiber propageras till ett inverterat mikroskop. Det inverterade mikroskopet består av ett mikrofluidiskt chip, ett objektiv och en linstub. Objektstrålen som passerat mikroskopet reflekteras sedan till den andra stråldelaren (polarisationsoberoende). Referensstrålen passerar en andra halvvågsplatta, propagerar genom en optisk fiberkabel, en beam expander och reflekteras två gånger innan strålen kombineras med objektstrålen i stråldelaren. Slutligen detekteras de kombinerade strålarna i en CCD-kamera.



Figur 2.9: Figur för experimentuppställning. Komponenter: 1. strålexpanderare, 2. halvvågsplatta, 3. stråldelare, 4. optiskt fiber, 5. inverterat mikroskop, 6. provrör, 7. objektiv, 8. tublins.

2.4.4 Mie-teori

Mie-teori är döpt efter fysikern Gustav Mie och beskriver hur ljus bryts i homogena sfäriska objekt, t.ex. små partiklar. Teorin är i grunden exakt och inga approximationer behövs för att beskriva det spridda fältet [28]. Mie visade att lösningen till den vektoriella vågekvationen för sfäriska objekt kan skrivas som en summa av vector spherical harmonics, närmare bestämt magnetic harmonics och electric harmonics. Dessa ges av [28]:

$$\mathbf{M}_{\substack{e\\o}mn} = \nabla \times (\mathbf{r}\psi_{\substack{e\\o}mn})$$

och

$$\mathbf{N}_{\substack{e\\o}mn} = \frac{\nabla \times \mathbf{M}_{\substack{e\\o}mn}}{k}$$

där

$$\psi_{emn} = \cos(m\phi)P_n^m(\cos\theta)z_n(kr)$$

och

$$\psi_{omn} = \sin(m\phi)P_n^m(\cos\theta)z_n(kr).$$

Ovan är P_n^m de associerade Legendre-polynomen och $z_n(kr)$ en av de sfäriska Besselfunktionerna. Bohren och Huffman visar i *Absorption and Scattering of Light by Small Particles* [28] hur man kan bestämma det spridda fältet om man antar att det infallande fältet är en plan våg. Först skriver de om det infallande fältet som en summa av vector spherical

harmonics, och löser sedan ut expansionskoefficienterna genom att utnyttja ortogonaliteten hos Legendre-polynomen. Efter detta så kan man visa att det spridda fältet från en sfär blir:

$$\mathbf{E_s} = \sum_{n=1}^{\infty} E_n(ia_n\mathbf{N}_{eln}^{(3)} - b_n\mathbf{M}_{oln}^{(3)}),$$

där indexet, $^{(3)}$, syftar på att funktionerna genererats med sfäriska Besselfunktioner av tredje graden, även kallat Hankelfunktioner. Koefficienterna i ekvationen löses sedan med hjälp av angivna randvillkor. När denna teori sedan används numeriskt i en dator, vilket är fallet för programmet DeepTrack som används för att simulera träningsbilderna, används en approximation där man endast använder ett begränsat antal termer i summan ovan. Antal termer kallas ofta för L-talet.

2.4.5 Diffraktion

Mie-teorin beskriver det spridda fältet från partiklarna, men för övrig propagation i experimentuppställnigen så är det diffraktionsteori som ligger till grund för det uppmätta holografiska interferensmönstret. Eftersom detta arbete handlar om digital holografisk mikroskopi, och eftersom simuleringar görs, är det de numeriska metoderna som är mest intressanta. Det finns flera olika numeriska metoder som har både fördelar och nackdelar, bland annat: Fresnel diffraction, Huygens Convolution och Angular Spectrum Method. Metoden som används i DeepTrack är den sistnämnda, d.v.s. Angular Spectrum Method och bygger på principen att med hjälp av fouriertransform dela upp det infallande fältet i en summa av plana vågor, och sedan propagera dessa. Antag att vi har ett fält som propagerar i z-riktningen, då ges det resulterade fältet i termer av det ursprungliga fältet E_0 som [2]:

$$E(x,y;z) = \mathcal{F}^{-1}\{\mathcal{F}\{E_0(x_0,y_0)\}[k_x,k_y]\exp\left[iz\sqrt{k^2-k_x^2-k_y^2}\right]\operatorname{circ}\left(\frac{\sqrt{k_x^2+k_y^2}}{k}\right)[x,y]\}.$$

I numeriska beräkningar används ofta den diskreta Fast Fourier Transform (FFT). I två dimensioner ges denna som [2]:

$$\mathcal{F}(k_x,k_y) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{i2\pi(k_x x/M + k_y y/N)}.$$

I DHM när fältet propagerar genom ett mikroskop ges den slutgiltiga bilden av faltningen mellan det spridda fältet och mikroskopets komplexa så kallade *point spread function* (PSF) [2]. Detta gör att partikelbestämningen blir extra komplicerad.

2.4.6 Brusmiljöer vid DHM

Datorgenererad holografi producerar ideala hologram utan något brus. För att det neurala nätverket ska erhålla hög detekteringsnoggrannhet behöver det tränas i verkliga mätnings-

förhållanden med brus. Brusmiljön som förekommer vid DHM kan delas in i bakgrundsbrus och strukturellt brus. Bakgrundsbruset förekommer på grund av fotonfluktuationer som i sin tur ger upphov till elektronfluktuationer i CCD-kameran. Denna form av brus kallas ofta shot noise och dess fördelning följer en Poissonfördelning. Fotonfluktationerna beror på att fotoner är diskreta partiklar och detta leder till en liten varians i det ljus som träffar detektorns pixlar vilket ger upphov till bakgrundsbruset. Antalet fotoner som träffar en pixel under en bildtagning är, i vår experimentuppställning, grovt uppskattat i ordningen av 10^5 så bruset kan approximeras med en normalfördelning enligt centrala gränsvärdessatsen. Det strukturella bruset beror på att det uppträder reflektioner i den optiska anordningen, alltså linser, stråldelare och mikrofluidika chip, vilket ger upphov till ett karakteristiskt vågmönster.

2.5 Bestämma partikelns radie

Här presenteras två metoder för att bestämma partikelns radie utifrån den komplexa bilden som genereras från DHM, antingen med hjälp av Stoke-Einsteins relation eller genom att studera fasfältet och applicera Mie-teori.

Den metod som vanligtvis används för att mäta radien på nanopartiklarna fås genom Stoke-Einsteins relation. På grund av kollisioner med vibrerande vattenmolekylerna kommer partiklarna att ge upphov till en Brownsk rörelse, vilket betyder att dess rörelse är slumpmässig. Ett sätt att kvantifiera denna form av rörelse är genom diffusionskoefficienten D. Diffusionen av en partikel med Brownsk rörelse kan teoretisk beskrivas enligt

$$D = \mu k_B T \tag{2.12}$$

där T är vätskans temperatur, k_B är Boltzmanns konstant och μ är dess mobilitet. För sfäriska partiklar med lågt Reynoldtal, alltså vid laminärt flöde, kan mobiliteten beskrivas enligt $\mu = \frac{1}{6\pi\eta r}$ [29] där η är vätskans viskositet och r är den sfäriska partikelns radie. Insättning i ekvation (2.12) ger Stoke-Einsteins relation

$$D = \frac{k_B T}{6\pi\eta r}. \tag{2.13}$$

Partikelns rörelse bestäms genom att länka dess position från påföljande bilder tills ett komplett spår (*eng.* trace) erhålls. Från spåret beräknas diffusionskoefficienten och partikelns radie fås då från ekvation (2.13).

Den andra metoden använder sig av hologrammets fasfält. Den genererade bilden från hologrammet representeras av en komplex matris A från vilken med enkel matematik fasfältet kan erhållas. Fasfältet innehåller information om den optiska tjockleken. Den optiska tjockleken är en produkt av partikelns brytningsindex n och dess fysiska tjocklek t. Genom att anpassa distributionen av fasfältet till Mie-teori går det att bestämma storleken på partikelns radie.

2.6 Tidigare djupinlärningsmetoder för partikelspårning och karakterisering

Altman och Grier [30] utvecklade Catch-modellen vilket är en sammansättning av två nätverk. Det första nätverket är en mindre version av YOLOv3 [22], Tiny-YOLO, som

bestämmer partiklarnas x- och y-position samt en detektionslåda kring partikeln. Det andra nätverket bestämmer sedan partikelns z-position, radie och brytningsindex på beskärningar av orginalbilden som enbart innehåller en partikel. Modellen har en nackdel i att inferenstiden beror på antal partiklar i bilden. Det är en aspekt vi försöker förbättra i vårt arbete genom att arbeta med en end-to-end djupinlärningsmodell.

3

Metod

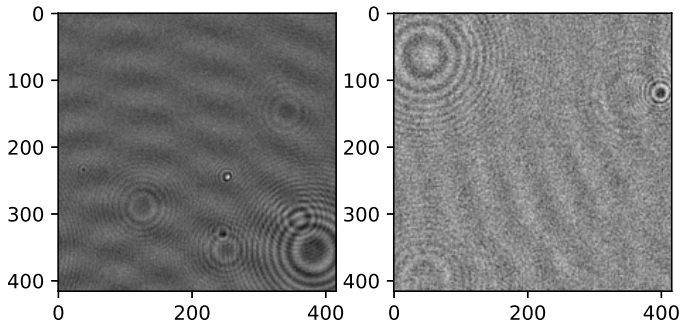
3.1 DeepTrack

DeepTrack, ursprungligen skapat av Helgadottir, Argun och Volpe [31], är ett python-bibliotek som kan användas bland annat för att simulera holografiska bilder. Paketet har senare vidareutvecklats till DeepTrack 2.0, nu även med B. och D. Midtvedt, Argun och Pineda som utvecklare [32]. I vårt projekt använder vi oss av DeepTrack för att simulera holografiska mikroskopbilder och konstruerar en egen implementation av YOLOv3-nätverket för partikelspårning.

I DeepTrack finns många funktioner för att simulera olika typer av holografiska mikroskopbilder och vi använder oss framförallt av deras inbyggda stöd för att simulera bilder från en off-axis konfiguration där partiklarna beskrivs av Mie-sfärer. Eftersom de rekonstruerade bilderna från Brightfield-mikroskopi och off-axis blir likartade, används deras funktioner för Brightfield-optik även för att simulera off-axis bilder. För att simulera det spridda fältet från Mie-sfärerna används Mie-teorin beskriven i avsnitt 2.4.4 och vid övrig diffraktion för att simulera en verklig experimentuppställning används Angular Spectrum Method som beskrivs i avsnitt 2.4.5.

3.2 Simulering av träningsbilder

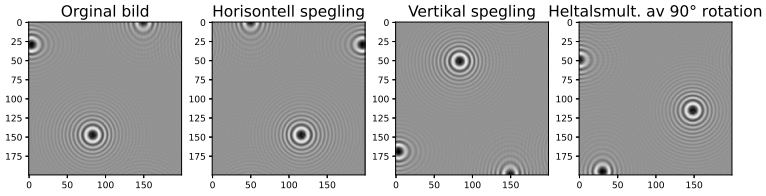
Träningsbilderna kan simuleras med en mängd olika parametrar. För ett flertal av dessa parametrar användes Pythons lambda-funktioner för att i varje bild och varje partikel ha olika parametrar. Dessa lambda-funktioner gjordes så att de gav en likformig fördelning mellan två gränser för parametrarna: partiklarnas x-, y- och z-position, brytningsindex samt radie men även det simulerade bruset, illuminations-gradienten, mängden partiklar i bilden, samt koma (en optisk aberration). Övriga parametrar sattes till lämpliga värden som överensstämde med laborationsuppställningen för den experimentella datan. Gränserna för de likformiga fördelningarna valdes för att likna olika typer av experimentbilder. De kan naturligtvis ändras till andra intervall om det skulle behövas ett större omfång eller om andra specifika parametrar ska studeras. Träningsbilderna var 416x416 samt 448x448 pixlar med en pixelstorlek på $1{,}14 \cdot 10^{-6}$ m. Vi använde en förstoring (*eng.* magnification) på 10. I figur 3.1 visas två exempelbilder:



Figur 3.1: Ett exempel på två simulerade bilder. Varje bild har olika mängd partiklar, brusmängd och illuminations-gradient. Detta för att skapa ett brett omfång vilket gör att nätverket lär sig på flera olika sorters bilder. Alla axlar har enheten pixlar.

3.3 Augmentering av simulerade bilder

Simuleringen av träningsdatan är beräkningsmässigt dyr och därför augmenteras de simulerade bilderna för att spara tid och processeringskraft. Augmenteringarna utförs med pythonpaketet *Albumentations* och appliceras på bilderna innan den skickas in i modellen under nätverkets träningsprocess. Augmenteringarna som appliceras på bilderna är vertikal spegling, horisontell spegling och slumpmässig rotation i heltalsmultiplar av 90°, se figur 3.2 för exempel på augmenteringarna. Applikationen av augmenteringarna på simuleringsbildenerna är viktade med olika sannolikheter där de kan förekomma i olika kombinationer. Totalt ökas mångfalden av datan med en faktor 8.



Figur 3.2: Augmentationer som appliceras på de simulerade bilderna. Alla axlar har enheten pixlar.

3.4 Tillägg av brus i simulerade bilder

Brusmiljöer som förekommer vid DHM som nämnts i 2.4.6 är bakgrundsbrus och strukturellt brus. Bakgrundsbruset implementeras med inbyggda funktioner från DeepTrack. Bruset adderas i mikroskopets illumination tillsammans med eventuell illuminationsgradient. En Gaussisk brusfördelning väljs med $\mu = 0$ samt likformigt fördelad standardavvikelse i intervallet $\sigma = [0, 0.05]$.

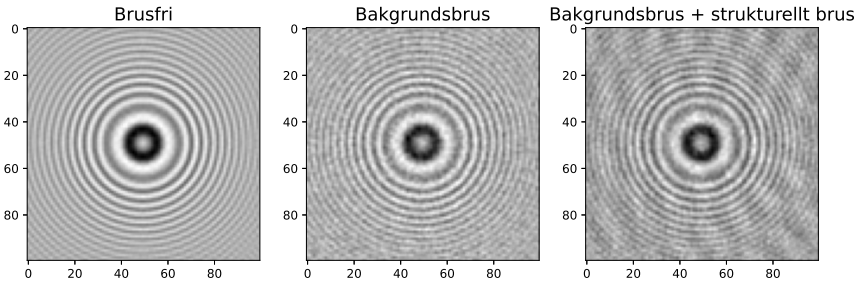
Det strukturella bruset superponeras med bilden som simuleras från DeepTrack. Bruset realiseras med bruskällor vars positioner är slumpade inom en annulus centrerad i bildcentrum med innerradie $r_i = 0.75d$ och ytterradie $r_y = d$ där d är bildens diameter. Brusets

komplexa fält i en bildpixel $\vec{r_p}$ definieras som

$$E_{SB}(\vec{r_p}) = \sum_{\text{källor}} A \left(\frac{\sin(R_s f)}{1 + R_s \gamma} \right) \cdot e^{if R_s} \tag{3.1}$$

där A är en amplitudsfaktor, γ är en avtagningsfaktor för amplituden, f är relaterad till oscillationshastigheten av vågmönstret och $R_s = \left\| \vec{r_p} - \vec{r_b} \right\|$ är avståndet mellan bildpixeln $\vec{r_p}$ och bruskällan $\vec{r_b}$. Antalet bruskällor väljs från en likformig fördelning mellan en och två källor varför en summa används i ekvation (3.1).

Tillägget av båda brusformerna visas i figur 3.3. Interferensringarna för de brusfria bilderna är helt synliga, även de svagaste långt ifrån partikelcentrum. En stor nackdel med de simulerade bilderna är att på grund av periodiska randvillkor inom matematiken uppstår det en ofysikalisk reflektion av interferensmönstret vid bildens rand vilket ger upphov till att ringarna interfererar med sig själv. Denna ofysikaliska effekt löses genom tillägget av bakgrundsbruset eftersom intensiteten av de yttre ringarna är betydligt svagare än brusgolvet. Utökningen med det strukturella bruset medför vågfronter som både interfererar med partiklarnas ringar och modifierar bakgrundsbelysningen likt de omständigheter som uppstår vid experimentella mätningar.



Figur 3.3: Simulerad bild från DeepTrack utan brus, med bakgrundsbrus och båda brusformerna. Alla axlar har enheten pixlar.

3.5 Nätverkets arkitektur

Modellen bygger på YOLOv3-arkitekturen och tränades på de simulerade bilderna enligt en process som beskrivs i avsnitt 3.8. För en film av experimentbilder kan nätverket sedan bestämma parametrar i varje sekvens av filmen. Den originella YOLOv3-modellen bestämmer objektets egenskaper med parametrarna: position i x. respektive y-led, höjd, bredd samt vilken klassindelning objektet tillhör. Modellen i föreliggande rapport använder sig av samma struktur men bestämmer istället parametrarna: position i x respektive y-led, position i z-led, radie på partikeln och brytningsindex. Modellen förutsäger alltså förutom position i x, y även positionen i z-led och de unika parametrarna radie och brytningsindex som är högst relevanta för just partikeligenkänning. Vid träning och implementering av modellen användes genomgående pythonbiblioteket PyTorch.

3.6 Generering av ankare till modell med K-means clustering

Som beskrivet i 2.2 implementerade YOLOv2 ankare (*eng.* anchor boxes) [25] för att lättare kunna bestämma objekt av olika storlekar. Det fanns totalt nio i deras YOLO-modell och deras bredd och höjd bestämdes med metoden K-means clustering. Vid detektion av nanopartiklar kan däremot bredd och höjden definieras på olika sätt, som exempelvis i [30] där de definieras som ett antal interferensringar bort från centrum. Denna definition kommer, på grund av att partiklarna är sfäriskt symmetriska, att förutsäga lika stor bredd som höjd och metoden ger ett mått på interferensringarnas period. Samma definition på bredd och höjd testades även på våra partiklar, men på grund av att radien är av större intresse än interferensringarnas period bestämdes det att modellen inte skulle definiera bredd och höjd för våra ankare. Även om interferensringarnas period tillsammans med partikelns brytningsindex relateras till partikelns radie enligt Mie-teori bestämdes det istället att vår modells ankare skulle definieras med radien.

Med andra ord, istället för att vår modell har ankare definierade med bredder och höjder, är de definierade som flera olika radier. Den första fördelen med detta är att modellen då kommer förutsäga radien direkt, vilket är av intresse när man vill bestämma vilken typ av partikel det är. Den andra fördelen är att, på samma sätt som YOLO-modellen, att nätverket blir bättre på att bestämma partiklar med olika stora radier. Det blir alltså lättare för nätverket att upptäcka partiklar som har starkt varierande radie i samma bild. Våra ankare som också var nio stycken bestämdes på samma sätt som för YOLO-modellen, med metoden K-means-clustering. Ankarna som vår modell använde är (avrundat):

$$\begin{aligned}&(292,\ 275,\ 257,\\ &\ 240,\ 223,\ 206,\\ &\ 190,\ 173,\ 158)\ \text{nm}\end{aligned} \tag{3.2}$$

där varje rad tillhör var sin rutnätsskala. Ankarna skulle kunna modifieras om datan innehåller större eller mindre partiklar.

3.7 Modellens kostnadsfunktion

Målet med träningen av modellen är att kunna bestämma partikelns x-, y- och z-position samt partikelns radie och brytningsindex. Till största del användes kostnadsfunktionen från YOLOv3 [22] men med modifieringar för att inkludera bestämning av partikelradien, z-positionen och partikelns brytningsindex. I ursprungliga publiceringen av YOLO delas kostnadsfunktionen in i fyra delar: icke-objektskostnad, objektskostnad, koordinatkostnad och klasskostnad [25]. Vår modell använder enbart icke-objektskostnad, objektskostnad och koordinatkostnad och delar upp koordinatkostnaden i fyra delar för x-, y- och z-positionen samt radie och brytningsindex. Modellen utför positionsbestämningar och karakteriseringar av partiklar på tre olika rutnätsskalor där antal celler beror på bildens storlek. För varje cell utför modellen tre olika förutsägelser, en för varje ankare som är associerad till den rutnätsskalan. Modellen tränades på data med motsvarande korrekta positionsbestämningar och karakteriseringar av partiklar. Härefter används position för att hänvisa till en av förutsägelserna (motsvarande en viss ankare) i en specifik cell i modellens output och på en viss skala. Varje position har en binär objektsklassificering som avgör om den har tilldelats ett objekt i bilden. I denna rapport avser objekt en partikel.

För de positioner utan objekt definieras indikatorparametern

$$\mathbb{1}_{aij}^{noobj} = \begin{cases} 1 & \text{om } t_{aij} \text{ ej har ett objekt} \\ 0 & \text{annars} \end{cases} \tag{3.3}$$

och för positionerna som tilldelats ett objekt sätts motsvarande indikatorparameter, $\mathbb{1}_{aij}^{obj}$, till 1. Modellens output för en viss bild betecknas t. För en särskild position är det t_{aij} där a är index för den ankare positionen associeras med på en viss skala, i är raden och j är kolumnen i det rutnät för vilket modellen ger sin output. Varje position t_{aij} har sex värden för: objektssannolikheten, x-, y-, och z-koordinater samt radiebestämning (r) och brytningsindex (n). Motsvarande korrekta positioner betecknas y_{aij}. För en viss partikel i bilden bestäms a för att minimera 1-normen mellan partikelns radie och ankaren som associeras med positionen. Indexeringen (i,j) syftar till den cell i modellens output där partikeln har sin mittpunkt.

Icke-objekts- eller objektskostnaden beräknas för samtliga positioner men koordinatkostnaden beräknas endast för de positioner som tilldelats objekt i en viss bild. För de positioner som inte tilldelats ett objekt är objektsklassificeringen 0 och vi applicerar en binary cross entropy (BCE) kostnad enligt

$$L_{noobj}^{aij} = -\log\left(1 - \sigma\left(t_{aij}^{o}\right)\right),$$

där o betecknar objektssannolikheten i modellens output. Sigmoid-funktionen, σ, beräknas enligt (2.5).

För de positioner som har ett objekt ska modellen bestämma en objektssannolikhet som representerar avståndet i x- och y-led till en partikel i bilden. För en viss position innebär det

$$d_{aij} = \max\left(1 - \frac{1}{2}\left(|\sigma(t_{aij}^{x}) - y_{aij}^{x}| + |\sigma(t_{aij}^{y}) - y_{aij}^{y}|\right), 0\right).$$

Objektskostnaden för en position beräknas sedan med BCE enligt

$$L_{obj}^{aij} = -\left[d_{aij} \cdot \log\sigma(t_{aij}^{o}) + (1 - d_{aij}) \cdot \log\left(1 - \sigma(t_{aij}^{o})\right)\right],$$

och för samtliga positioner med eller utan objekt ges kostnaden av

$$L_{\substack{noobj \\ obj}} = \frac{1}{\sum_{a,i,j} \mathbb{1}_{aij}^{\substack{noobj \\ obj}}} \sum_{a,i,j \in \mathbb{1}_{aij}^{\substack{noobj \\ obj}}} L_{\substack{noobj_{aij} \\ obj}}^{aij}. \tag{3.4}$$

Koordinatkostnaden beräknas med MSE och för x- och y-koordinaterna ges kostnaden av

$$L_{xy} = \frac{1}{\sum_{a,i,j} \mathbb{1}_{aij}^{\text{obj}}} \sum_{a,i,j \in \mathbb{1}_{a\ ij}^{\text{obj}}} \frac{1}{2}\left(\left(\sigma(t_{aij}^{x}) - y_{aij}^{x}\right)^2 + \left(\sigma(t_{aij}^{y}) - y_{aij}^{y}\right)^2\right). \tag{3.5}$$

Motsvarande kostnader för z-koordinaten, radien och brytningsindex ges av

$$L_{n}^{z} = \frac{1}{\sum_{a,i,j} \mathbb{1}_{aij}^{\text{obj}}} \sum_{a,i,j \in \mathbb{1}_{a\ ij}^{\text{obj}}} \left(t_{aij}^{\substack{z \\ n}} - y_{aij}^{\substack{z \\ n}}\right)^2 \quad \text{och} \tag{3.6}$$

$$L_{r} = \frac{1}{\sum_{aij} \mathbb{1}_{aij}^{\text{obj}}} \sum_{a,i,j \in \mathbb{1}_{aij}^{\text{obj}}} \left(10^{-7} t_{aij}^{r} - y_{aij}^{r}\right)^2. \tag{3.7}$$

Den totala kostnaden ges av en viktad summa av de enskilda kostnaderna (3.4), (3.5), (3.6), (3.7) enligt

$$L = \lambda_{noobj} L_{noobj} + \lambda_{obj} L_{obj} + \lambda_{xy} L_{xy} + \lambda_z L_z + \lambda_r L_r + \lambda_n L_n, \tag{3.8}$$

där λ_i är konstanter och bestäms för att balansera kostnadstermerna.

3.8 Träning och hyperparametrar

Modellen tränades på 150 000 kvadratiska simulerade bilder av storlek 416-448 pixlar. Totala träningen upptog ungefär 300 epoker och utfördes på delar av träningsdatan om 20 000-50 000 bilder. Varje del av träningsdatan användes under den träningstid som krävdes för att maximera förbättringen i precision och recall på valideringsdatan. Modellens initiala learning rate sattes till $\alpha = 10^{-4}$ och den sänktes sedan i två steg till $\alpha = 10^{-5}$ i och med att minskningen i kostnadsfunktionen stagnerade. Kostnadstermerna balanserades med λ_i för att vara ungefär samma magnitud under träningen. Vi använde L2-regularisering med parameter $l = 10^{-5}$ för att motverka överanpassning till träningsdatan. Dataaugmentering samt randomisering av graden på strukturellt brus användes även under delar av träningen.

3.9 Postprocessering av output från djupinlärningsmodell

Modellens output är en array av tre tensorer som motsvarar var och en av de tre skalor modellen tränas på med olika rutnätsstorlekar S. Tensorn från en viss skala med storlek S har dimensioner $(N, 3, S, S, 6)$. Vi betecknar antal bilder i en träningsbatch N och antal ankare som associeras till varje skala är tre. Modellens output, t, för varje position i en bild består av värden för $\{o, x, y, z, r, n\}$ där o är objektssannolikheten, r är modellens radiebestämning och n betecknar brytningsindex. Vi konverterar modellens output enligt följande formler

$$\begin{aligned} b_o &= \sigma(t_o) \\ b_x &= \sigma\left(t_x\right) + c_x \\ b_y &= \sigma\left(t_y\right) + c_y \\ b_z &= t_z \\ b_r &= 10^{-7} t_r \\ b_n &= t_n \end{aligned}$$

för en partikelbestämning t där c_x och c_y är kolumnen respektive raden i det rutnät som modellen gör sin output på. Sigmoid-funktionen, σ, beräknas enligt (2.5) och används för att normalisera modellens output till intervallet [0,1]. Oavsett om det finns en partikel på positionen ges koordinater i modellens output. För att bestämma vilka koordinater som motsvarar en faktisk partikel tar vi bort alla förutsägelser som har $b_o < \rho$ där ρ är ett gränsvärde som sätts för att optimera precision och recall för modellen. På de kvarvarande förutsägelserna utför vi sedan Non-maximum Suppression (NMS) som eliminerar positionsbestämningar som är närmare varandra än ett visst avstånd ϵ i pixlar i x- och y-position. NMS är en algoritm som iterativt jämför två positionsbestämningar och om avståndet mellan dem är lägre än en konstant ϵ behålls den med högst b_o. Pseudokod för NMS visas i Appendix A.1 där jämförelser av två detektionslådor utförs med snittet över unionen.

3.10 Algoritm för partikeldetektion

För att nätverket ska göra en detektion på en bild görs först flera transformationer på bilden innan den skickas in i nätverket. Först konverteras bilden till att ha tre kanaler. Dessa kanaler skulle i en vanlig bild motsvara RGB-värden (röd, grön och blå) men i vår modell är de tre kanalerna den reella, imaginära, samt absolutbeloppet av det komplexa fältet som fås vid en mikroskopisk holografi-mätning. Varje kanal utfylls (*eng.* pad) så att bilden blir kvadratisk, därefter normaliseras kanalerna. Detta görs med python-biblioteket Albumentations. När bilden har modifierats enligt dessa transformationer skickas den in i nätverket i flera beskärda segment (*eng.* patches). Nätverket körs sedan på varje av dessa segment. Efteråt utförs postprocessering av modellens output med NMS, se 3.9.

4

Resultat

4.1 Resultat på simulerad data

Den simulerade testdatan består av 5000 bilder med 448x448 pixlar med 0-10 partiklar i varje bild. Modellen fick på testdatan recall 0.96 och precision 0.87 vilket ger ett F1 score på 0.91. I tabell 4.1 visas modellens resultat i koordinatbestämmelserna med root mean square error (RMS) samt det relativa felet.

Tabell 4.1: Resultat på simulerad testdata bestående av 5000 bilder av storlek 448x448 pixlar med 0-10 partiklar på varje bild.

	x	y	z	radie	brytningsindex
RMS	0,89 px	0,88 px	80 nm	23 nm	0,049
relativt fel	0,028	0,038	0,0753	0,0754	0,0227

Det relativa felet och RMS mäts endast på de partiklar som betraktas som TP i förhållande till de korrekta positionerna. Vi räknar en partikel som en TP detektion om medelvärdet för RMS i x- och y-led är mindre än 5 pixlar från en korrekt partikelposition.

Modellen testades även på 60 simulerade bilder av storlek 2048x2048 pixlar med 20-80 partiklar. Dessa bilder har en högre maximal partikeldensitet än på både experimentbilder och den simulerade träningsdatan. På dessa bilder är felet i bestämmelserna för z, radien och brytningsindex motsvarande vad som redovisas i tabell 4.1 och vi uppnår RMS i x- och y-position på mindre än 0.7 pixlar. Däremot detekterar vi endast 90 % av partiklarna i dessa bilder. I de fall där partiklar missas verkar det vid inspektion som att det sker framförallt i bilder med högre antal partiklar.

4.2 Resultat på experimentell data

Den experimentella datan är från en holografisk mikroskopi-mätning där partiklar av samma typ användes (220 nm radie och brytningsindex på 1,58). Med den traditionella algoritm som vi jämför vår modell med bestäms partiklarnas x- och y-koordinater med en annan grund än i den simulerade datan som vi tränat på. Av detta skäl finns ett systematiskt fel i x- och y-positionsbestämmelserna samt modellens recall och precision är svårare att kvantifiera.

På grund av det systematiska felet i positionen räknar vi en partikel som en TP detektion om medelvärdet för RMS i x- och y-led är mindre än 10 pixlar från en korrekt partikel-

position. Vi kan då detektera 97 % av partiklarna i bilderna. På grund av att det saknas korrekta positioner för flera partiklar i varje bild är inte precision ett relevant mått. I tabell 4.2 visas modellens resultat för bestämning av radien och brytningsindex. Vidare är medelvärdet av radien och brytningsindexet över alla partiklar 200 nm samt 1,50 respektive.

Tabell 4.2: Resultat på experimentell testdata bestående av en film med 140 sekvenser av storlek 1944x1458 pixlar med 30-50 partiklar på varje bild.

	radie	brytningsindex
RMS	35 nm	0,1
relativt fel	0,144	0,0592

4.3 Inferenshastighet

Vi testar hastigheten för inferens på bilder av storlek 2240x2240 pixlar och mäter modellens körningstid samt postprocessering av output med konvertering av positioner och NMS. Testen utförs på en RTX3090. I tabell 4.3 visas tiderna för inferens på både överlappande och ej överlappande beskärningar av bilderna. Vi ser ingen signifikant skillnad i felet på koordinatbestämmelserna utan överlappningar och på den experimentalla testdatan detekterar vi 95 % av partiklarna i bilderna.

Tabell 4.3: Inferenstider för modell och postprocessering för en bild av storlek 2240x2240 pixlar. Vid överskjutande beskärningar av bilden körs modellen på bilder av storlek 448x448 med en steglängd på 224 i bredd och höjd.

	överlappande	ej överlappande
ms/bild	536	183

4.4 Jämförelse med traditionella metoder för partikeldetektion

Att positionsbestämma och karakterisera partiklar från holografiska mikroskopibilder är en komplicerad process, speciellt vid bestämning av partiklarnas z-position. Två av de mest använda teknikerna är Lorenz-Mie Fitting och Rayleigh-Sommerfeld Back-Propagation. I vår analys kommer vi framförallt jämföra vår modell med Lorenz-Mie Fitting.

Lorenz-Mie Fitting bygger på Mie-teorin beskriven i avsnitt 2.4.4 och innebär att man gör en modellanpassning som så nära som möjligt överensstämmer med den uppmätta bilden. Utifrån den anpassade modellen kan man sedan få ut partikelns position, radie och brytningsindex. Metoden har visat sig vara väldigt framgångsrik framförallt på grund av dess höga precision som vid rätt förutsättningar uppskattas vara under 1 nm. Dock är det svårt att verifiera eftersom det inte finns andra tekniker med samma höga precision [33]. En av de stora nackdelarna är att man behöver en viss kännedom om partiklarna, t.ex. antalet, deras brytningsindex eller ungefärlig storlek för att kunna göra en bra modellanpassning. Metoden är också väldigt beräkningstung och känslig för brus [34].

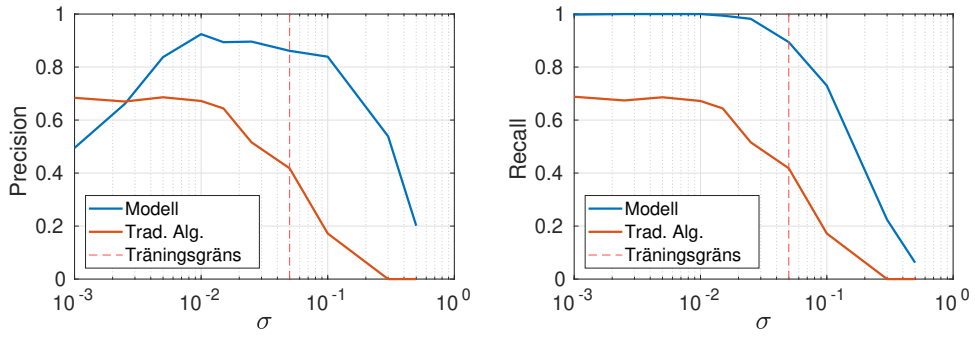
För att jämföra vår modell med Lorenz-Mie Fitting används ett python bibliotek kallat

Pylorenzmie som har funktioner för att anpassa en modell utifrån holografiska mikroskopibilder. Denna anpassning har just nackdelen att man måste ha lämpliga initialvärden för att få en bra anpassning. En vanlig metod för att komma runt detta hinder är att först använda en annan teknik för att lokalisera och isolera mönstret kring varje partikel. Detta görs i Pylorenzmie genom att först applicera en cirkulär transformation som förstärker cirkulära mönster i bilden. Efter transformationen bestäms partiklarnas centrum med hjälp av gradienten för den transformerade bilden. Den fullständiga algoritmen kallas för *Orientation Alignment Transform* (OAT) och beskrivs mer grundligt i [35]. Efter att partiklarnas position har uppskattats så uppskattas även en detektionslåda, d.v.s. ett kvadratiskt område kring varje partikel. Detta område används sedan för att ansätta en modell efter Mie-teorin, med endast en partikel per bild. Detta gör att antalet partiklar för varje anpassning är känt, men det gör också att metoden inte är lämplig i de fall när partiklar är så nära varandra att deras ringmönster överlappar.

4.4.1 Lokalisering

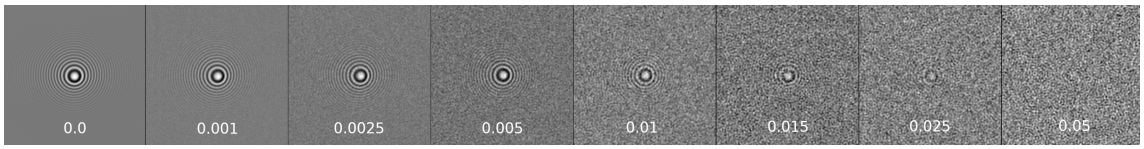
Eftersom partikelspårningen har en tydlig uppdelning i Pylorenzmie-paket är det naturligt att också jämföra dessa två moment för sig. Först jämför vi alltså lokaliseringen av ringmönstrerna som ger en grov uppskattning om vart partiklarna befinner sig. Båda metodernas precision och recall mäts och dessa parametrar jämförs med varandra. Detta görs vid varierande grad av Gaussiskt brus. Jämförelsen görs dessutom med varierande antal och överlappande partiklar.

I figur 4.1 visas recall och precision för vår modell respektive OAT vid varierande nivå av brus. Mätningen består av analyser av 500 stycken simulerade bilder med en partikel i varje. På x-axeln är standardavvikelsen σ för den Gaussiska fördelning som användes för att lägga till brus i den simulerade bilden. I figur 4.2 visas flera simulerade bilder med angiven standardavvikelse för bruset som jämförelse. Detta visar alltså hur väl vårt nätverk presterar i det fallet då det endast är en partikel som ska detekteras. Notera att precision för vår modell vid väldigt låg brus är något lägre än för lite högre brus.



Figur 4.1: Precision (vänster) och recall (höger) för vår modell och OAT-algoritmen vid lokalisering av en partikel. Precision och recall plottas mot σ som är standardavvikelsen för den Gaussiska fördelning som användes för att genera bruset i bilden. Träningsgränsen motsvarar det maximala brus som användes vid träningen. Notera att x-axeln är logaritmerad.

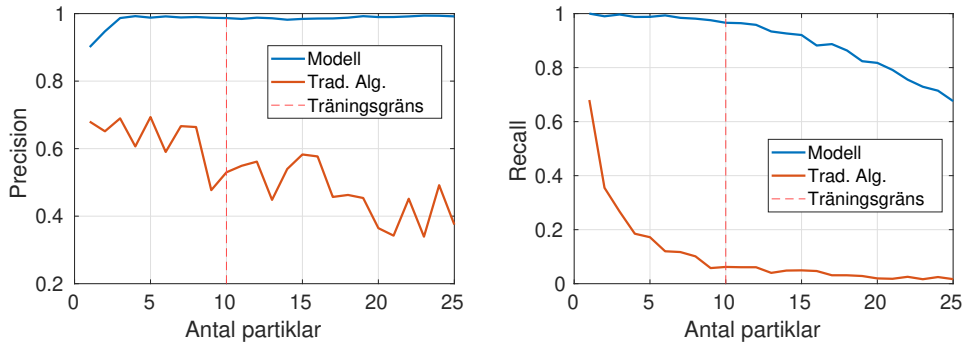
Resultatet för recall och precision vid varierande antal partiklar kan ses i figur 4.3. I denna mätning simulerades bilder med storleken 448x448 pixlar och det konstanta värdet på standardavvikelsen 0.01 för det Gaussiska bruset. Notera att recall faller betydligt



Figur 4.2: Simulerade bilder med en partikel och olika varierande nivå av brus som jämförelse. Värdet på σ är angiven i respektive bild, och värdet 0.05, d.v.s. bilden längst till höger motsvarar träningsgränsen i figur 4.1.

snabbare för OAT än för vår modell, vilket tyder på att djupinlärningsmodellen hanterar bilder med fler partiklar betydligt bättre.

Med en högre partikeldensitet ökar också sannolikheten för att partiklarnas ringmönster överlappar, vilken innebär problem för de flesta traditionella metoder. I figur 4.4 visas resultatet från en analys där bilder med två partiklar simulerats med varierande pixel-avstånd mellan varandra. För varje avstånd har 50 simuleringar gjorts och det är total precision och recall som är angiven. NMS gräns motsvarar det pixel-avstånd i NMS-steget inom vilken modellen anser att två förutsägelser syftar på samma partikel. Detta är en hård gräns, och modellen kan omöjligt identifiera partiklar närmare varandra.

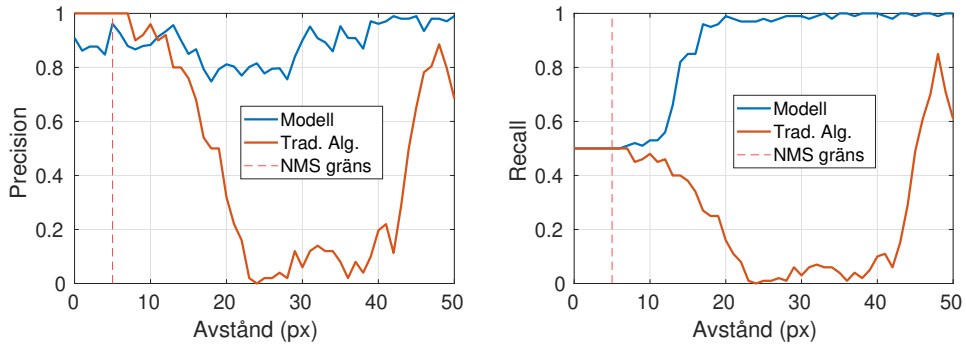


Figur 4.3: Precision (vänster) och recall (höger) för vår modell och OAT-algoritmen vid lokaliseringen av fler partiklar. Precision och recall plottas mot antalet partiklar i bilden. Träningsgräns motsvarar det maximala antalet partiklar som användes vid träningen. Bilderna har det konstanta värdet 0.01 på standardavvikelsen för det Gaussiska bruset, och för varje antal partiklar har 100 bilder analyserats.

4.4.2 Karakterisering

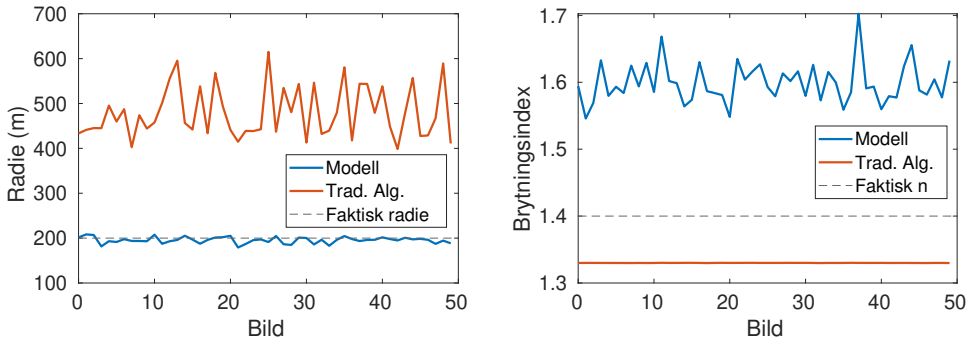
Vår modell lokaliserar och bestämmer alla parametrar i ett steg, men vid traditionella metoder, och även en del DL-metoder (se t.ex. [30]), delar man oftast upp lokalisering och karakterisering i två steg. I paket Pylorenzmie lokaliseras partiklarna först med hjälp av OAT-algoritmen, och en detektionslåda till varje partikel bestäms. Den grova uppskattningen används sedan som initialvärden till den Lorenz-Mie-anpassning som görs för varje detektionslåda.

För att göra en rättvis jämförelse med den noggrannhet som kan uppnås med Lorenz-Mie-anpassning, och för att den ursprungliga lokaliseringen inte ska påverka resultatet, simuleras nya bilder med en partikel och OAT-algoritmens uppskattningar används som initialvillkor.



Figur 4.4: Resultatet av analys där två partiklar simuleras intill varandra. På y-axlarna är precision (vänster) respektive recall (höger) och på x-axeln är avståndet mellan partiklarna. Vid varje avstånd har 50 olika bilder simulerats med slumpmässig vinkel mellan partiklarna.

I figur 4.5 visas resultatet av hur väl vår modell bestämt radien och brytningsindex hos en partikel i 50 olika simulerade bilder. Det är samma partikel i samtliga bilder, och standardavvikelsen hos det Gaussiska bruset är 0,005, men de resulterade bilderna varierar fortfarande något eftersom både bruset och belysningen i simuleringen är slumpmässiga. Som jämförelse används funktionen för att göra en Mie-anpassning i paketet Pylorenzmie. Notera att det är fördelningen som ger ett värde på metodens effektivitet, och att en offset inte har stor betydelse på grund skillnaden mellan olika experimentella uppställningar. Exempelvis kan olika uppställningar ha olika typer av behållare för provet med olika brytningsindex och därmed förändra bilderna. I Pylorenzmie är också Mie-anpassningen anpassad efter deras experimentuppställning och det är inte oväntat att det uppskattade värdet skiljer sig lite från det faktiska. Det är även värt att notera att Mie-anpassningen kunde fått bättre resultat om någon parameter hade angivits exakt. Nu när ingen parameter är känd så finns det en risk att algoritmen korrigerar sin karakterisering felaktigt. Standardavvikelsen för både Mie-anpassningens uppskattningar och vår modell kan ses i tabell 4.4.



Figur 4.5: Uppskattade värden på radie och brytningsindex av vår modell och Mie-anpassningen i Pylorenzmie-paketet. Analysen har gjorts på 50 stycken olika simulerade bilder med en partikel i varje. Trad. Alg. syftar på den traditionella algoritmen, d.v.s. en version av Mie-anpassning. Även det faktiska värdet på radien och brytningsindex är markerade i bilden.

Tabell 4.4: Standardavvikelsen för uppskattad radie och brytningsindex för 50 stycken simulerade bilder med en partikel i varje. Trad. Alg. syftar på den traditionella metod som jämförs med, i detta fall en version av Mie-anpassning.

	Modell	Trad.
σ_r	6,9587 nm	57,397 nm
σ_n	0,0301	9,42e-05

4.5 Jämförelse med CATCH

Det finns andra väldigt lyckade försök att använda sig av djupinlärning för att karakterisera och spåra nanopartiklar varav den mest nämnvärda troligtvis är CATCH [30] av Altman och Grier. Den största skillnaden mellan deras metod och vår är att de har delat upp problemet i två delar. Först lokaliserar ett nätverk partiklarna och deras omgivande ringmönster. En beskärning av en omgivning kring varje partikel är sedan input i ytterligare ett nätverk som bestämmer resterande parametrar. De rapporterar en recall på över 0,99 vilket är något högre än den recall på 0,96 vi presenterade i avsnitt 4.1, men det är svårt att göra en rättvis jämförelse då deras simulerade hologram endast hade 5 % additivt Gaussiskt brus och heller inget strukturellt brus. Altman och Grier skriver också att deras modell inte rapporterade några FP medan vår modell tenderade att göra det vid väldigt lågt brus. Vidare skriver Altman och Grier att de på simulerad data kan bestämma x- och y-positionerna med ett fel mindre än 1,5 pixlar, vilket är något högre än det pixelfel på 0,9 som vi rapporterade i avsnitt 4.1.

På experimentell data rapporterar Altman och Grier RMS-värden på 89 nm för radien och 0,04 för brytningsindex [30]. Jämför vi med resultatet från vår modell, se tabell 4.2, så har vi nästan hälften så stort fel vid bestämning av radie, men dubbelt så stort fel för brytningsindex. De arbetar också med större partiklar vilka är lättare att storleksbestämma, och därför går det inte att jämföra resultaten rakt av. Värt att nämna är att Altman och Grier rapporterar samma RMS-värden för radie och brytningsindex för både simulerad och experimentell data, medan vi fick något sämre resultat på experimentdata, jämfört med våra simuleringar.

Det går inte att dra några säkra slutsatser om felmarginalerna i position eftersom den faktiska positionen inte är känd för experimentell data, istället blir man tvungen att enbart jämföra resultaten med en annan metod som också har en felmarginal. Dock rapporterar de att deras modell rapporterar en förväntad partikeldensitet och alltså lyckas lokalisera de flesta av partiklarna från experimentdata. Detta stämmer väl överens med den recall på 97 % som vi presenterade i avsnitt 4.2.

5

Diskussion

5.1 Utvärdering av resultat på simulerad data

Den simulerade testdatan har samma fördelning som träningsdatan och simulerades för att likna den experimentella datan. Fler än en partikel men sällan fler än tio är vanligt vid experimentell mätning likväl som vid simulering vid vald bildstorlek.

Bilder utan partiklar inkluderas för att testa nätverket på bilder med avsaknaden av partiklar. Eventuella felmätningar kan vid experiment ge upphov till defekta bilder som inte innehåller partiklar eller ens relevant information. Modeller måste därför prövas på simulerad testdata som även representerar den typen av mätning.

Det relativa felet för z-koordinaten är klart högre än för x respektive y. Detta beror på att kameran filmar x-y-planet och z-axeln går rakt längs kamerariktningen. Den enda indikatorn på z-koordinaten är partikelns skenbara storlek i bilden varför att avgöra z-koordinaten med hög precision är besvärligt både för det mänskliga ögat och modellen.

Att bestämning av koordinaterna z, radie och brytningsindex för simulerad testdata bestående av 20-80 partiklar motsvaras av de erhållna resultaten av 0-10 partiklar är ett gott tecken för modellens brukbarhet för mer partikeltäta bilder. Att felet i x- samt y-koordinater vidare är lägre än 0.7 pixlar är ytterligare en indikation på att modellen klarar av ett stort antal partiklar utan bekostnad av precision av de påträffade partiklarna. De 10 % av partiklarna som emellertid inte upptäcks tyder på att modellen har svårt att hitta partiklar i bilder med högre partikeldensitet än träningsdatan.

5.2 Utvärdering av resultat på experimentell data

Den experimentella testdatan valdes till en film av 140 sekvenser och upplösningen 1944x1458 samt 30-50 partiklar per bild eftersom föreliggande data är representativ av vad som fås av experimentell mätning vid digital holografi.

En detektion på 97 % av partiklarna i bilderna anses vara goda resultat. Att precision saknar relevans som mått på grund av avsaknaden av riktiga positioner för åtskilliga partiklar försvårar dock utvärderingen av resultatet något.

Vidare är det relativa felet av radien för resultatet på den experimentella datan nästan dubbla det relativa felet av radien för resultatet på den simulerade datan. Även det relativa felet för uppskattning av brytningsindex är för resultatet på den experimentella datan

nästan dubbla den simulerade datan. Det finns samtidigt en klar anledning att förvänta sig ett större fel på igenkänning av experimentell data gentemot simulerad data av anledningen att modellen tränats på simulerad data och det finns onekligen skillnader mellan simulerad och experimentell data. Dessutom blir storleksbestämningen på den experimentella datan något sämre eftersom det finns svårigheter i att simulera mikroskopets komplexa point spread function (PSF) med hög precision.

Den experimentella datan antogs bestå av partiklar med samma radie (220 nm) och brytningsindex (1,58). Nätverket förutsäger däremot en lite bredare distribution runt dessa värden och verkar ha detekterat andra partiklar med annorlunda radie och brytningsindex. Nätverkets prestanda på radie och brytningsindex skulle dessutom kunna förbättras genom att medelvärdesbilda deras värden över flera bilder i en film, då experimentell data oftast består av en film över partiklarna.

Vidare skulle nätverket kunna tränas för att hantera ett ännu större antal partiklar. En förbättring av den simulerade datan för att ytterligare efterlikna experimentella förhållanden skulle även kunna minska skillnaden i resultaten på den simulerade datan och experimentell data.

5.3 Utvärdering av modellens resultat jämfört med andra metoder

Vi kan från avsnitt 4.4. se att vår modell generellt sett är betydligt bättre på att framförallt lokalisera partiklar utifrån holografiska bilder jämfört med OAT-algoritmen. När det kommer till karakterisering fungerar modellen inte lika väl, men med en osäkerhet i alla initiala parametrar till Mie-anpassningen lyckades vår modell bättre med att bestämma radien, men något sämre för brytningsindex. Dock är det inte en helt rättvis jämförelse eftersom det finns många parametrar inom både OAT-algoritmen och Mie-Fitting som kan justeras för att uppnå ännu bättre resultat. Eftersom vår modell utvecklats till att vara en snabb och effektiv metod, utan större krav på anpassning av diverse parametrar, jämfördes metoderna med deras ursprungliga inställningar. Vår modell skulle givitvis också gå att justera för att prestera ännu bättre inom vissa specifika områden, men målet var att konstruera en relativt generell modell som fungerar för flera olika typer av partiklar och bilder.

5.3.1 Lokalisering

En av de största nackdelarna med de flesta traditionella metoder är att de har svårigheter att lokalisera fler än en partikel per bild, och även om OAT-algoritmen kunde identifiera fler än en minskade dess recall snabbt när antalet partiklar i bilden ökade, se figur 4.3. Redan vid 5 partiklar var dess recall under 0,2, medan vår modell behöll en recall över 0,8 även vid 20 partiklar i bilden. Detta är en stor förbättring och går säkert att utveckla ännu mer genom att träna modellen med högre partikeldensitet. Eftersom bilderna har en begränsad yta (i detta fall 448x448 pixlar) kommer partiklarnas ringmönster dock oundvikligt börja överlappa varandra.

Speciellt överlappningen av ringmönstrerna var en stor svårighet för OAT-algoritmen (se figur 4.4). Mest intressant är det att studera kurvan för recall för de två metoderna. De börjar inledningvis båda på 0,5, vilket innebär att de ser de två partiklarna som en. När avståndet mellan de två partiklarna sedan ökar, börjar vår modell kunna identifiera båda

två, medan OAT-algoritmen istället inte lyckas hitta någon. Detta beror troligtvis på att partiklarnas spridda ljus interfererar och de karaktäristiska ringarna runt partiklarna förvrängs. Då blir resultatet av den inledande cirkulär-transformen inte alls samma, eftersom det cirkulära mönstret inte längre är lika tydligt. Detta är ett fenomen som vår modell inte verkar ha några större svårigheter med.

Vi har också påvisat att vår modell hanterar brus bättre än traditionella metoder, speciellt för lokalisering av partiklar, se figur 4.1. Även om bruset ibland är så högt att vi inte själva kan identifiera partiklarna från bilderna lyckas modellen hitta dem. Dock kan vi se till vänster i samma figur att precision vid lägre brus är något lägre inledningsvis. Detta beror på att vår modell tenderar att göra många FP förutsägelser vid väldigt lågt brus. Genom att höja ρ en aning i post-processeringen, se 3.9, skulle detta motverkas, men det kan istället leda till en något sämre recall. Eftersom det i verkligheten är väldigt ovanligt med sådant lågt brus i bilderna kommer en lägre precision vid lågt brus inte att ha en så stor betydelse i många tillämpningsområden.

I avsnitt 4.5 såg vi också att vår modell uppnår liknande eller bättre resultat jämfört med djupinlärnings-modellen CATCH. De rapporterade något högre recall och skrev också att deras modell inte rapporterade några FP vilket är imponerande. Dock är det, precis som tidigare sagt, svårt att göra en rättvis jämförelse eftersom de simulerade bilderna inte är samma, och de applicerade endast 5 % additivt gaussiskt brus på sina ideala holografibilder. Vi simulerade istället bilder med en varierande brusnivå och hade därför bilder med både lägre men även betydligt mer brus. Dessutom lade vi till strukturellt brus i våra simuleringar och hade en något högre partikeldensitet. Därför är det inte mer än rimligt att våra värden på precision och recall är något lägre, och det skulle varit intressant att jämföra dessa två modeller sida vid sida med samma data.

Eftersom den slutgiltiga tillämpningen handlar om att spåra partiklarnas rörelse, så gör det ingenting om precision är något lägre, så länge man inte missar några partiklar, d.v.s. har hög recall. Om modellen skulle göra en false-positive i en uppskattning, så kommer den troligtvis inte göra samma fel i nästa bild. Då kommer inte den falska uppskattningen kunna paras ihop med en existerande partikel och kan därför filtreras bort i post-processeringen.

5.3.2 Karakterisering

Resultatet för karakteriseringen var inte lika lovande som lokaliseringen, och modellen hade speciellt svårt att uppskatta z-positionen. Detta är dock väntat då detta är ett väldigt komplex problem, men jämförelsevis uppnådde vi ungefär motsvarande resultat i uppskattningarna som CATCH på både simulerade bilder och experimentdata.

När vi jämför karakteriseringen med den traditionella metoden Mie-anpassning på simulerad data så noterar vi (i figur 4.5) att vår modell tenderade att överskatta brytningsindex för partikeln. Detta beror troligtvis på att dessa bilder skiljer sig från träningsdatan. För att kunna göra en bra Mie-anpassning så simulerades partiklarna längre ifrån fokus, d.v.s. större z-koordinat. Detta resulterar i att partikelns ringmöster blir större i den rekonstruerade bilden (vilket underlättar Mie-anpassning), och det kan hända att vår modell har förknippat denna storlek med brytningsindexet, eller helt enkelt inte presterar lika bra så långt ifrån fokus. Så länge metoden är konsekvent i det fel som introduceras vid olika parametrar så gör det dock inte så mycket att det absoluta värdet inte är exakt, eftersom det då går att kalibrera för detta. Därför är det många gånger mer intressant att studera

spridningen av uppskattningarna, se tabell 4.4. I den tabellen ser vi att vår modell lyckas bestämma radien mer konsekvent än den naiva Mie-anpassningen, men är något sämre på brytningsindex.

5.4 Vidareutveckling för simulering av träningsdata

Som tidigare nämnts i kapitel 3, simuleras partikelbilderna med Deeptrack och superponeras med strukturellt brus som genereras med en separat funktion. Därefter augmenteras bilderna innan de skickas in i modellen. Områden för utveckling och förbättringar för simuleringen av datan ligger framförallt i processeringen av det strukturella brus samt augmenteringar. Vidare föreslår vi hur simuleringen av data skulle kunna utvecklas för att representera en större variation av experimentbilder.

Formen på det strukturella bruset motiverades i avsnitt 3.4 vara tillräckligt likt det verkliga bruset på lokal nivå men skiljer sig i bildens helhet då vågmönstret i verkligheten förekommer irreguljärt medan det implementerade bruset är sinusformad. För att återskapa det irreguljära vågmönstret behöver den matematiska definitionen i ekvation (3.1) modifieras. Exempelvis kan brusparametrarna A, γ, f variera som funktion av det radiella avstånd R_s från bruskällan. Detta kommer att erhålla en mer verklig brusmiljö för de simulerade bilderna men nätverkets prestanda kommer troligtvis ej förbättras signifikant av detta.

Under träningen uppkom vissa problem med kompatibiliteten med de augmentationsmetoder vi använde vilket försämrade både modellens träningstid och positionsbestämning. Vid framtida träning av modeller för partikelspårning rekommenderas därför tillägget av en separat pythonklass för augmentationer som är kompatibel med den valda träningsprocessen där de simulerade bilderna augmenteras just innan en träningsiteration.

För framtida tillämpningar och för att förbättra hur väl den simulerade och experimentella datan överensstämmer tror vi vidare att ett större intervall för z-positionen och radien skulle vara gynnsamt. Man skulle också kunna överväga att simulera partiklar som har de mest förekommande kombinationerna av radie och brytningsindex som tillämpningen ska studera. En större variation på träningsdatan skulle även minska risken för överanpassning i modellen.

5.5 Experiment med modellen som inte fungerade

Under träningen av modellen utförde vi flera experiment som inte bidrog till förbättrade resultat. Vi tar upp de viktigaste nedan.

5.5.1 Ankare med både z-koordinat och radien

Målet var att kunna bestämma z-position, radie och brytningsindex i ett steg utan att vara beroende av att först bestämma partikelns utbredning likt [30]. Initialt ersatte vi bredden och höjden i ankarna med z och radien eftersom båda har en viss korrelation med hur partikeln ser ut. Vi fick dock höga koordinatfel på både radien och z-positionen med denna metod. Detta beror troligtvis på att det inte finns en korrelation mellan partikelns z-position och radie för en viss partikel i datan vilket är fallet för objektets höjd och bredd i de ursprungliga ankarna i YOLOv3 [22].

5.5.2 Beräkna radiens kostnad med ankare

Den initiala kostnadsfunktion för radien skalade modellens output genom

$$b_r = p_r \exp(t_r),$$

vilket motsvarar skalningen av bredden och höjden i YOLOv3. Ankarna, p_r, har värden mellan 100 och 300 nm. Detta gav dock väldigt instabila radiebestämmelser vilket vi tror beror på att det innebar att intervallet för korrekta värden t_r blev litet i jämförelse med $x-$,$y-$,$z-$, $\eta-$koordinaterna. Exempelvis kunde ett fel av magnitud 10^{-3} i t_r ge ett relativt fel på över 10 för radiebestämmelsen. Detta gjorde radien till en svårare uppgift att bestämma än de andra koordinaterna. Vi testade olika lösningar till detta och använde slutligen

$$b_r = 10^{-7} t_r.$$

Det är dock möjligt att det finns andra sätt att normalisera radiebestämningen vilka även inkluderar ankarna.

5.6 Vidareutveckling av modellen

För att förbättra modellens resultat ser vi flera utvecklingsområden för modellens design. Dessa skulle framförallt bidra till att förbättra modellens precision, recall och radie- samt brytningsindexbestämning där vi identifierat vissa begränsningar för modellen i 4.4.2 och 4.5.

5.6.1 Detektionskapacitet

I avsnitt 4 ser vi att vid testning på bilder med högre partikeldensiteten har modellen ett något lägre recall än när densiteten överensstämmer med träningsbilderna. Det indikerar att för optimal prestanda behövs en kongruens mellan tränings- och testbilderna. En ytterligare möjlig anledning till dessa resultat är att även om YOLOv3-modellen kan utföra över 10000 positionsbestämningar i varje bild, innebär modellens design att NMS måste utföras. På det sätt som NMS används nu elimineras partiklar utifrån avståndet i x- och y-position. Det kan resultera i att förutsägelser för partiklar som är nära överlappande riskerar att elimineras. Detta problem kan hanteras på två sätt: designa NMS för att undvika problemet eller utveckla en modell som inte använder NMS.

Den första åtgärden innebär att utvidga NMS-algoritmen från att enbart jämföra två partiklars x- och y-koordinat för att bestämma om två positioner avser samma partikel. Genom att i NMS även ta i åtanke alla partikelns koordinater och jämföra z-koordinaten, radien och brytningindex för partikelpositionen minskas sannolikheten för att korrekta partikelbestämmelser elimineras. Detta bygger på antagandet om att det är sällan två partiklar av samma typ befinner sig väldigt nära varandra i alla rumskoordinater. Under vissa experimentuppställningar stämmer dock inte detta antagande och andra metoder skulle krävas.

Forskningen inom objektdetektering är dock väldigt aktiv och ny utveckling indikerar att modeller i framtiden inte kommer vara beroende av NMS [36]. För tillämpningar inom objektdetektering som hanterar nära överlappande objekt kan därför dessa modeller vara ett bättre alternativ.

5.6.2 Bestämning av radie och brytningsindex

I vår modell är målet att kunna bestämma radie och brytningindex i en end-to-end process. Det är möjligt att ett mindre fel hade kunnat uppnås om man, likt Altman [30], delade in partikelspårningen och karakteriseringen i två olika delar. Detta generaliserar dock sämre till högre partikeldensiteter på grund av att inferenstiden beror på antal partiklar i bilden.

I våra experiment används ankare för partikelradien enbart under träningen av modellen för att bestämma vilka positioner som tränas. YOLOv3 [22] hade dock stor framgång i att använda dessa även för att konvertera modellens output vid inferens enligt (2.4). Ett intressant område vore därför att med utgångspunkt från experimenten i 5.5.2 undersöka metoder för att normalisera modellens koordinatbestämmelser med hjälp av ankarna för att förbättra radiebestämningen. Att bestämma ankare som ett set av radien och brytningsindex på de partikeltyper som tillämpningen behandlar skulle också kunna förbättra modellens karakterisering.

5.6.3 Inferenstid

Eftersom det i projektets begynnelse inte fanns en tilltänkt storlek på bilderna som modellen ska tillämpas på tränades modellen på kvadratiska simulerade bilder av storlek 416-448 pixlar. Under inferens beskär vi sedan experimentbilderna i överlappande segment för att matcha träningsstorleken. I ett specifikt tillämpningsområde skulle inferenstiden kunna minskas om modellen tränades på experimentdatans storlek eftersom färre koordinatbestämmelser skulle utföras av modellen och överlappningar inte skulle krävas. Notera även att inferens utan överlappande beskärningar resulterar i en signifikant minskning av inferenstiden, se tabell 4.3, med endast en liten minskning i recall.

Framtida experiment för att förbättra inferenstiden skulle också kunna undersöka hur modellens prestanda påverkas av modellstorleken. Vår modellen tränades, likt YOLOv3, på att bestämma koordinater på tre olika rutnätsskalor som specialiserats för olika partikltyper efter partikelns radie. Att enbart träna den finaste skalan och att använda ett mindre backbone nätverk som DarkNet-19 är två sätt att minska storleken på modellen men behålla de viktigaste egenskaperna av YOLOv3.

6

Slutsats

Djupinlärningsmetoden YOLOv3 med träning på simulerad data från pythonbiblioteket DeepTrack har i arbetet visat goda resultat för detektering av partiklar. Modellen lyckas uppnå klart bättre resultat jämfört med traditionella algoritmiska metoder (exempelvis [35]) för partikeligenkänning, särskilt för lokalisering av partiklar men något sämre för karakterisering av partiklars egenskaper. Modellen har dessutom visat resultat jämförbara med tidigare djupinlärningsmetoder som exempelvis [30]. Nätverket har även en oavbruten end-to-end-process för partikelspårning och karakterisering till skillnad från flera tidigare maskininlärningsmetoder samt traditionella algoritmer. Det ger en möjlighet att spåra och karakterisera samtliga partiklar i en filmsekvens parallellt. En tydlig fördel med modellen är också att den inte sätter samma krav på användaren för detaljstyrning av arbetsprocessen vilket kan minska arbetsbördan vid framtida spårning av partiklar. Emellertid är djupinlärningsmodeller för spårning av partiklar fortfarande tämligen outforskat. I föreliggande arbete prövades ramverket för en typ av djupinlärningsmodell, YOLOv3, och de ingående parametrarna justerades för att uppnå fullgoda resultat. Det finns dock ingenting som tyder på att modellen inte kan förbättras genom att vidare optimera de ingående parametrarna samt genom träning på större mängder simulerad data.

Vidare kan i framtiden modeller utvecklas för att även fullständigt kategorisera och klassificera alla möjliga partiklar för att understödja arbetet vid alla typer av partikeligenkänning. Även utvidgning av partikelspårning för att hantera ett större spann av storlekar på partiklar förväntas i framtiden kunna förenas i en och samma modell. Arbetet går även i linje med den trend som representerar mer automatiserade arbetsprocesser genom användning av maskininlärning i forskningssammanhang.

Referenser

[1] D. Gabor, "A New Microscopic Principle," *Nature*, vol. 161, nr. 4098, s. 777–778, maj 1948, DOI: 10.1038/161777a0.

[2] M. K. Kim, "Principles and techniques of digital holographic microscopy," *Journal of Photonics for Energy*, s. 018 005, april 2010, DOI: 10.1117/6.0000006.

[3] E. N. Leith och J. Upatnieks, "Reconstructed Wavefronts and Communication Theory*," *Journal of the Optical Society of America*, vol. 52, nr. 10, s. 1123, okt. 1962, DOI: 10.1364/JOSA.52.001123.

[4] ——, "Wavefront Reconstruction with Continuous-Tone Objects*," *Journal of the Optical Society of America*, vol. 53, nr. 12, s. 1377, dec. 1963, DOI: 10.1364/JOSA.53.001377.

[5] ——, "Wavefront Reconstruction with Diffused Illumination and Three-Dimensional Objects*," *Journal of the Optical Society of America*, vol. 54, nr. 11, s. 1295, nov. 1964, DOI: 10.1364/JOSA.54.001295.

[6] U. Schnars och W. Jüptner, "Direct recording of holograms by a CCD target and numerical reconstruction," *Applied Optics*, vol. 33, nr. 2, s. 179, jan. 1994, DOI: 10.1364/AO.33.000179.

[7] T. Tahara, X. Quan, R. Otani, Y. Takaki och O. Matoba, "Digital holography and its multidimensional imaging applications: a review," *Microscopy*, vol. 67, nr. 2, s. 55–67, april 2018, DOI: 10.1093/jmicro/dfy007.

[8] T. Noda, S. Kawata och S. Minami, "Three-dimensional phase-contrast imaging by a computed-tomography microscope," *Applied Optics*, vol. 31, nr. 5, s. 670, febr. 1992, DOI: 10.1364/AO.31.000670.

[9] C. J. Mann, L. Yu, C.-M. Lo och M. K. Kim, "High-resolution quantitative phase-contrast microscopy by digital holography," *Optics Express*, vol. 13, nr. 22, s. 8693, 2005, DOI: 10.1364/OPEX.13.008693.

[10] B. Rappaz, A. Barbul, A. Hoffmann m. fl., "Spatial analysis of erythrocyte membrane fluctuations by digital holographic microscopy," *Blood Cells, Molecules, and Diseases*, vol. 42, nr. 3, s. 228–232, maj 2009, DOI: 10.1016/j.bcmd.2009.01.018.

[11] B. Kemper och G. von Bally, "Digital holographic microscopy for live cell applications and technical inspection," *Applied Optics*, vol. 47, nr. 4, A52, febr. 2008, DOI: 10.1364/AO.47.000A52.

[12] P. Marquet, B. Rappaz, P. J. Magistretti m. fl., "Digital holographic microscopy: a noninvasive contrast imaging technique allowing quantitative visualization of living cells with subwavelength axial accuracy," *Optics Letters*, vol. 30, nr. 5, s. 468, mars 2005, DOI: 10.1364/OL.30.000468.

[13] K. D. Hinsch, "Three-dimensional particle velocimetry," *Measurement Science and Technology*, vol. 6, nr. 6, s. 742–753, juni 1995, DOI: 10.1088/0957-0233/6/6/012.

[14] E. Talpes, D. D. Sarma, G. Venkataramanan m. fl., "Compute Solution for Tesla's Full Self-Driving Computer," *IEEE Micro*, vol. 40, nr. 2, s. 25–35, mars 2020, DOI: 10.1109/MM.2020.2975764.

[15] Q. Hu, "Evaluation of Deep Learning Models for Kannada Handwritten Digit Recognition," i *2020 International Conference on Computing and Data Science (CDS)*, Stanford, CA, USA: IEEE, aug. 2020, s. 125–130. DOI: 10.1109/CDS49703.2020.00031.

[16] P. Wang, "Research and Design of Smart Home Speech Recognition System Based on Deep Learning," i *2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL)*, Chongqing, China: IEEE, juli 2020, s. 218–221. DOI: 10.1109/CVIDL51233.2020.00-98.

[17] Z.-Q. Zhao, P. Zheng, S.-T. Xu och X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, nr. 11, s. 3212–3232, nov. 2019, DOI: 10.1109/TNNLS.2018.2876865.

[18] L. Liu, W. Ouyang, X. Wang m. fl., "Deep Learning for Generic Object Detection: A Survey," *arXiv:1809.02165*, aug. 2019.

[19] A. R. Pathak, M. Pandey och S. Rautaray, "Application of Deep Learning for Object Detection," *Procedia Computer Science*, vol. 132, s. 1706–1717, 2018, DOI: 10.1016/j.procs.2018.05.144.

[20] R. Girshick, J. Donahue, T. Darrell och J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *arXiv:1311.2524*, okt. 2014.

[21] J. Redmon, S. Divvala, R. Girshick och A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *arXiv:1506.02640*, maj 2016.

[22] J. Redmon och A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv:1804.02767*, april 2018.

[23] B. Mehlig, "Machine learning with neural networks," *arXiv:1901.05639*, febr. 2021.

[24] S. Ren, K. He, R. Girshick och J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *arXiv:1506.01497*, jan. 2016.

[25] J. Redmon och A. Farhadi, "YOLO9000: Better, Faster, Stronger," *arXiv:1612.08242*, dec. 2016.

[26] O. Russakovsky, J. Deng, H. Su m. fl., "ImageNet Large Scale Visual Recognition Challenge," *arXiv:1409.0575*, jan. 2015.

[27] T.-Y. Lin, M. Maire, S. Belongie m. fl., "Microsoft COCO: Common Objects in Context," *arXiv:1405.0312*, febr. 2015.

[28] C. F. Bohren och D. R. Huffman, *Absorption and scattering of light by small particles*. Weinheim, Tyskland: Wiley-VCH, 2004.

[29] M. Cappelezzo, C. A. Capellari, S. H. Pezzin och L. A. F. Coelho, "Stokes-Einstein relation for pure simple fluids," *The Journal of Chemical Physics*, vol. 126, nr. 22, s. 224 516, juni 2007, DOI: 10.1063/1.2738063.

[30] L. E. Altman och D. G. Grier, "CATCH: Characterizing and Tracking Colloids Holographically Using Deep Neural Networks," *The Journal of Physical Chemistry B*, acs.jpcb.9b10463, febr. 2020, DOI: 10.1021/acs.jpcb.9b10463.

[31] S. Helgadottir, A. Argun och G. Volpe, "Digital video microscopy enhanced by deep learning," *arXiv:1812.02653*, dec. 2018.

[32] B. Midtvedt, S. Helgadottir, A. Argun m. fl., "Quantitative digital microscopy with deep learning," *Applied Physics Reviews*, vol. 8, nr. 1, s. 011 310, mars 2021, DOI: 10.1063/5.0034891.

[33] H. W. Moyses, B. J. Krishnatreya och D. G. Grier, "Robustness of Lorenz-Mie microscopy against defects in illumination," *Optics Express*, vol. 21, nr. 5, s. 5968, mars 2013, DOI: 10.1364/OE.21.005968.

[34] R. Alexander, B. Leahy och V. N. Manoharan, "Precise measurements in digital holographic microscopy by modeling the optical train," *Journal of Applied Physics*, vol. 128, nr. 6, s. 060 902, aug. 2020, DOI: 10.1063/5.0015976.

[35] B. J. Krishnatreya och D. G. Grier, "Fast feature identification for holographic tracking: the orientation alignment transform," *Optics Express*, vol. 22, nr. 11, s. 12 773, juni 2014, DOI: 10.1364/OE.22.012773.

[36] N. Carion, F. Massa, G. Synnaeve m. fl., "End-to-End Object Detection with Transformers," *arXiv:2005.12872*, maj 2020.

A

Appendix

A.1 Algoritm för Non-maximum Suppression

Algorithm 1: Non-maximum Suppression (comparisons by IOU)

Input: - $\mathcal{B}$ detections sorted in ascending order by object score b_o
Result: $\mathcal{D}$ - the detections left after NMS
$\mathcal{D} = []$;
while $\mathcal{B}$ **do**
    c = $\mathcal{B}$.pop();
    for *box in* $\mathcal{B}$ **do**
        if *IOU(box, c)* $> \epsilon$ **then**
            $\mathcal{B}$.remove(box)
        end
    end
    $\mathcal{D}$.append(c)
end

Fysikinstitutionen
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige

CHALMERS