YRKESHÖGSKOLAN
NOVIA

Jämförelse av kameraprogram med djupinlärning

Axel Johansson

Examensarbete för ingenjörs (YH)-examen

El- och automationsteknik

Vasa 2022

EXAMENSARBETE

Författare: Axel Johansson

Utbildning och ort: El- och automationsteknik, Vasa

Inriktning: Automation

Handledare: Kaj Wikman, Johnny Backlund (Prevex)

Titel: Jämförelse av kameraprogram med djupinlärning

Datum: 20.5.2022 Sidantal: 39

Abstrakt

Detta examensarbete är en jämförelse och ett test av två olika kameraprogram som använder sig av djupinlärning för att granska bilder.

Syftet med examensarbetet var att ta reda på om det med hjälp av dessa program enkelt skulle gå att granska en produkt där det förekommer varierande defekter. Man hade tidigare konstaterat att det skulle vara väldigt svårt och ta lång tid att fånga dessa defekter med en vanlig smartkamera, så uppgiften blev då att testa två djupinlärningslösningar från företagen MVTec och Cognex. Från MVTec så testades Halcon i samband med Deep Learning Tool och från Cognex så testades In-Sight Vision Suite där djupinlärningsprogrammet In-Sight ViDi fanns integrerat.

Arbetets praktiska del gick ut på att ta i bruk de olika programmen och jämföra båda tillverkarnas lösningar. Saker som jämförs är programmens djupinlärningsmetoder, svårighetsgraden att få i gång ett system, precision, hur små variationer som kan detekteras och flexibiliteten för användaren av programmet. Målet var dock inte att skapa några helt fungerande system, utan att utvärdera programmen och deras egenskaper för att se ifall de skulle lämpa sig för granskningen av en viss produkt på företaget Prevex.

Resultatet blev en kortfattad introduktion till djupinlärning, samt dokumentation av testningen av kameraprogrammen och en jämförelse mellan programmen.

Språk: svenska
Nyckelord: djupinlärning, maskinseende, Halcon, Cognex, kvalitetskontroll

OPINNÄYTETYÖ

Tekijä: Axel Johansson
Koulutus ja paikkakunta: Sähkö- ja automaatiotekniikka, Vaasa
Suuntautumisvaihtoehto: Automaatio
Ohjaaja(t): Kaj Wikman, Johnny Backlund (Prevex)

Nimike: Kameraohjelmiston vertailu syvään oppimiseen

Päivämäärä: 20.5.2022 Sivumäärä: 39

Tiivistelmä

Tämä opinnäytetyö on kahden erilaisen kameraohjelman vertailu ja testi, jotka käyttävät syväoppimista kuvien tarkastelussa.

Opinnäytetyön tarkoituksena oli selvittää, olisiko näillä ohjelmilla mahdollista helposti tarkastaa tuote, jossa on erilaisia vikoja. Aiemmin oli todettu, että näiden vikojen havaitseminen tavallisella älykameralla olisi erittäin vaikeaa ja kestää kauan, joten tehtäväksi tuli sitten testata kahta syväoppimisratkaisua MVTec- ja Cognex-yhtiöiltä. MVTecistä Halconia testattiin Deep Learning Toolin yhteydessä ja Cognexista In-Sight Vision Suitea, johon integroitiin syväoppimisohjelma In-Sight ViDi.

Työn käytännön osa sisälsi eri ohjelmien käyttöönottoa ja molempien valmistajien ratkaisujen vertailua. Verrattavia asioita ovat ohjelmien syväoppimismenetelmät, järjestelmän käyttöönoton vaikeus, tarkkuus, pienten vaihteluiden havaitseminen ja joustavuus ohjelman käyttäjän kannalta. Tavoitteena ei kuitenkaan ollut luoda täysin toimivia järjestelmiä, vaan arvioida ohjelmia ja niiden ominaisuuksia ja katsoa, soveltuuko ne Prevex-yhtiön tietyn tuotteen arviointiin.

Tuloksena oli lyhyt johdatus syvään oppimiseen sekä kameraohjelmien testauksen dokumentointi ja ohjelmien vertailu.

Kieli: ruotsi
Avainsanat: syväoppiminen, konenäkö, Halcon, Cognex, laadunvalvonta

BACHELOR'S THESIS

Author: Axel Johansson
Degree Programme: Electricity and automation technology, Vaasa
Specialisation: Automation
Supervisor(s): Kaj Wikman, Johnny Backlund (Prevex)

Title: Comparison of camera software with deep learning

Date: 22.5.2022 Number of pages: 39

Abstract

This thesis is a comparison and a test of two different camera programs that use deep learning to examine images.

The purpose of the thesis was to find out if it would be possible to easily inspect a product with various defects using these programs. It had previously been established that it would be very difficult and take a long time to catch these defects with a regular smart camera, so the task then became to test two deep learning solutions from the companies MVTec and Cognex. From MVTec, Halcon was tested along with the Deep Learning Tool and from Cognex, the In-Sight Vision Suite was tested where the deep learning program In-Sight ViDi was integrated.

The practical part of the work involved putting the various programs into use and comparing both manufacturers' solutions. Things that are compared are the programs' deep learning methods, the difficulty of getting a system up and running, precision, how small variations can be detected, and the flexibility for the user of the program. However, the goal was not to create any fully functional systems, but to evaluate the programs and their characteristics to see if they would be suitable for the review of a certain product at the company Prevex.

The result was a brief introduction to deep learning, as well as documentation of the testing of the camera programs and a comparison between the programs.

Language: Swedish
Key words: deep learning, machine vision, Halcon, Cognex, quality control

Innehållsförteckning

Ordlista

Djupinlärning	En maskininlärningsteknik som använder sig av neurala nätverk för att lära sig själv via exempel på liknande sätt som människor lär sig.
Artificiell intelligens	Datorprogram med förmågor att efterlikna mänsklig intelligens.
Maskininlärning	En process som hör till kategorin artificiell intelligens som gör det möjligt för datorer att lära sig något utan att uttryckligen ha blivit programmerad till det.
HMI	Människa-maskingränssnitt i form av en operatörspanel.
Confusion matrix	En matris som inom området maskininlärning används för att visa resultatet hur träningen av en algoritm har gått.

1 Inledning

Detta examensarbete handlar om att jämföra och testa två kameraprogram med maskinseende som använder sig av djupinlärning för att granska bilder. Djupinlärning är en teknik som idag används mera och mera i samband med kameror när man till exempel vill granska eller sortera produkter. MVTec och Cognex är företag som de senaste åren har börjat integrera djupinlärning i sina kameraprogram och de är sedan tidigare kända för sina program som använder sig av maskinseende, därför valdes deras nya djupinlärningsprogram för testning.

Examensarbetet utfördes åt Prevex Oy Ab som i huvudsak tillverkar produkter i plast med hjälp av formsprutningsteknik. På vissa av produkterna kan det förekomma varierande defekter, som det med traditionella smartkameror skulle vara väldigt svårt och tidskrävande att skapa program för som skulle hitta alla dessa defekter. Målet med detta examensarbete är således att testa de två programmen på en specifik produkt för att se om man med hjälp av djupinlärning kan kvalitetsgranska den produkten och att samtidigt jämföra de två tidigare nämnda tillverkarnas olika lösningar.

1.1 Bakgrund

Prevex specialiserar sig på tillverkning av vattenlås och när man tillverkar produkter som fylls med vatten så vill man säkerställa att produkterna inte kommer att läcka. Hos företaget finns sedan tidigare en hel del kvalitetsgranskning med hjälp av så kallade smartkameror som använder sig av Machine Learning-teknik. Nackdelen med dessa kameror är att man måste programmera eller skapa program för dessa kameror, vilket kan ta väldigt lång tid och bli aningen komplicerat om man vill granska mer än en typ av defekt på flera områden på en produkt. Detta har lett till att produkter som inte går att granska med vanliga smartkameror så granskas i stället för hand av personalen.

På automationsavdelningen på företaget hade man tidigare hört om kameraprogram som använder sig av djupinlärning, som annonserades som program som skulle vara lätta att ta i bruk och programmen skulle enkelt hitta defekter av olika slag, men ingen på företaget hade haft tid att testa något av dessa program. Eftersom man på företaget

gärna vill bli av med den manuella granskningen av produkter så erbjöds möjligheten att undersöka några av dessa program som ett examensarbete.

1.2 Syfte

Syftet med arbetet var att undersöka programmen från MVTec och Cognex för att få en uppfattning om hur dessa program fungerar och ifall de går att använda till mera komplicerad kvalitetsgranskning. Programmen skall också jämföras med varandra och egenskaper som kommer att jämföras är bland annat komplexitet, precision, noggrannhet och flexibilitet.

Prevex tillverkar skalet till Mirkas DEROS slipmaskin och det är skalet till just den som har valts som testobjekt för djupinlärningsprogrammen. På skalet kan det uppstå små gropar i det svarta gummit efter formsprutningsprocessen och dessa gropar anses vara defekter. På skalet stämplas även en text på två olika ställen, när texten stämplas så kan detta leda till att produkten anses vara defekt ifall färgen från stämpeln börjar rinna eller om en stämpling utförs två gånger på samma ställe och texten blir för tjock eller suddig.

Figur 1. Mirka DEROS slipmaskin. (MIRKA Mirka DEROS 550CV 125mm pölynpoisto 5,0, n.d.).

1.3 Prevex

Prevex är ett företag som specialiserar sig på tillverkning av vattenlås till kök och badrum, men även tillverkning av andra produkter i plast görs på företaget. Prevex är en del av KWH Group och företaget är baserat i Nykarleby, Jakobstad och även i Poznań i Polen.

Produkterna tillverkas med hjälp av formsprutningsteknik och en stor del av produkterna monteras och packas automatiskt med hjälp av robotar och annan automation. Årligen producerar företaget över 4.4 miljoner vattenlås och omsätter ca. 32 miljoner euro. På Prevex fanns år 2019 sammanlagt 259 anställda i Finland och Polen.

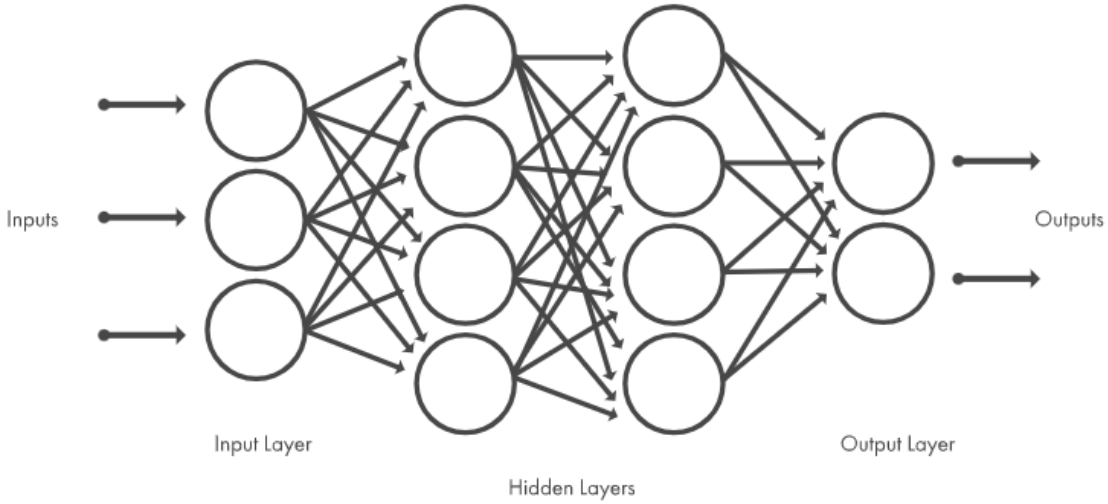
Hållbarhet ligger väldigt högt i fokus på företaget och det syns genom åtgärder som till exempel byte till användning av 100 % förnybar elektricitet och nyligen har även en stor del av vattenlåstillverkningen övergått till användning av återvunnen plast. På företaget finns över 50 robotar i produktionen och satsningen på automation ökar ständigt, många manuella processer automatiseras och mera automationslösningar införskaffas ständigt till företaget. (PREVEX About us, u.d.; KWH GROUP Prevex, 2019).

Figur 1. Prevex logo. (PREVEX About us, n.d.).

2 Deep Learning

Djupinlärning är en typ av maskininlärning som använder sig av neurala nätverk. Med traditionella kameraprogram som använder maskininlärning så måste användaren välja ut vilka sorters fel som programmet ska söka efter och var det ska söka, men med djupinlärning behövs endast bilder som är sorterade och märkta, sedan lär sig det neurala nätverket bildernas egenskaper och kan själv till exempel plocka ut felen eller sortera

bilderna. Djupinlärningen fungerar alltså mera som inlärningen för en människa gör, man lär sig saker genom att se olika exempel och med djupinlärning fungerar det ungefär likadant. Orsaken att djupinlärning kallas för just djupinlärning är för att de neurala nätverken som används kan innehålla upp till 150 gömda lager, jämfört med traditionella neurala nätverk som endast använder två till tre gömda lager. (MATHWORKS What Is Deep Learning, n.d.).



Figur 3. Neurala nätverk. (MATHWORKS What Is Deep Learning, n.d.).

Djupinlärning används idag inom många områden, till exempel röststyrning och självkörande bilar, men djupinlärningen har även nått kvalitetsgranskningen som är ett väldigt viktigt område i alla fabriker. För att man ska kunna lita på att djupinlärningsbaserade kameraprogram fungerar lika bra eller bättre än det mänskliga ögat så krävs många exempelbilder och ordentlig testning. Fördelen med djupinlärningen är att det krävs mycket mindre arbete för att skapa ett program som kontrollerar en produkt, man slipper den krångliga programmeringen som behövs med traditionella program med maskinseende och detta gör att vilken ingenjör som helst kan skapa ett program med djupinlärning. (COGNEX Deep learning solutions, n.d.).

2.1 Vanliga metoder för granskning

När man granskar en eller flera produkter med djupinlärning så erbjuder de flesta tillverkarna av kameraprogrammen olika metoder eller verktyg som man kan utföra granskningen med. Exempel på dessa metoder är:

- Lokalisering
- Analysering
- Klassificering
- Avläsning

Med lokalisering kan man söka efter ett eller flera föremål i bilden som granskas, metoden kan användas till exempel för att identifiera objekt och sortera eller jämföra dem.

Analyseringen kan användas till att söka efter defekter och med denna metod så ger man exempelbilder med defekter och utan defekter till programmet så att det lär sig att identifiera defekterna. Denna metod används oftast när man vill ha ett godkänt eller icke godkänt resultat från analyseringen och lämpar sig därför väl till kvalitetsgranskning.

Med klassificeringen lär sig programmet att klassificera olika föremål som finns i en bild beroende på föremålets egenskaper och även defekterna kan klassificeras separat, beroende på vilken typ av defekt det är.

Djupinlärningen möjliggör också avläsning av texter från olika föremål på ett väldigt enkelt sätt, vanligtvis används invecklade OCR-algoritmer för att en maskin ska kunna avläsa text från en bild, men med denna teknik kan programmet även avläsa mera otydliga texter som traditionella metoder skulle ha svårt med. (COGNEX In-sight D900 tools, n.d.).

2.2 MVTec

Den första lösningen som testas i detta examensarbete är skapad av företaget MVTec. Företaget grundades år 1996 och är baserat i München i Tyskland. Det är en leverantör av mjukvara för maskinseende och de har länge varit en drivande kraft för utvecklingen inom branschen. Företaget satsar hela tiden på att bli mera internationellt och de säger att goda relationer till kunderna och att lyssna på kunderna är väldigt viktigt eftersom det har bidragit till företagets framgång. (MVTec Company, n.d.).

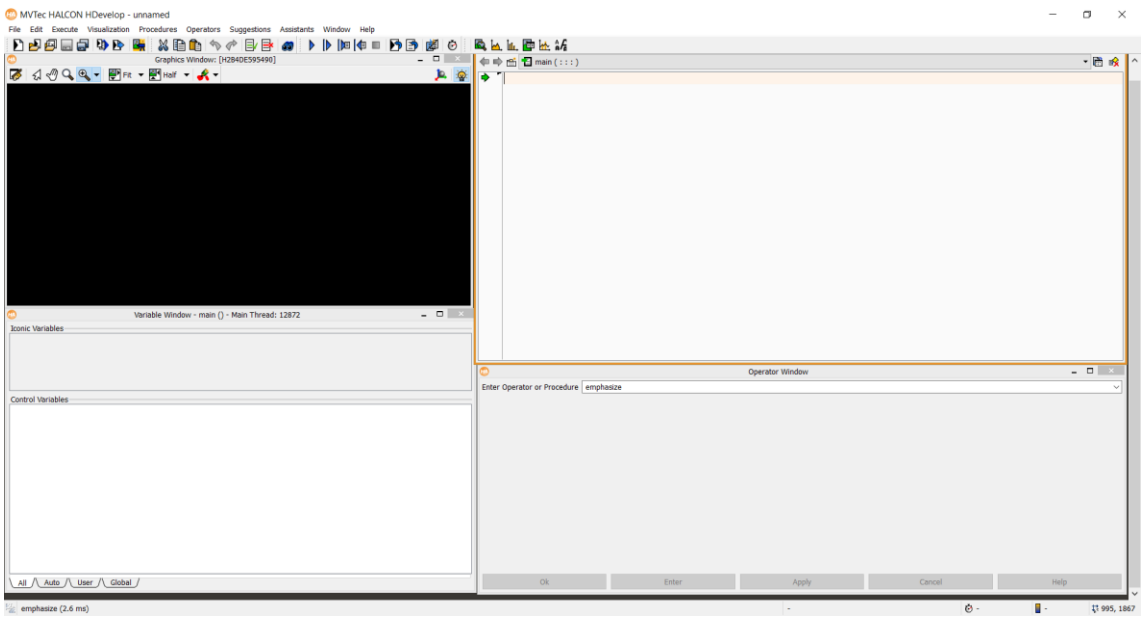
För tillfället erbjuder MVTec två olika fullständiga program för maskinseende, Halcon och Merlic, av vilka Halcon är det mera avancerade programmet och Merlic annonseras som ett mera lättare program att använda eftersom ingen programmering behövs. Förutom dessa så finns det relativt nya tilläggsprogrammet Deep Learning Tool som är avsett för att användas tillsammans med Halcon eller Merlic för att göra det lättare att tillämpa djupinlärning till de två programmen.

2.2.1 Halcon

Halcon är den kraftfullaste mjukvaran som MVTec erbjuder och den sägs ska klara av alla typer av applikationer med maskinseende som kan tänkas behövas. Programmet är PC-baserat men kan också köras på inbäddade system som till exempel Arm®-baserade plattformar. Halcon stöder bland annat GenICam, GigE Vision, Video4Linux och USB3 Vision och kan därför användas med många olika kameror, vilket gör programmet väldigt flexibelt eftersom man inte är låst till en specifik kamera. (MVTec Halcon, n.d.).

Det finns två olika typer av licenser man kan välja, Steady eller Progress, med Steady köper man en version av Halcon för en engångssumma och med Progress betalar man en årsavgift. Steady versionen kan inte uppgraderas, man har den versionen permanent, men med Progress får man uppgradera programmet direkt som en ny uppdaterad version av Halcon släpps. Deep Learning Tool kan laddas ner gratis från MVTecs webbsida eftersom man inte kan skapa ett fullständigt program med verktyget, det krävs att man också har endera Halcon eller Merlic för att man ska ha någon nytta av verktyget.

Själva programmeringsmiljön som används i Halcon heter HDevelop och i figur 4 ser man hur layouten för Halcon HDevelop ser ut.



Figur 4. Halcon HDevelop layout.

Halcon erbjuder många funktioner och några av dessa som MVTec gör reklam för är:

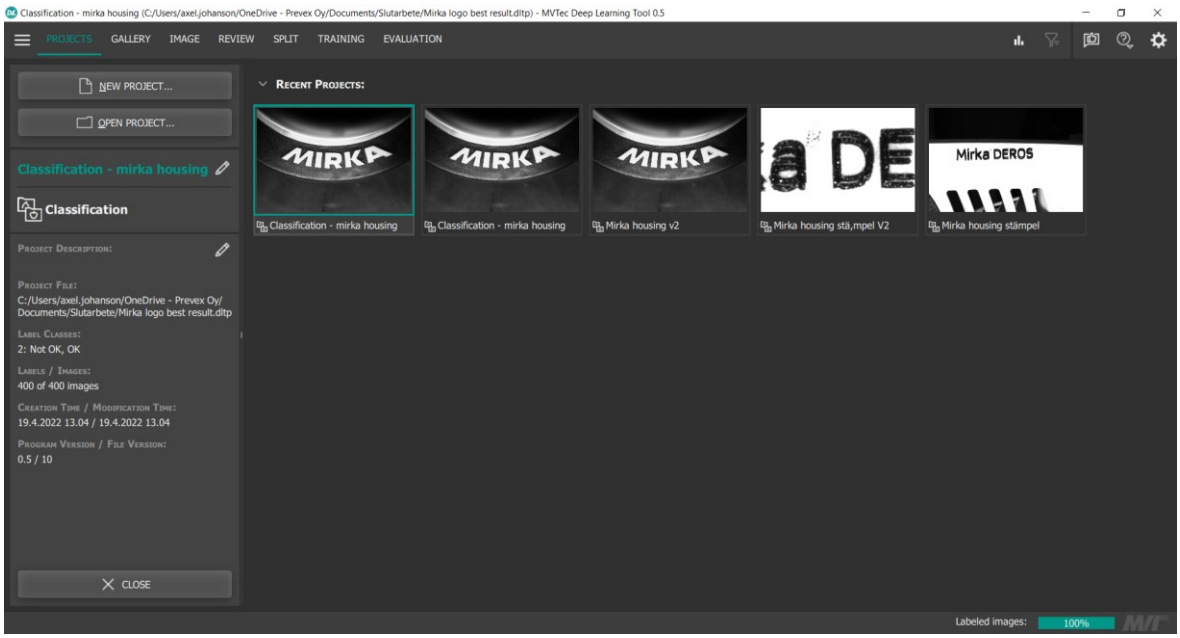
- Revolutionerande mjukvara för 3D-maskinseende.
- Matchning för att upptäcka roterade eller skymda objekt.
- Klumpanalys med mer än 50 form- och gråvärdesfunktioner.
- Mätning med hög noggrannhet.
- Stort utbud av de senaste djupinlärningsteknologierna.
- Optisk teckenigenkänning och verifiering (OCR/OCV).
- Godtyckligt formade regioner av intresse (ROI) för betydande flexibilitet och snabbhet.
- Detektering av linjer, cirklar och ellipser med en noggrannhet på upp till 1/50 pixel.
- Extremt snabb morfologi.
- Färgbildbehandling och hyperspektral avbildning.
- Bearbetning av extremt stora bilder (mer än 32k x 32k).
- Bildsekvensbearbetning.
- Noggrann 3D-kamerakalibrering. (MVTec Why HALCON, n.d.).

Dessutom finns det en hel del exempel avsedda för specifika uppgifter i HDevelop som endera kan användas direkt eller modifieras enligt behov av användaren. Fördelen med

dessa exempel är att man inte behöver skriva all kod från grunden och man lär sig samtidigt hur Halcon fungerar genom att studera exempelkoden och modifiera den själv.

2.2.2 Deep Learning Tool

Deep Learning Tool är ett verktyg som MVTec har utvecklat för att göra det enklare för Halcon och Merlic användare att börja använda djupinlärning. Programmet använder sig av så kallad övervakad inlärning det vill säga bilderna som används för träningen måste först märkas manuellt av användaren, vill man till exempel göra en modell för kvalitetsgranskning så måste exempelbilderna vara märkta som dåliga eller bra bilder. Det enkla med Deep Learning Tool är att ingen programmering alls krävs för att man ska kunna använda detta program och programmet tar användaren stegvist genom processen för att skapa en djupinlärningsmodell som sedan kan användas i till exempel Halcon. (MVTec Deep Learning Tool, n.d.).



Figur 5. Deep Learning Tool.

2.3 Cognex

Cognex grundades år 1981 när amerikanen Dr. Robert J. Shillman, som på den tiden var verksam som föreläsare vid Massachusetts Institute of Technology (MIT), bestämde sig för att ge upp sitt jobb som föreläsare och investera alla sina livsbesparingar i att grunda företaget. Dr. Shillman bjöd in två utexaminerade MIT-studenter att komma och jobba på företaget och det var dessa tre personer, Dr. Shillman, Marilyn Matz och Bill Silver, som kom att skapa grunden för Cognex. Namnet Cognex kommer från uttrycket Cognition Experts.

Året efter att företaget grundades så släpptes det första systemet för maskinseende, DataMan, som var med de första systemen att använda sig av OCR-teknik. OCR står för Optical Character Recognition och med denna teknik kunde systemet läsa och kontrollera kvaliteten av bokstäver och siffror på olika komponenter och delar.

Cognex vision handlar om att hjälpa företag med kvalitetsgranskningen så att man kan eliminera produktionsfelen och minska tillverkningskostnaderna, samt förbättra kvalitén på produkterna. Allt detta vill man erbjuda till ett förmånligt pris så att företagen kan överträffa konsumenternas förväntningar på högkvalitativa produkter.

Företaget börsnoterades år 1989 med ett pris på 1.38$ per aktie, bara ett år senare hade aktiepriset tredubblats och idag (2022) ligger priset per aktie på ca. 45 dollar. (COGNEX Company, n.d.).

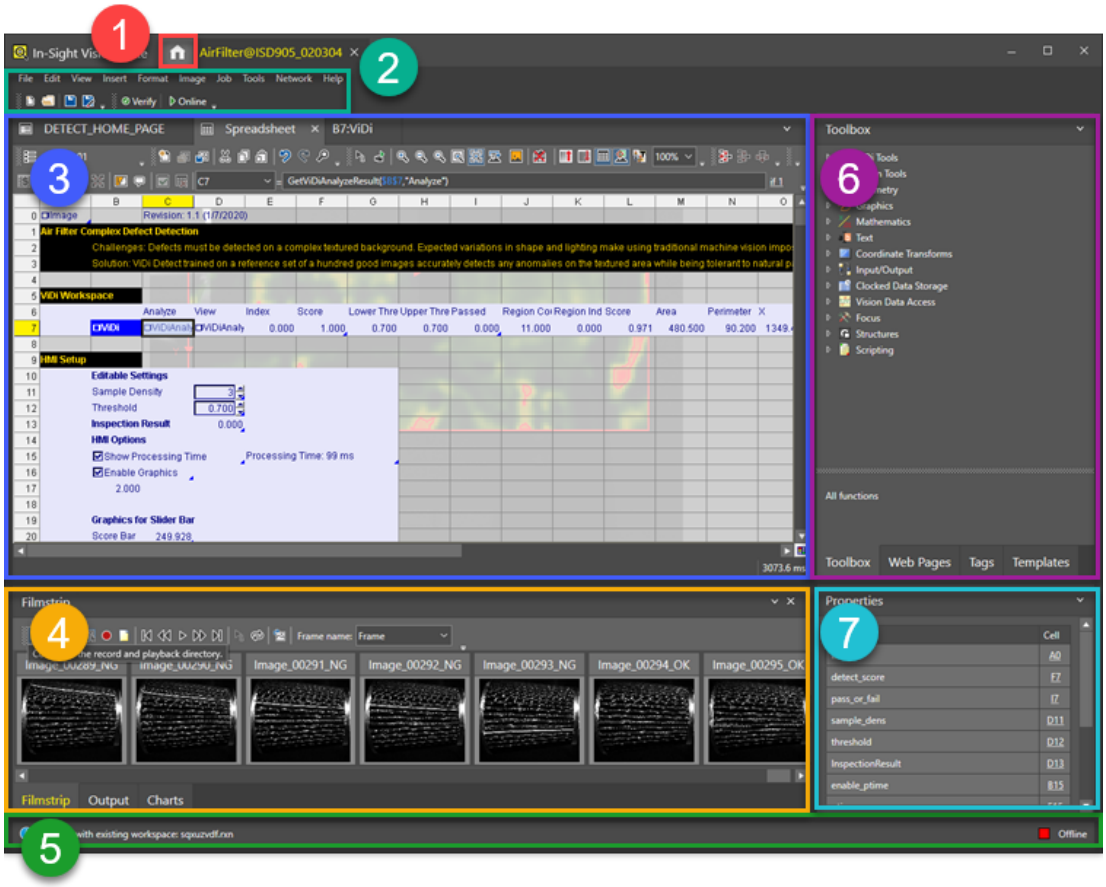
2.3.1 In-Sight ViDi

In-Sight ViDi är ett program gjort av Cognex som tagits fram specifikt för automation i fabriker, programmet är ett djupinlärningsbaserat program för analysering av bilder. Programmet är baserat på toppmoderna maskininlärningsalgoritmer. En Cognex kamera behövs för att man kunna skapa ett helt fungerande system med In-Sight ViDi, men det finns en simuleringsfunktion i programmet som gör att man även kan använda bilder tagna med vilken kamera som helst för att testa programmet och dess funktioner. Cognex har utvecklat en smartkamera, In-Sight D900, som är specifikt ämnad för

djupinlärningsapplikationer och kan därför användas med In-Sight Vidi. (COGNEX Deep learning solutions, n.d.).

Figur 6. In-Sight D900 kamera från Cognex. (IN-SIGHT D900 VISION SYSTEM, n.d.).

Själva skapandet av programmet som ska köras i kameran sker via ett program som heter In-Sight Vision Suite, men efter att man skapat programmet behövs ingen dator kopplad till kameran, utan programmet laddas då över till kameran och körs från den. Inne i In-Sight Vision Suite finns ViDi och en hel del andra verktyg inkluderade, man kan till exempel editera bilder och lägga till filter eller skapa en webbaserad HMI till kameran.



Figur 7. In-Sight ViDi. (COGNEX In-Sight ViDi Help - In-Sight ViDi User Interface, 2022).

I figur 7 ser man hur In-Sight ViDi applikationen ser ut. Applikationen kan delas in i sju olika områden som syns på bilden och funktionerna för dessa områden beskrivs i listan nedan:

1. Hemknappen används för att återvända till In-Sight Vision Suite hemskärmen.
2. Menyfältet där man hittar inställningarna och olika alternativ för programmet.
3. Editeringsfönstret som är ett kalkylblad dit alla verktyg och data som används plockas in.
4. Utmatningsfönstret där man kan visa jobbverifiering, olika grafer eller filmremsan som syns i figur 7.
5. Loggningsfältet där olika meddelanden och varningar visas.
6. Jobbutforskaren där man hittar olika jobbelement som går att lägga till i editeringsfönstret.
7. Egenskapsfönstret där man ser informationen för den jobbkomponenten som för tillfället är markerad.

3 Utförande

Examensarbetet inleddes med att bekanta sig med MVTec-programmet Halcon HDevelop och att använda det programmet för att ta bilder av produkterna som skall granskas i arbetet. När bilderna var tagna fortsatte arbetet med att bekanta sig med och testa MVTec-verktyget Deep Learning Tool som sedan används tillsammans med Halcon. Efter att MVTecs program blivit testade utfördes liknande tester med det Cognex baserade programmet In-Sight ViDi.

I detta kapitel kommer man att få se hur testningen utfördes och olika egenskaper för programmen kommer att utvärderas och senare jämföras i kapitel fyra. Produkten som testas med de olika programmen hade en formsprutad logo och två stämplade texter som man ville granska.

3.1 Testbilder

För att få några testbilder av texten som stämplas på produkterna så skapades ett litet program i Halcon HDevelop för att kunna ta bilder med en kameratestrigg som fanns till förfogande vid fabriken, orsaken till att bilderna togs med just Halcon var för att ingen specifik kamera behövs så länge kameran som används har ett gränssnitt som Halcon stöder, i detta fall var det en kamera med gränssnittet GigE Vision som användes.

Figur 8. Den stämplade texten.

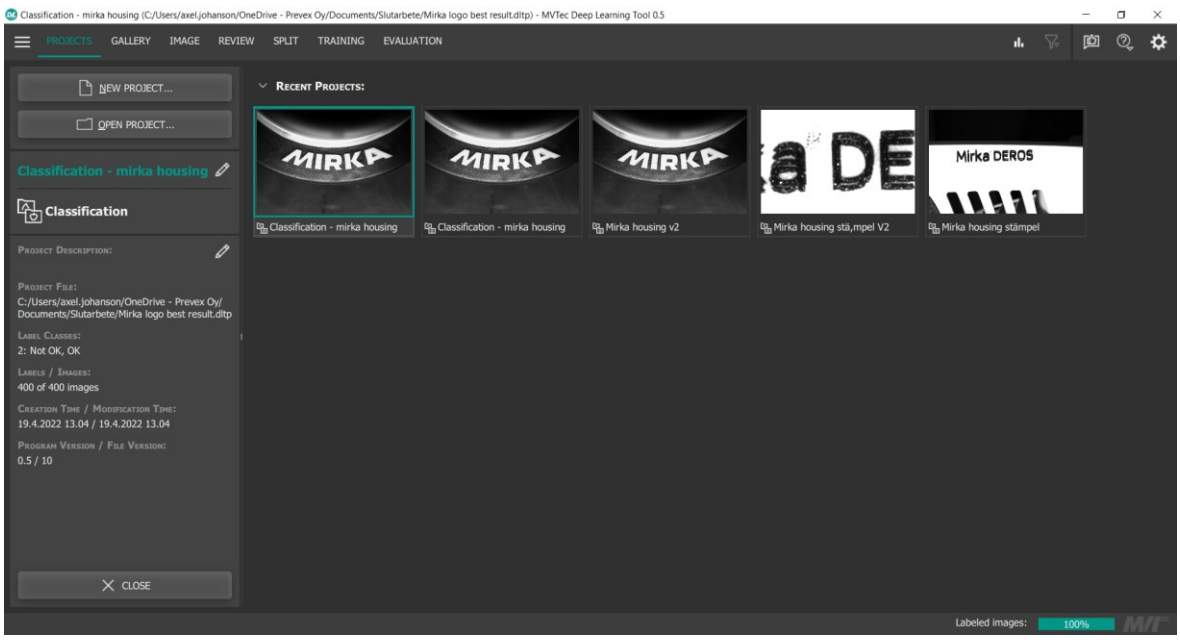
Bilderna av den formsprutade logon kunde loggas från en kamera som fotograferar delarna när de kommer ut från maskinen som tillverkar dem. Dessa bilder har lite jämnare kvalitet eftersom det är en robot som håller i delarna när de fotograferas, jämfört med bilderna av den stämplade logon som togs för hand.

Figur 9. Den formsprutade logon.

3.2 MVTec

Efter att testbilderna var tagna så var MVTecs Deep Learning Tool det första programmet som testades. Programmet kan laddas ner gratis från MVTecs hemsida, endast registrering på sidan krävs, sedan kan man utföra nedladdningen av programmet.

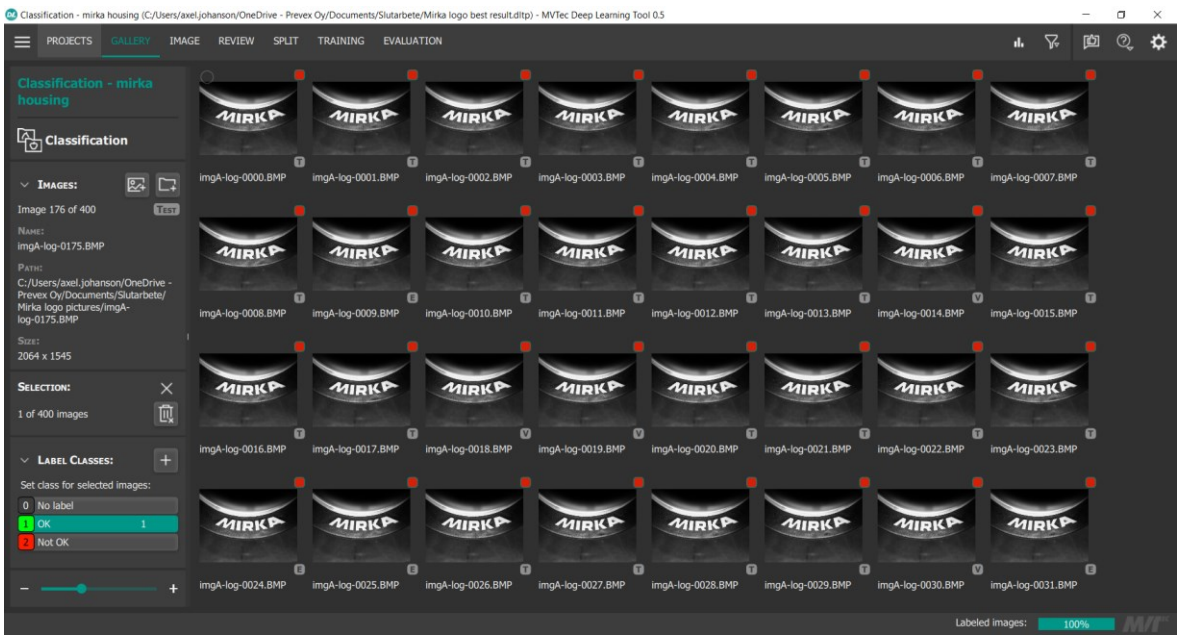
Programmet består av sju olika sidor som man stegvis går igenom för att skapa en djupinlärningsmodell som sedan går att använda i Halcon-programmet för att skapa ett fullständigt kvalitetsgranskningssystem med djupinlärning.



Figur 10. Deep Learning Tool.

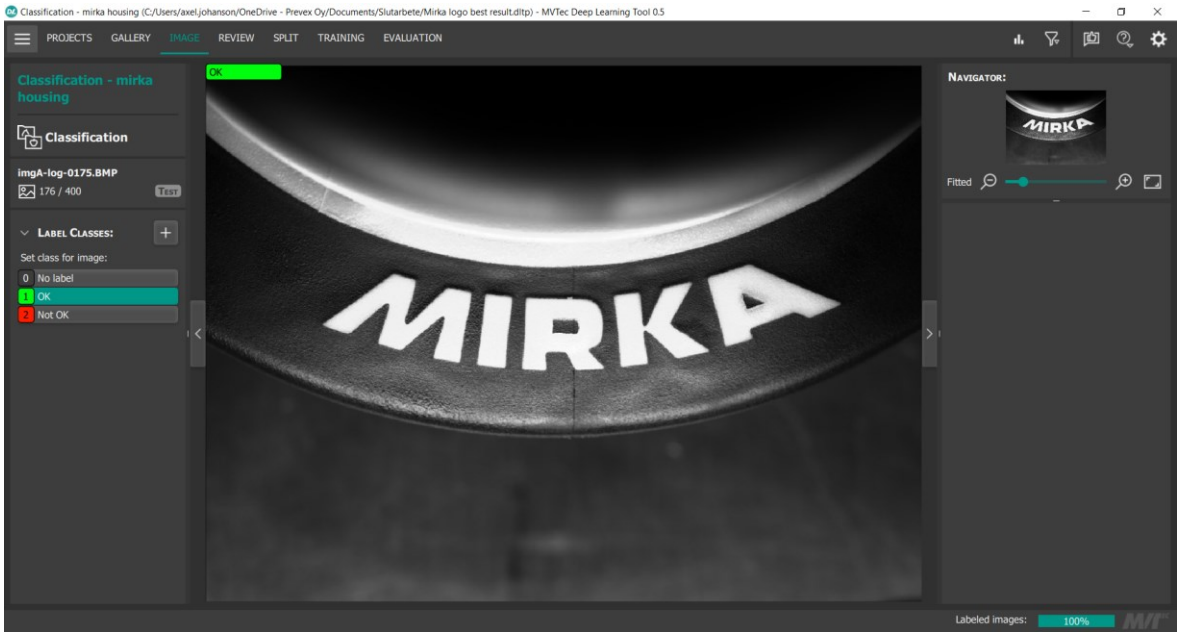
På den första sidan i programmet som syns i figur 10 skapar man nya eller öppnar gamla projekt. I denna version av Deep Learning Tool kan man välja mellan att skapa ett projekt som använder metoden Classification eller ett projekt med Object detection, i framtida versioner kommer det troligtvis att finnas ytterligare metoder man kan använda sig av. För detta test valdes metoden Classification, eftersom den metoden passar bäst för kvalitetsgranskning.

Den andra sidan är gallerisidan där man importerar bilderna som ska användas, det finns ingen funktion i Deep Learning Tool för att ansluta en kamera direkt till programmet och ta bilder, därför användes Halcon tidigare för att ta bilderna som efteråt importerades till galleriet.



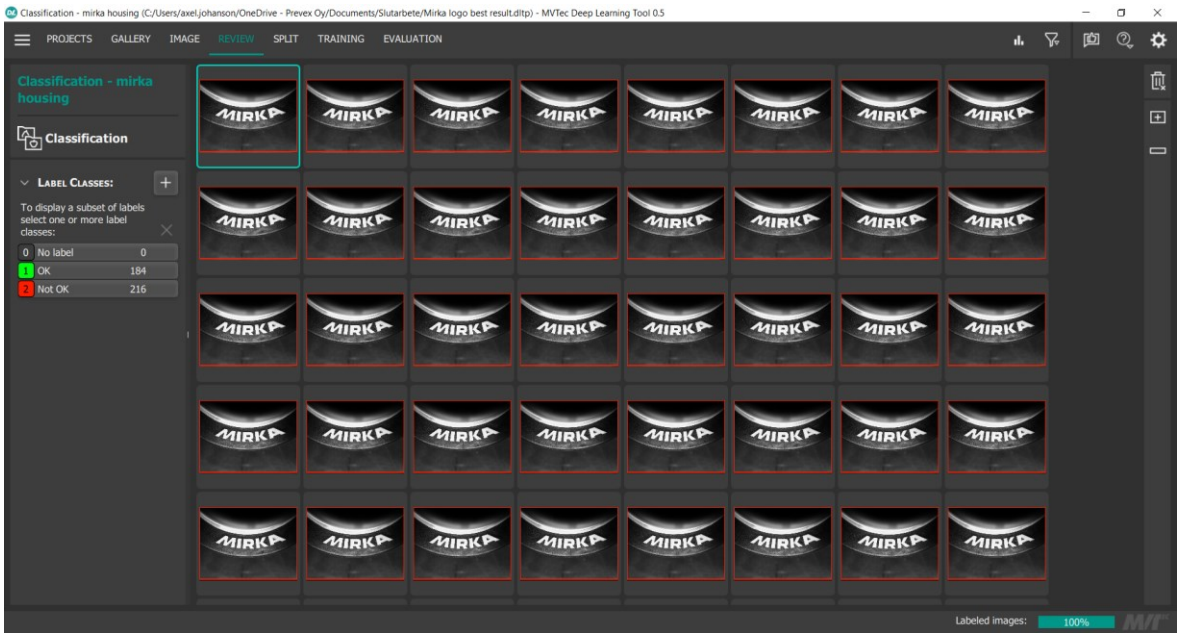
Figur 11. Galleri.

Eftersom Classification-metoden valdes så gällde det på den tredje sidan att klassificera varje bild enskilt med etiketterna ok eller icke ok, eftersom metoden för det mesta används för kvalitetsgranskning där man vill utesluta om produkten som granskas är defekt. Skulle man ha valt Object detection-metoden skulle man i stället på den tredje sidan ha skapat flera olika etiketter ifall att bilderna innehöll flera olika objekt och sedan använt etiketterna för att märka ut objekten på bilderna.



Figur 12. Bildsidan där bilderna granskas enskilt.

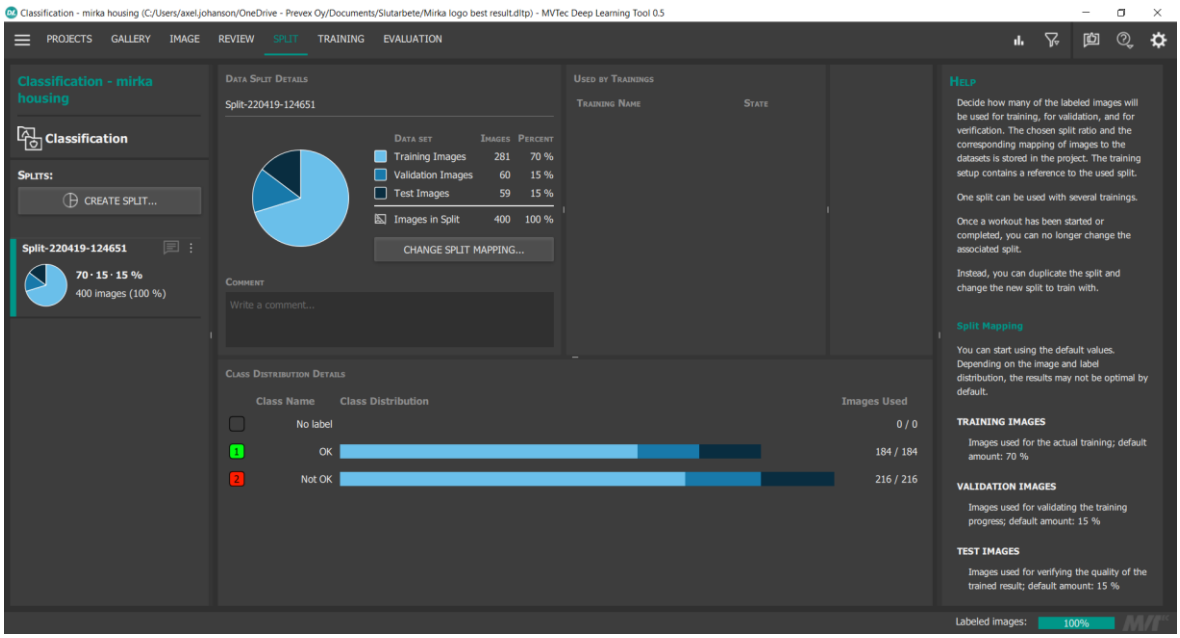
På sidan Review får man en överblick över alla bilder som har blivit sorterade, men man kan även markera flera bilder samtidigt och sätta etiketter på dem ifall man till exempel inte vill gå igenom alla bilder enskilt på föregående sida.



Figur 13. Review-sidan.

När alla bilder hade etiketter kunde man gå vidare till den femte sidan och skapa en så kallad split eller cirkeldiagram som delar upp alla bilder i tre grupper som kallas träningsbilder, valideringsbilder och testbilder. Träningsbilderna som är den största gruppen är bilderna som programmet använder för att träna djupinlärningsalgoritmen, valideringsbilderna är bilder som används för att bekräfta hur träningen framskrider och testbilderna används till att verifiera hur bra träningsresultatet är.

Alla tre grupper behövs för att man ska kunna skapa ett träningsprogram så även om det går att ändra på hur stor andel bilder man använder i en viss grupp så går det inte att ha noll bilder i någondera av grupperna, dessutom kan resultatet bli mera opålitligt ifall man går och ändrar värdena så det rekommenderas att man håller sig till standardindelningen. Eftersom det går att ändra på indelningen så kan man skapa flera cirkeldiagram vid behov för att testa med olika värden i träningsmomentet på den sjätte sidan i programmet.

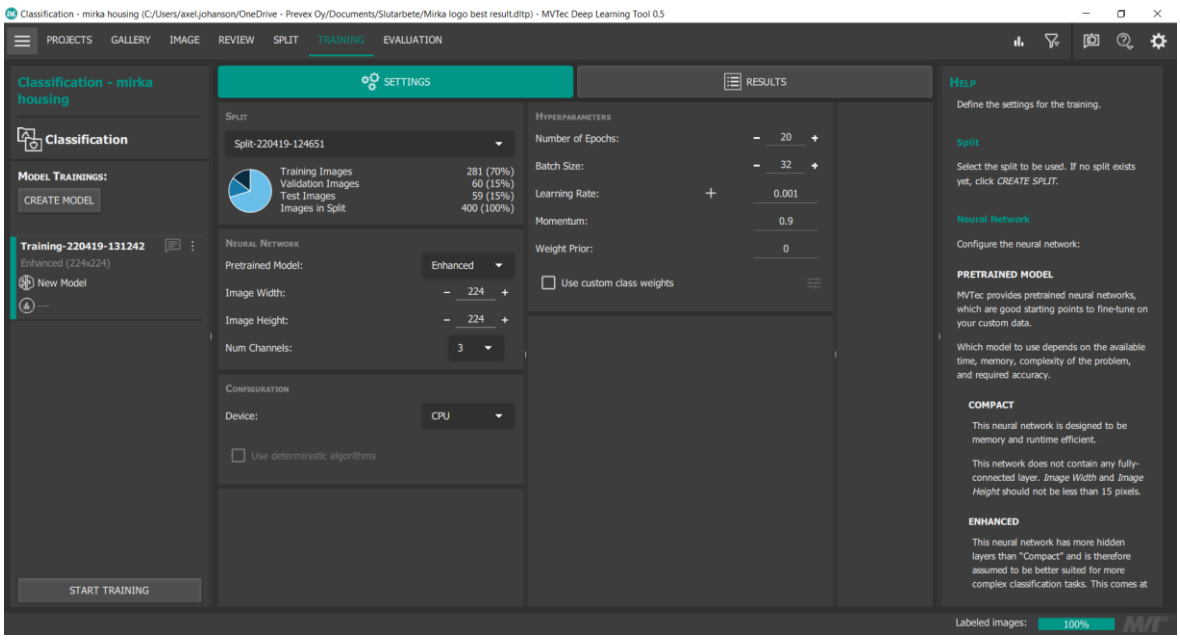


Figur 14. Split-sidan.

När man går vidare till nästa sida väljer man vilken split man vill använda för träningsmodellen och när man valt den kan man välja mellan fem olika neurala nätverk, i figur 15 har nätverket Enhanced valts men man kan även välja mellan Compact, ResNet-50, MobileNetV2 eller AlexNet. Alla nätverken testades men Enhanced nätverket visade sig fungera bäst för bilderna som granskades i denna undersökning, i figur 16 syns

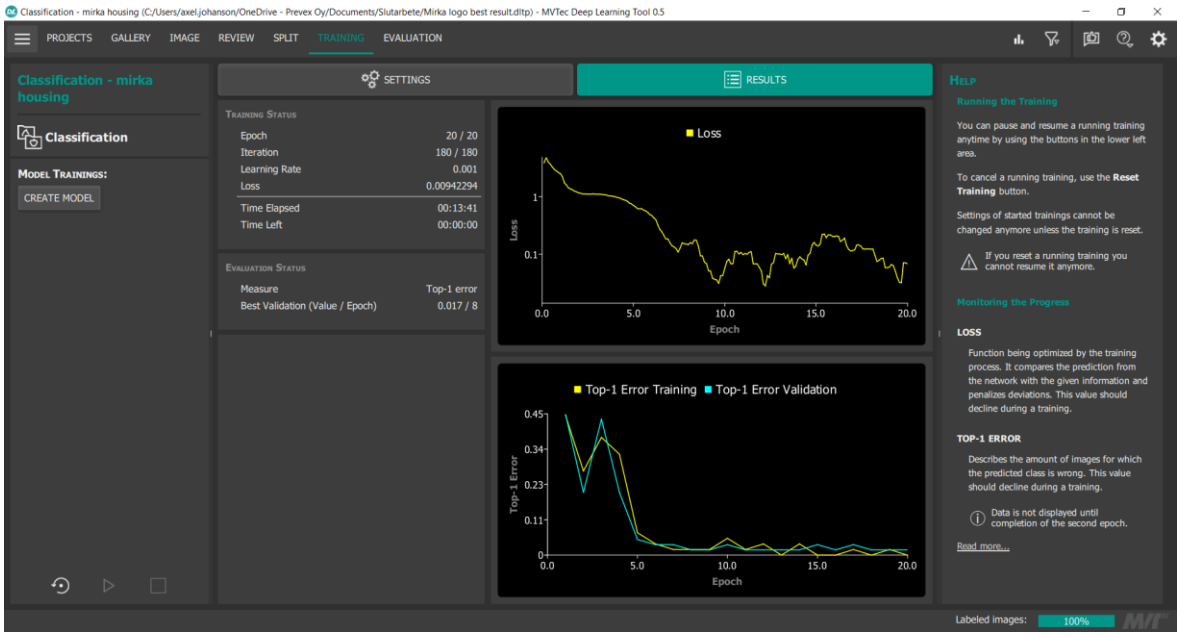
resultatet för en träning med Enhanced-nätverket, med de andra nätverken blev kurvan betydligt sämre än kurvan i den figuren.

För att utföra träningen kan man använda datorns grafikkort (GPU) eller processor (CPU), vanligtvis är grafikkortet det snabbare valet men av någon okänd orsak kraschade programmet varje gång när grafikkortet användes, i stället användes datorns CPU för alla test. Det finns även några parametrar, för mera avancerade användare, som man kan ändra på under rubriken hyperparameters men endast de förinställda värdena användes för detta test. När allt är inställt som man vill ha det trycker man sedan på Start Training-knappen för att starta träningen.



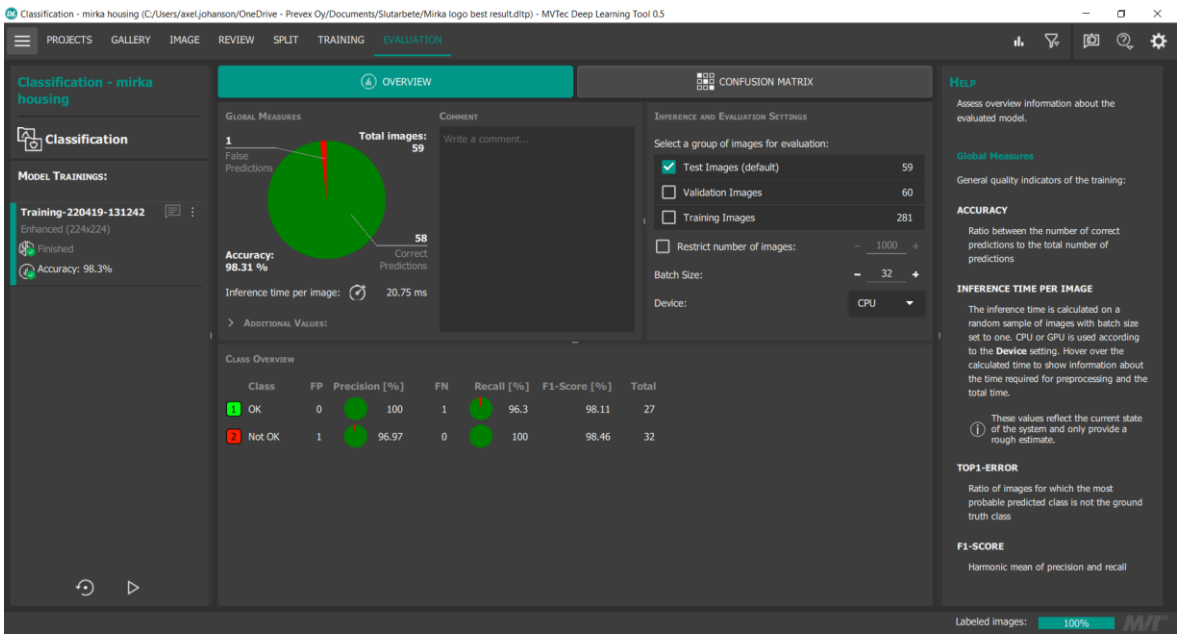
Figur 15. Inställningarna på tränings-sidan.

När träningen är klar går man vidare till resultatfliken för att granska resultatet av träningen. Om träningen har fungerat som den ska går den gula kurvan neråt i graferna och ju lägre ner den går ju bättre har träningen gått, om kurvan inte sjunker är det något som gått snett i inställningarna eller så kan det vara något fel med bilderna som används. I den andra grafen där det även finns en blå kurva ska den blå kurvan ungefär följa den gula kurvan ifall det inte finns allt för mycket avvikelser i träningen. Men för att tydligare se hur träningen riktigt har gått går man vidare till den sjunde och sista sidan i programmet.



Figur 16. Resultatet på tränings-sidan.

På den sista fliken, den så kallade Evaluation-fliken, finns de två olika sidorna Overview och Confusion Matrix som man kan granska resultatet från träningen på. Ett cirkeldiagram med antalet falska och korrekta förutsägelser visas på överblickssidan och det kanske viktigaste värdet som visas är noggrannheten för träningen som i figur 17 var 98,31%, vilket var ett av de högsta värdena som nåddes upp till med de testbilder som användes.

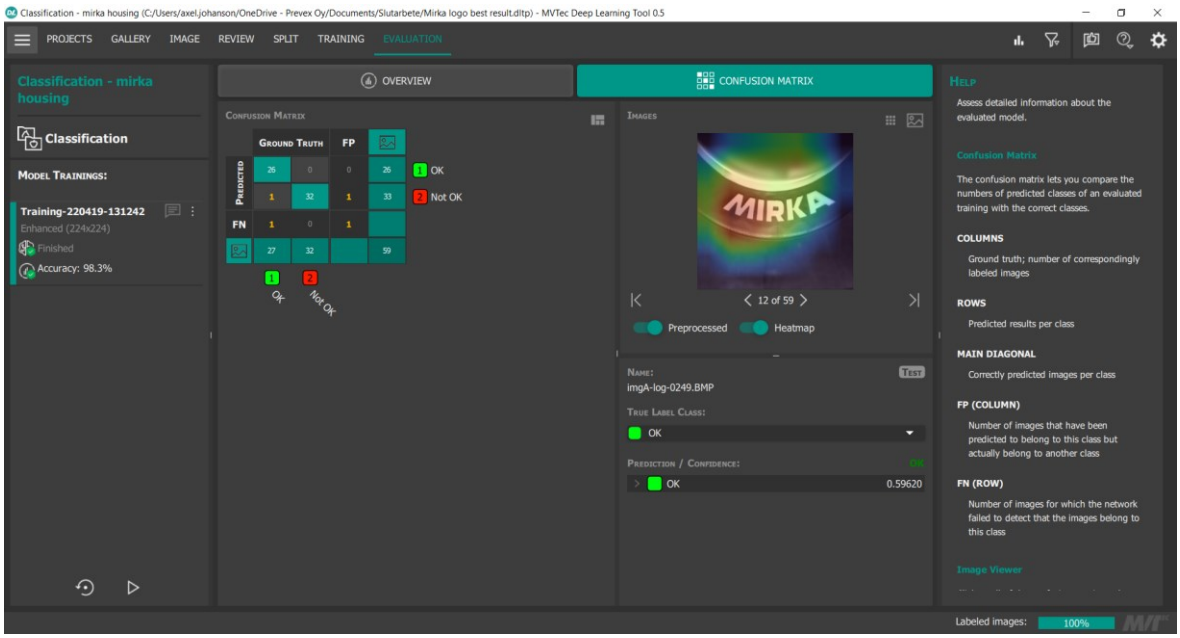


Figur 17. Overview.

I figur 18 kan man se träningsresultatet uppdelat i en liten matris där de gröna rutorna är antalet bilder som träningsalgoritmen har förutspått rätt, medan de gråa rutorna innehåller felaktigt förutspådda bilder. Längst nere till vänster ser man en etta i den grå rutan, vilket betyder att programmet har förutspått en bild felaktigt, den bilden är i verkligheten en acceptabel bild men träningsalgoritmen klassificerade den som en icke godkänd bild.

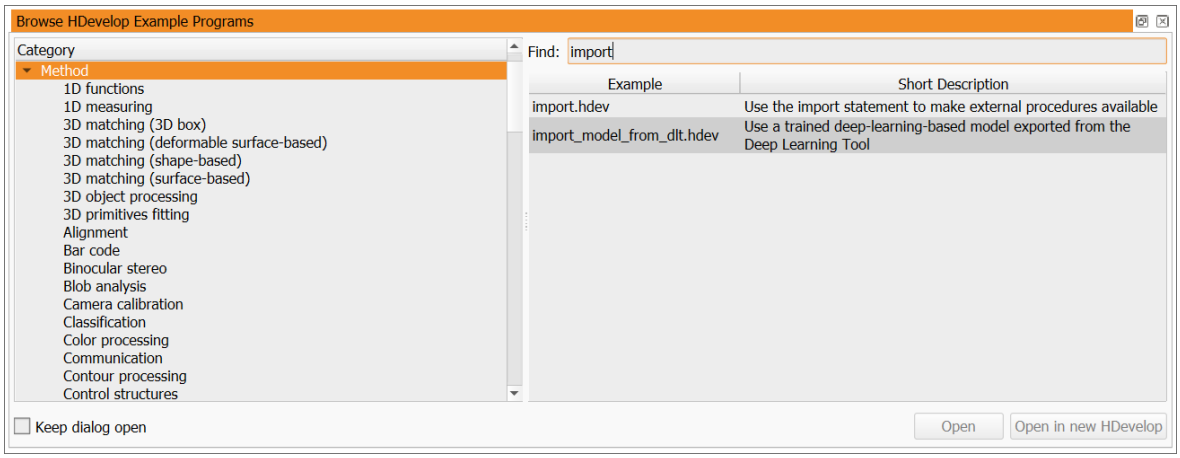
Till höger om matrisen finns en ruta där man kan bläddra igenom alla bilder och genom att aktivera alternativen Preprocessed och Heatmap under bilden kan man se en värmekarta över bilderna som visar vilket område som programmet har granska mest intensivt för att klassificera bilden. Detta verktyg är väldigt värdefullt och med detta kan man ganska enkelt avgöra ifall träningsalgoritmen kollar på de områden som man vill att ska granskas eller om algoritmen är helt ute och yrar.

Under bilden ser man namnet på bilden, bildens klass och ett Confidence-värde mellan 0 och 1. Ju närmare värdet är 1 så har algoritmen varit ganska säker på att bilden tillhör den godkända klassen och ju närmare 0 så har den varit säker på att den tillhör den icke godkända klassen, men om värdet är runt 0.5 har den inte varit riktigt säker på vilken klass bilden tillhör.



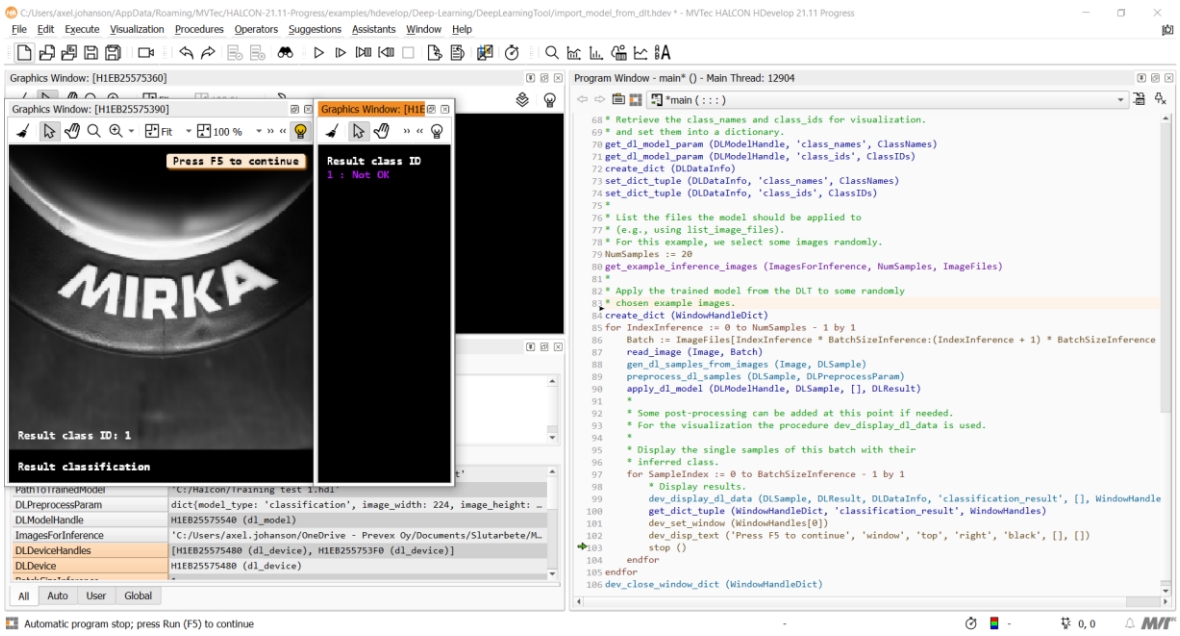
Figur 18. Confusion matrix.

När man har en träningsmodell i Deep Learning Tool som man är nöjd med kan man exportera träningsmodellen till Halcon-programmet med ett färdiggjort exempelprojekt i programmet som MVTec rekommenderar att man använder. (Deep Learning Classification with the MVTec Deep Learning Tool, 2021).



Figur 19. Importering till Halcon.

Det enda som behövde göras för att man skulle kunna testa träningsmodellen i Halcon var att i programfönstret specificera filsökvägarna till träningsmodellen och till mappen med bilderna man ville köra testet på. I figur 20 syns det hur en pågående testkörning kan se ut, i grafikfönstret ser man bilden som granskas och resultatet från granskningen.

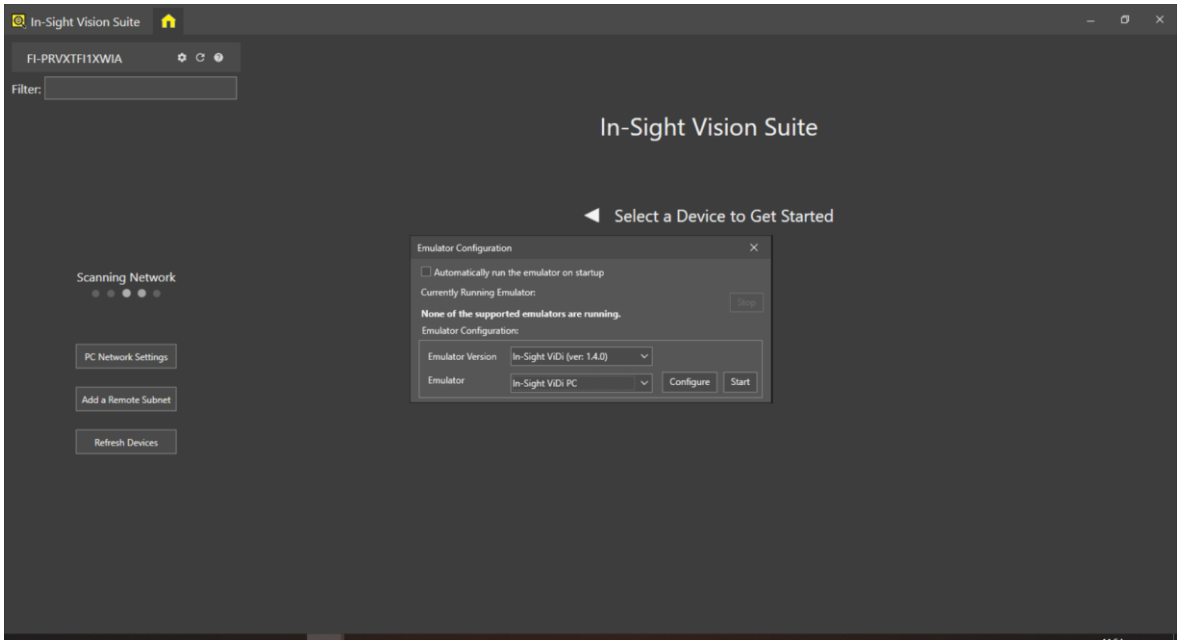


Figur 20. Testkörning av modellen i Halcon.

3.3 Cognex

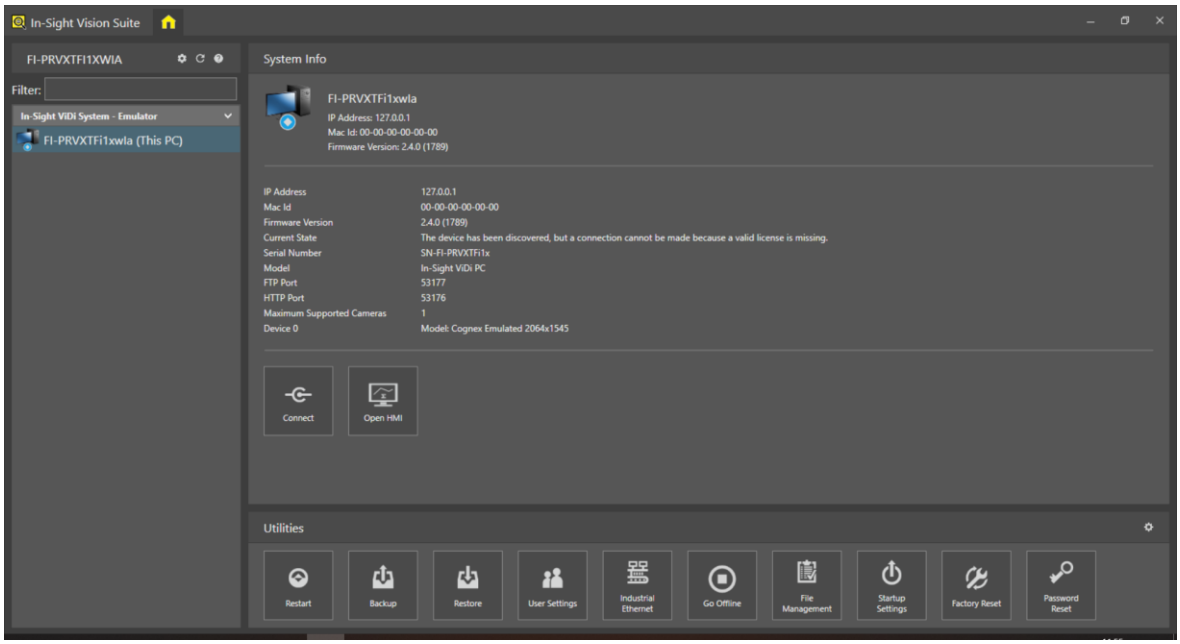
Som följande program testades In-Sight ViDi från Cognex, eftersom detta program kostar en hel del och endast skulle användas för testning för detta arbete lånades en licens från ett annat företag. Vanligtvis skulle även en Cognex kamera behövas för användning av programmet, men för testningssyften kan en emulator användas när man endast vill testa programmet med färdigt tagna testbilder. Samma testbilder som användes för testningen av MVTecs djupinlärningsprogram användes även i detta program.

Programmet som all konfiguration sker i heter In-Sight Vision Suite, men själva djupinlärningsprogrammet kallas för ViDi och detta program finns integrerat i Vision Suite. När programmet öppnades valdes emulatorn som testmetod eftersom ingen kamera fann till förfogande. Emulatorns konfigurationsruta syns i figur 21, där valdes typen av emulator och sen startades emulatorn i nedre högra hörnet av rutan. Det som kan vara värt att nämnas är att datorns virusskydd blockerade emulatorn och In-Sight ViDi behövde läggas till som ett undantag i brandväggen i virusskyddet för att emulatorn skulle starta.



Figur 21. Startsidan i In-Sight Vision Suite.

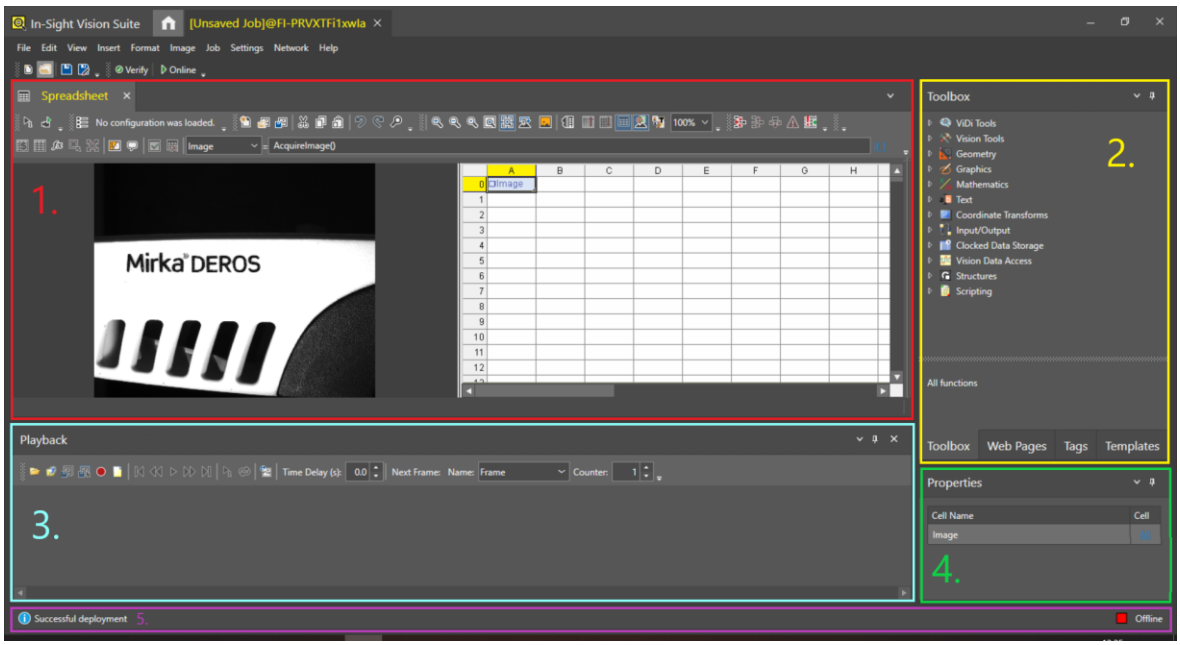
När emulatorn väl var startad kom man till menyn i figur 22, genom att trycka på Connect i den menyn ansluter man till emulatorn och tas vidare till själva projektsidan i programmet.



Figur 22. Emulatorn.

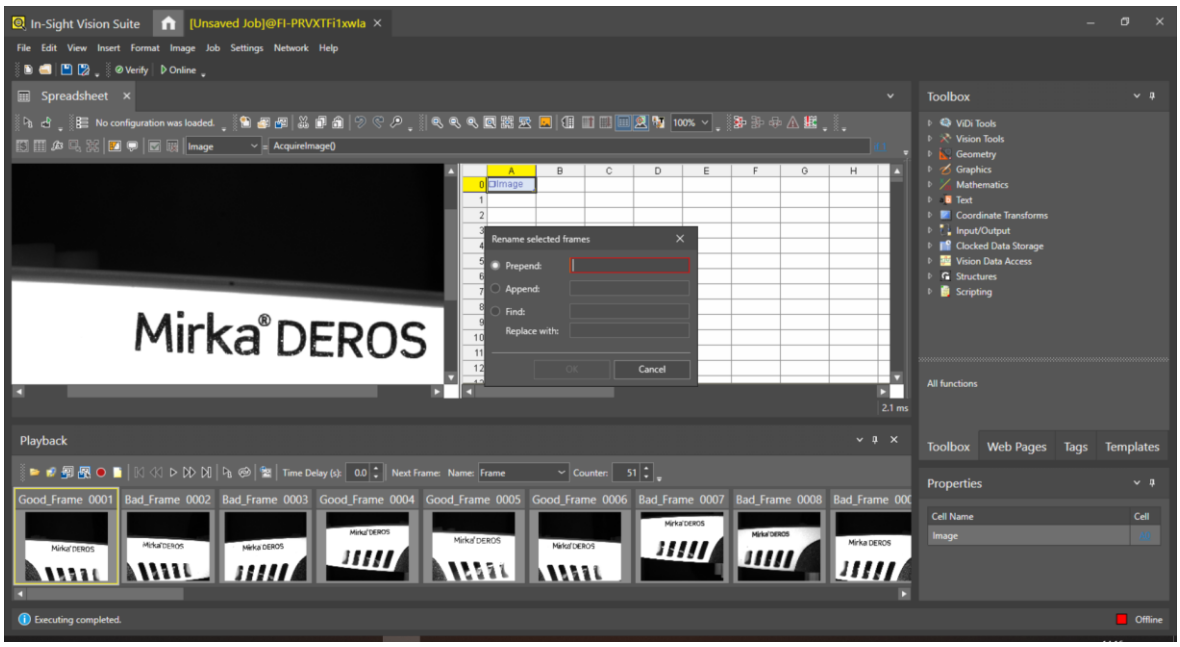
Projektsidan består av fem olika områden som är numrerade i figur 23. Det första området är kalkylbladet som är det område dit alla verktyg och all information som ska visas sätts. På det andra området kan man växla mellan olika flikar, men den mest väsentliga fliken som användes i detta test var verktygslådan, där hittar man ViDi Tools som är djupinlärningsverktygen. Det tredje området är uppspelningsrutan dit testbilderna hamnar, man kan endera ta bilder direkt i programmet ifall man har en kamera ansluten eller så importerar man en mapp med bilder, vilket gjordes i detta test. I det fjärde området hittar man egenskaperna för den markerade cellen på kalkylbladet. Det sista området som syns längst ner i figur 23 är statusfältet.

För att importera testbilderna behövde man endast trycka på mappikonen i uppspelningsrutan och specificera filsökvägen för bilderna. Hade en Cognex kamera varit ansluten skulle man ha kunnat ta bilderna direkt i programmet genom att trycka på den röda inspelningsknappen i uppspelningsrutan.



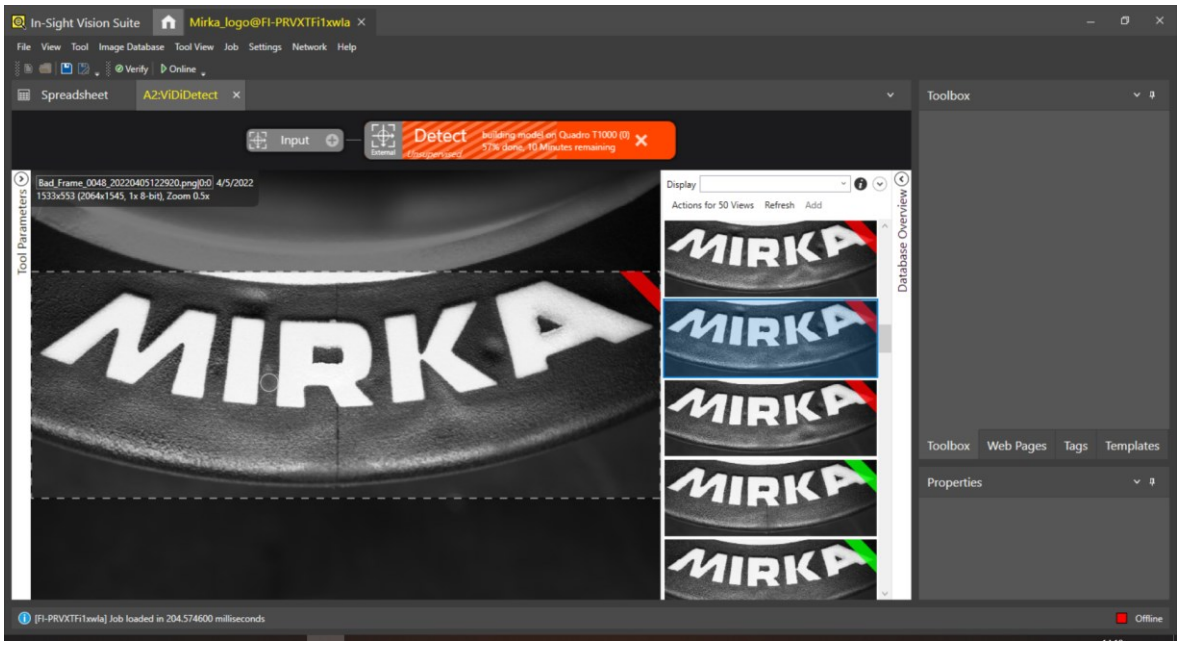
Figur 23. Projektsidan.

När bilderna var importerade kunde man lägga till exempelvis Good eller Bad framför bildernas namn genom att markera bilderna, högerklicka och välja Rename selected frames. I rutan som då öppnas kunde man lägga till texten före bildens namn genom att skriva in texten i Prepend-rutan. Detta gjordes för att man lättare ska kunna hålla koll på bilderna och sortera dem senare.



Figur 24. Namngivning av bilder.

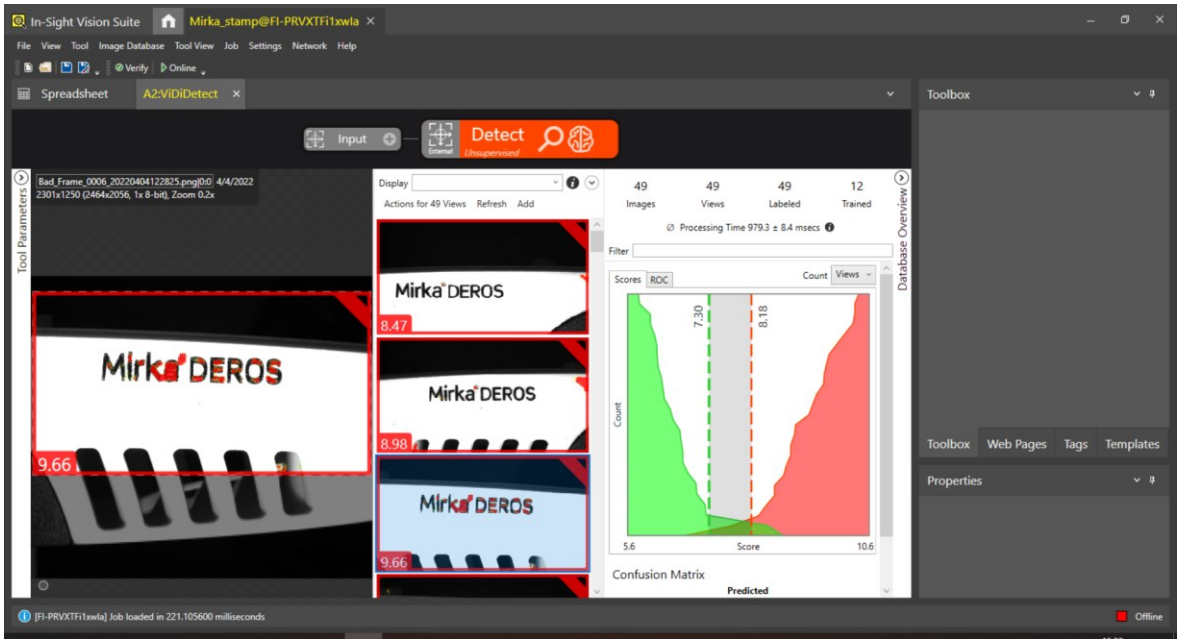
Som nästa steg drogs ett ViDi-verktyg in på kalkylbladet från verktygslådan, verktyget som användes i detta test var ViDiDetect som är det mest lämpliga verktyget för granskning av bilder där man vill hitta defekter i bilderna. De två andra verktygen som finns till förfogande är ViDiCheck och ViDiRead som går att använda skilt eller så kan man med verktyget som endast heter ViDi kombinera de olika verktygen. När verktyget dras in på kalkylbladet öppnas en ruta där man kan ändra hur stort område på bilderna som ska granskas, efter att man ställt in området gick man vidare till ViDi-redigeraren eller ViDi Editor som den kallas i programmet.



Figur 25. ViDi-redigeraren.

När ViDi öppnades fanns ingen data där, men genom att öppna Filmstrip, markera alla bilder och dra in och släppa dem i ViDi på det tomma vita fältet fick man dit bilderna som i figur 25. Genom att söka på Good i Display fältet kunde man visa endast de godkända bilderna och genom att trycka på Action for ## views kunde man markera alla godkända bilder som Good. Likadant gjordes för de icke godkända bilderna och de markerades som Bad. Med ViDiDetect-verktyget kan bilderna endast sorteras som bra eller dåliga eftersom man med detta verktyg vill utesluta vilka bilder som innehåller defekter, men inte vilka typer av defekter. Efter att bilderna hade rätt etikett kunde man ännu ställa in verktygsparametrarna för ViDiDetect på fliken i vänstra sidan, mera noggrann information om dessa inställningar behandlas i kapitel 3.4.

När inställningarna för verktyget var klara startades träningen genom att trycka på knappen som ska föreställa en hjärna bredvid förstoringsglaset uppe i den röda rutan.

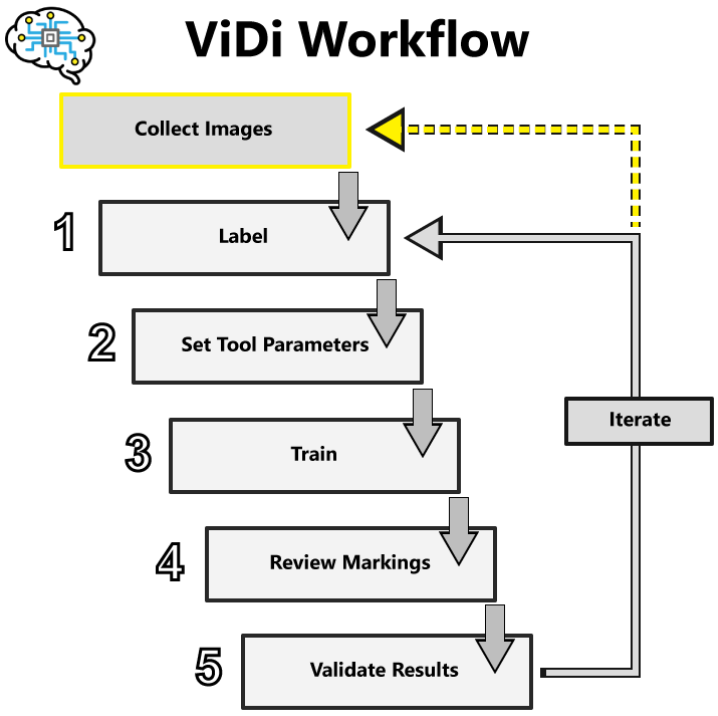


Figur 26. Träningsresultatet i ViDi.

När träningen var klar hade alla bilder som programmet godkände blivit markerade med en grön ram runt bilden och de icke-godkända med en röd ram, samt de defekter som programmet hittade i bilderna blev markerade med röda områden på bilderna. Alla bilder hade också fått ett poängvärde, genom att öppna Database Overview i högra hörnet fick man fram poänghistogrammet som syns i figur 26, där ser man ett bättre resultat på hur träningen har gått. Om det gröna och det röda området överlappar varandra betyder det att programmet har gjort en eller flera felsorteringar, för att resultatet ska vara så bra som möjlig så ska områdena inte överlappa varandra och ju längre bort från varandra de är desto bättre träningsresultat. I detta fall berodde felsorteringarna på att programmet hade sökt efter defekter på ett ställe i bilden som inte alls var relevant och då klassades bilderna som icke-godkända i stället för godkända, i figur 27 syns det hur programmet hittade ett fel på det svarta området i bilden. Detta kunde lätt ha undvikits genom att förminska området som man ville granska på bilderna.

Figur 27. Felsorterad bild.

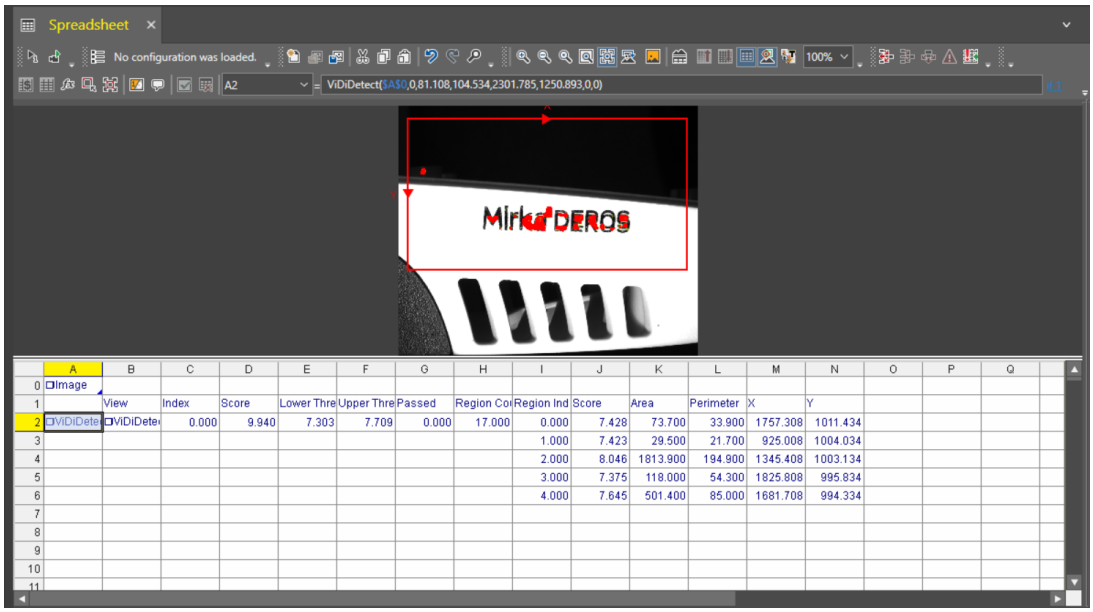
När man var nöjd med träningsresultatet kunde man förklara träningsmodellen färdig och gå tillbaka till kalkylbladet. Arbetsflödet som man går efter i ViDi kan beskrivas med bilden i figur 28 som hämtats från Cognex webbsida.



Figur 28. Arbetsflödet för ViDi. (Cognex VisionPro ViDi Help, 2020).

När man var tillbaka på kalkylbladet kunde man högerklicka på ViDiDetect-verktyget på kalkylbladet och trycka på Insert Getters för att få mera data från bilden som verktyget granskar. Den data som man får med kommandot kan användas till en hel del, till exempel om man vill skapa en HMI i ViDi behöver man data från Insert Getters-kommandot.

För att testa att djupinlärningsmodellen som blivit skapad med ViDi fungerade så markerades ViDiDetect på kalkylbladet och med play knappen uppe i vänstra hörnet kunde man gå igenom testbilderna en för en. Resultatet från varje bild såg man med värdena från Insert Getters-kommandot på kalkylbladet, samt genom att kolla på bilden ovanför kalkylbladet där defekterna på de underkända bilderna markerades med rött. För att testa med bilder som inte redan använts för träningen kunde man bara byta ut bilderna i uppspelningsrutan till nya bilder och hade man haft en kamera kopplad till programmet hade det bara varit att trigga kameran för att få nya bilder.

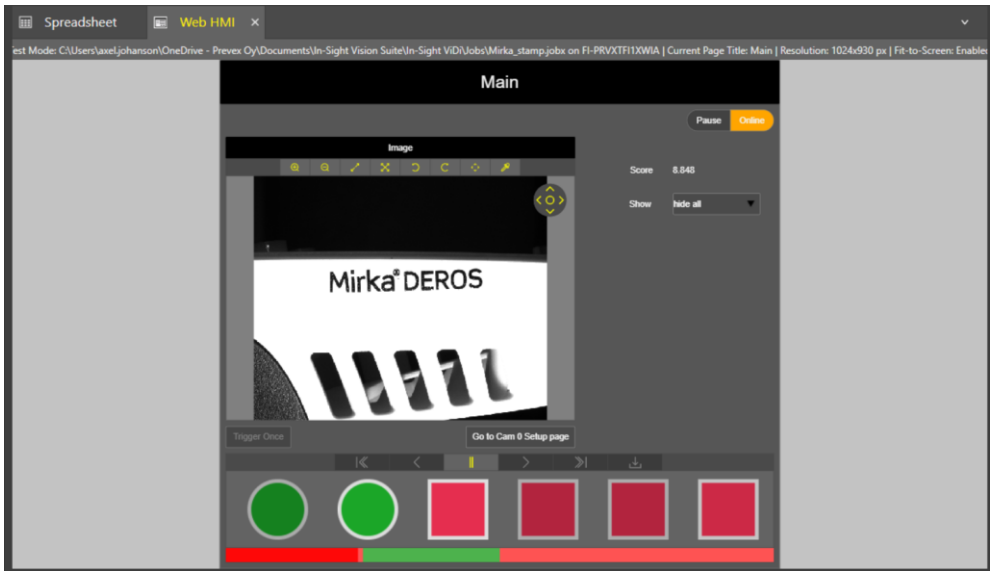


	A	B	C	D	E	F	G	H	I	J	K	L	M	N
0	Image													
1		View	Index	Score	Lower Thre	Upper Thre	Passed	Region Cor	Region Ind	Score	Area	Perimeter	X	Y
2	ViDiDete	ViDiDete	0.000	9.940	7.303	7.709	0.000	17.000	0.000	7.428	73.700	33.900	1757.308	1011.434
3									1.000	7.423	29.500	21.700	925.008	1004.034
4									2.000	8.046	1813.900	194.900	1345.408	1003.134
5									3.000	7.375	118.000	54.300	1825.808	995.834
6									4.000	7.645	501.400	85.000	1681.708	994.334

Figur 29. Insert Getters.

En HMI som den som syns i figur 30 är relativt enkel att skapa i ViDi, men eftersom ingen komplett djupinlärningslösning skapas i detta arbete gås det inte mera in på hur man gör för att skapa en liknande HMI. Eftersom In-Sight Vision Suite körs direkt från en Cognex kamera när programmet är färdigt konfigurerat så har Cognex gjort användningen av HMI:n väldigt enkel eftersom det är en webbaserad HMI, man behöver bara ansluta

kameran till det lokala nätverket så kan man gå in på HMI:n i webbläsaren på en dator eller smarttelefon som är ansluten till samma nätverk.



Figur 30. Exempel på en HMI i ViDi.

3.4 Jämförelse

Ett av de viktigaste momenten i detta examensarbete, jämförelsen av de två olika djupinlärningslösningarna från MVTec och Cognex, kommer att behandlas i detta kapitel. Jämförelsen gjordes i fyra olika punkter och dessa punkter granskades i de fyra kommande underkapitlen. Genom att göra jämförelsen fick man bättre överblick av de olika lösningarnas potential och användbarhet.

3.4.1 Komplexitet

En av de punkter som man ville veta mera om hos programmen var komplexiteten, med detta menas hur svårt det är att använda och få i gång ett fungerande system med programvaran. Den största märkbara skillnaden mellan MVTec och Cognex var att endast ett program behövde användas för att skapa ett system med Cognex, men med MVTec behövde man både Halcon och Deep Learning Tool för att skapa en fungerande helhet. Det betydde alltså att MVTecs lösning var nästan dubbelt mera tidskrävande att lära sig eftersom man ska lära sig två olika program och dessutom tog det längre tid att installera programmen. För båda företagen hittades videor gjorda av företagen själva som

handledde en genom grunderna för programmen, men hos Cognex fanns det mycket mera tydliga manualer på deras webbsida än vad det fanns på MVTecs webbsida.

Fördelarna med att skapa ett system med Cognex var att allt fanns i ett och samma program och programmet var relativt enkelt att navigera och förstå sig på med hjälp av videorna och manualerna som fanns till förfogande från företaget, ingen programmeringskunskap behövdes heller för programmet. Eftersom In-Sight ViDi endast testades med emulatorn i stället för en kamera från Cognex så går det inte att säga hur enkelt det skulle ha varit att ansluta en kamera, men användningen av emulatorn var väldigt enkel. Det som kan ses som en nackdel med detta program är att endast djupinlärningskameror från Cognex måste användas för att man ska kunna skapa ett fullt fungerande system, användningen av kameror från andra tillverkare är inte möjligt. Eftersom In-Sight ViDi körs direkt på kameran kan detta vara en nackdel om man har en produktionslinje där kvalitetsgranskning krävs på många ställen, då tvingas man köpa många kameror från Cognex, vilket kan bli väldigt kostsamt. Fördelen är dock att ingen dator behöver vara kopplad till kamerorna och hela systemet tar då väldigt lite utrymme.

Fördelarna med att ta i bruk ett system med MVTecs program var att Halcon, som användes för tagningen av testbilderna, stöder hundratals olika kameror och det finns då många förmånliga kameror att köpas. Användningen av Halcon kan ses som både en fördel och nackdel, detta på grund av att Halcon krävde en del programmering och var därför mera tidskrävande, men det ger också mera möjligheter att modifiera programmet till behov. Deep Learning Tool var väldigt användarvänligt och det var väldigt enkelt att skapa en djupinlärningsmodell med programmet. Nackdelarna är dock att användningen av ett MVTec system är mera tidskrävande och det kan kännas mera komplicerat nära två olika program måste användas.

3.4.2 Precision

Med precision anses hur väl systemet klarar av att ta rätt beslut, alltså hur bra klassificeringen fungerar under djupinlärningsträningen. Här kan det konstateras att både Deep Learning Tool och ViDi hade problem med att klassificera vissa av bilderna (se figur 18 och 26).

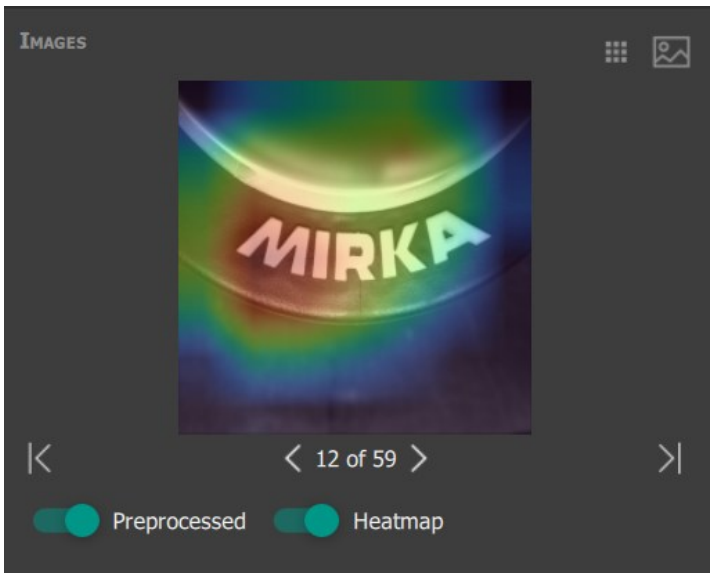
Vid testningen av ViDi skedde det mest felklassificeringar på grund av att för stort område på testbilderna granskades, felklassificeringarna kunde lätt ha undvikits genom att helt enkelt minska granskningsområdet i programmet. Som exempel kunde man se tidigare i figur 27 att programmet tog fel beslut på grund av ett fel som hittades helt utanför området som var intressant. Förutom felsorteringar som dessa så fungerade ViDi riktigt bra och bilderna blev rätt sorterade efter djupinlärningsträningen.

När Deep Learning Tool testades gick det inte riktigt lika bra, programmet hade ganska stora problem med att sortera bilderna rätt. Alla fem träningsmodeller testades och bästa resultatet fick man med Enhanced-modellen, men även den hade problem med vissa bilder, poängen för bilderna låg ofta på gränsen mellan godkänd och icke-godkänd. På värmekartan som visade var träningsmodellen sökt efter defekter kunde man se att den ibland sökt efter defekter på helt fel ställe, det verkade som att programmet hade svårt att urskilja defekter och det gick inte att minska på granskningsområdet som det gick att göra med ViDi.

3.4.3 Noggrannhet

Med noggrannhet menas hur små variationer man kan få fast med programmen. Här skiljde sig de två olika programmen en hel del från varandra, Deep Learning Tool hade problem med att hitta mindre detaljer och ViDi hittade däremot väldigt små detaljer.

I Deep Learning Tool fanns det väldigt få träningsparametrar att ändra på, det som gjorde störst skillnad var att välja rätt träningsmodell för att få fast så små variationer som möjligt. Med värmekartan kunde man se var programmet sökt efter defekter, men området som visades på värmekartan var alltid väldigt stort och det var ibland svårt att urskilja var defekten fanns och hur mycket programmet ansåg vara defekt. I figur 31 kan man se ett exempel på en defekt som programmet hittat, området markerat med rött är det område som urskiljer mest från referensbilderna och det anser programmet vara en defekt. På grund av områdets storlek var det väldigt svårt att urskilja defekten och som det syns i figur 31 så var två hela bokstäver och området runt dem markerat med rött även om defekten bara var en grop i det svarta materialet bredvid bokstäverna.



Figur 31. Värmekarta som markerar defekter i Deep Learning Tool.

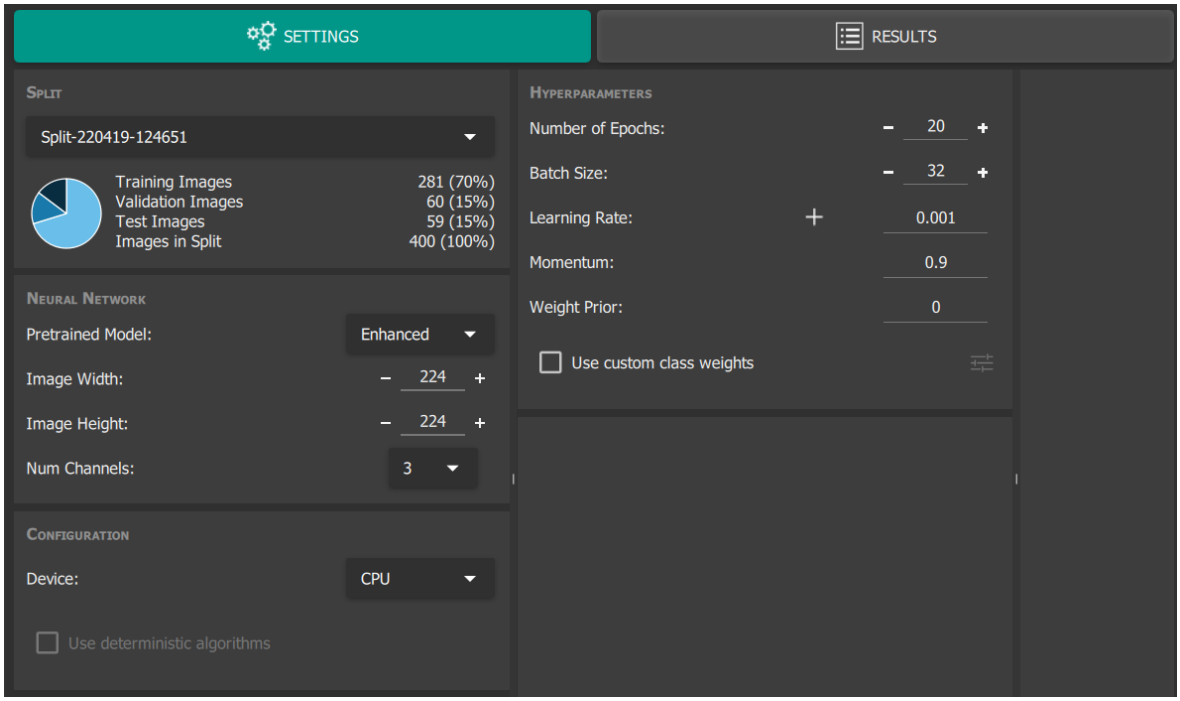
In-Sight ViDi-verktyget hade däremot mycket lättare att få fast små variationer och dessa variationer markerades med röd färg i stället för en värmekarta. Med denna metod var det mycket enklare att urskilja vilka variationer som programmet hade hittat i bilden och man behövde inte fundera vad programmet hade ansett som en defekt. I figur 32 var defekten att texten blivit stämplad två gånger och då hade texten blivit för tjock och programmet markerade då ställena där texten hade blivit för tjock eller otydlig. Med ViDi hade man dessutom möjligheten att välja hur stora defekter träningsmodellen ska söka efter, någon liknande parameter fanns inte i Deep Learning Tool.

Figur 32. Defekter markerade med rött i In-Sight ViDi.

3.4.4 Flexibilitet

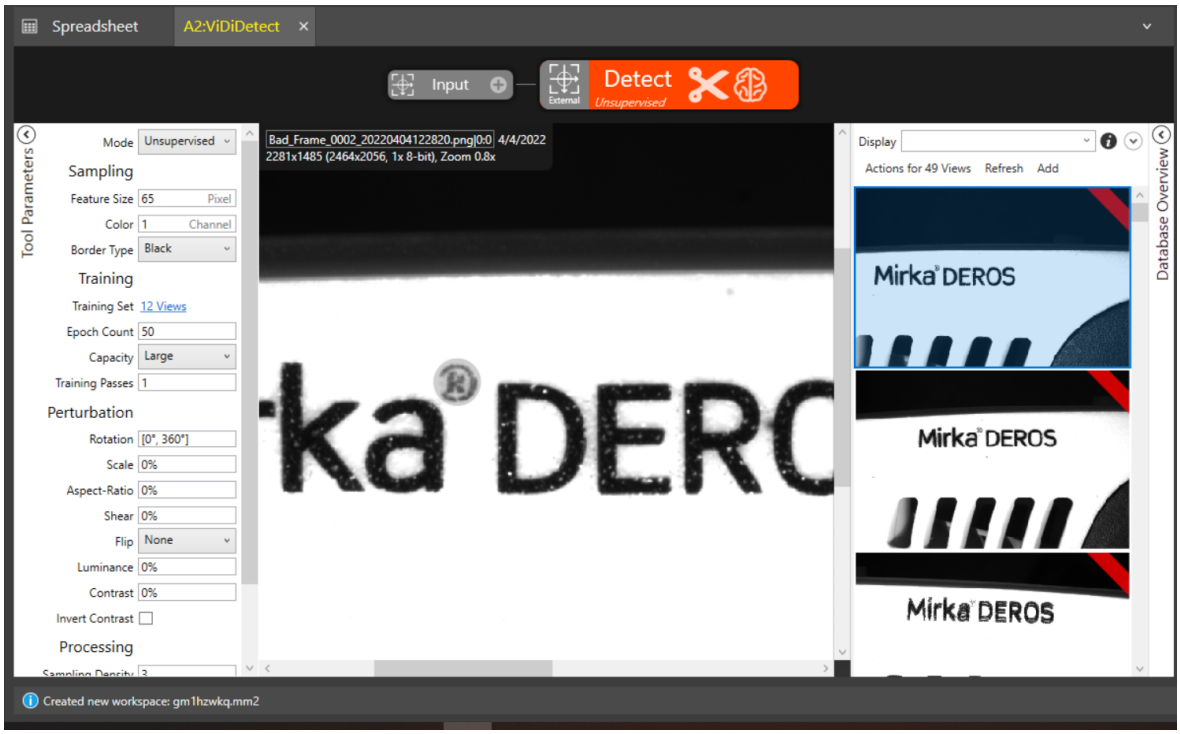
Möjligheten att påverka träningsresultatet med olika parametrar i programmen är vad som anses med flexibilitet. De två olika programmen var ganska olika i det här fallet, med Deep Learning Tool fanns det inte så många parametrar och inställningar att göra och med ViDi var det tvärtom, där fanns det betydligt flera parametrar som kunde påverka resultatet på djupinlärningsträningen.

Det som man kunde påverka i Deep Learning Tool, förutom kvalitén på bilderna, var parametrarna som syns i figur 33. Uppdelningen av antalet tränings-, validerings- och testbilder kunde man ändra. Man kunde även välja mellan fem olika träningsmodeller och ifall träningen ska utföras med datorns CPU eller GPU. Till slut fanns det även några hyperparametrar för träningsprocessen där man kunde ändra på hur många epoker träningen ska utföra, inlärningshastigheten för programmet och liknande saker. I stort sett fanns det alltså inte mycket parametrar, det verkade som att MVTec har tänkt att programmet ska i stort sett själv hitta defekterna i testbilderna och man har förlitat sig på de neurala nätverken, användaren har alltså inte så mycket att säga till om.



Figur 33. Träningsparametrar i Deep Learning Tool.

På kalkylbladssidan i ViDi fanns det möjligheten att ändra på hur stort område av testbilderna som ska granskas och i ViDiDetect fanns en hel del tränings- och störningsparametrar som syns i figur 34.



Figur 34. Parametrar i ViDi.

I ViDi hade man möjligheten att välja mellan de två olika lägena Unsupervised och Supervised, med den oövervakade metoden kategoriserar man bilderna med Good eller Bad och med den övervakade metoden märker man i stället bara ut defekterna som finns på bilderna. I Deep Learning Tool kunde man bara använda den oövervakade metoden eftersom det inte fanns något sätt att märka ut defekter på bilderna i programmet.

Andra värdefulla parametrar i ViDi var till exempel att man kunde ställa in storleken på defekterna med parametern Feature Size. Man kunde dra en grå cirkeln från nedre hörnet på bilden och placera den på en defekt och genom att minska eller öka storleken på cirkeln och anpassa den till defektens storlek fick man ett pixelvärde, vilket syns i parameterrutan, som hjälper programmet avgöra hur stora defekter det ska söka efter. Liksom i Deep Learning Tool kunde man även i ViDi ställa in antalet epoker som körs under träningen och genom att trycka på den blå texten bredvid Training Set fick man upp en ruta där man fick välja hur många procent av bilderna som ska användas för träning

och för testning. Med Rotation parametern kunde man göra det möjligt att granska bilder med föremål som blir roterade, genom att byta ut det andra värdet i parametern från 0 till 360 tar ViDi i beaktan att föremålet kan roteras till och med ett helt varv och klassificerar därför inte en rotation som en defekt.

4 Resultat

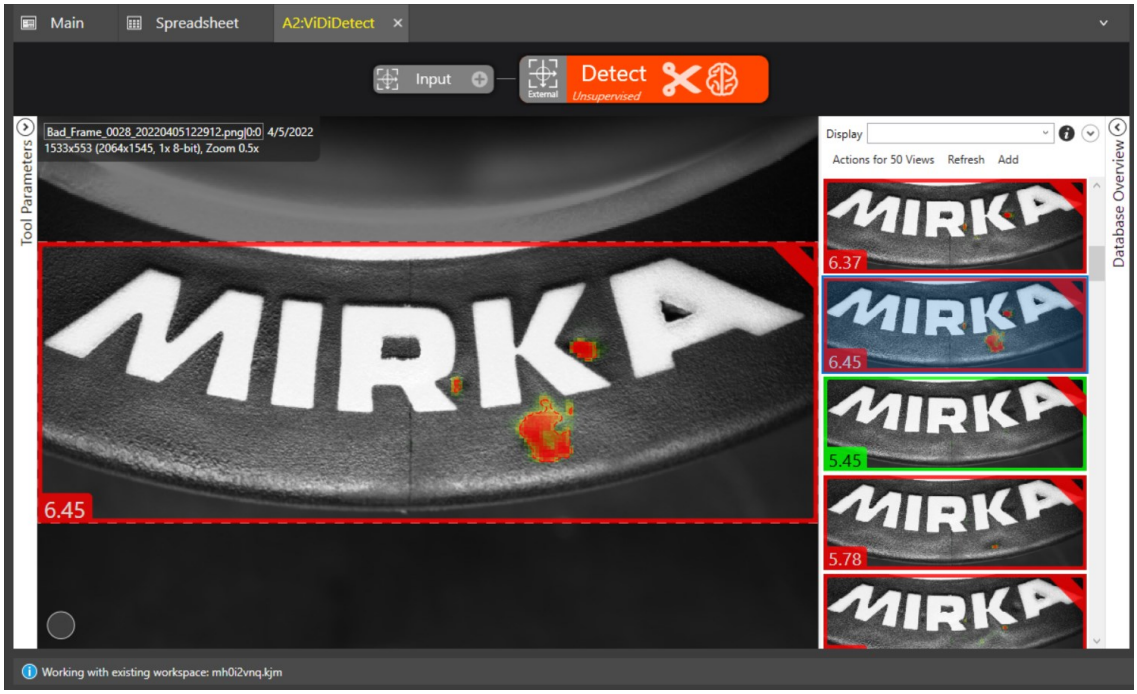
I detta kapitel presenteras resultatet för kvalitetsgranskningen av produkten med de två olika programmen och hur bra programmen lyckades klassificera bilderna med hjälp av djupinlärning.

4.1 Testresultat

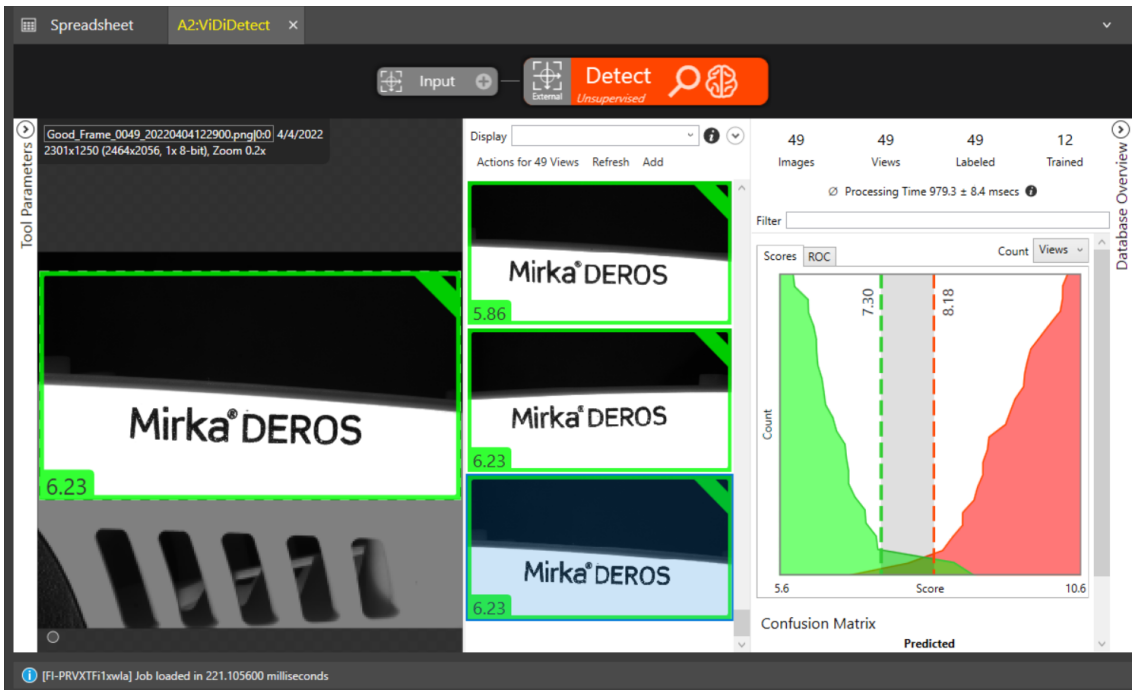
Programmet som lyckades bäst med kvalitetsgranskningen i detta test var Cognex In-Sight ViDi. De vanligaste förekommande defekterna som man ville få fast med programmet var om det fanns gropar i det svarta materialet på produkterna och så ville man få fast ifall Mirka® DEROS texten som blir stämplad på båda sidorna av produkterna hade blivit stämplad två gånger eller om färgen runnit ut och därför blivit otydlig. Dessa defekter hade ViDi inga större problem med att hitta, vid klassificeringen av bilderna som hade gropar i det svarta materialet gjorde programmet några enstaka felbedömningar under träningsprocessen. När träningen gjordes på bilderna med den stämplade texten så var resultatet väldigt bra och de enda bilderna som blev felkategoriserade var på grund av defekter som programmet hittade utanför området som skulle granskas, vilket syntes tidigare i kapitel 3.3 i figur 27, men detta var väldigt enkelt fixat genom att bara förminska granskningsområdet på bilderna.

Eftersom licensen till Cognex-programmet lånades under en kort period fanns det inte tid för så mycket testande och ingen Cognex djupinlärningskamera fanns till förfogande, bara emulatorn användes. Med tanke på dessa omständigheter och för att bara ha gjort några djupinlärningsträningar på testbilderna så fick man ett resultat som gott kunde ha använts till att vidareutveckla en fullständig kvalitetsgranskningsprocess med In-Sight ViDi-programmet.

I figur 35 nedan syns ett exempel på hur ViDi hittade gropar i materialet och med ViDis sätt att markera defekter med röd färg var det väldigt enkelt att se ifall programmet har förstått vad som ska klassas som defekter och var defekterna befann sig. I figur 36 ser man också en stämplad text utan defekter som är rätt klassificerad av programmet.



Figur 35. Defekter som ViDi upptäckte på produkten.

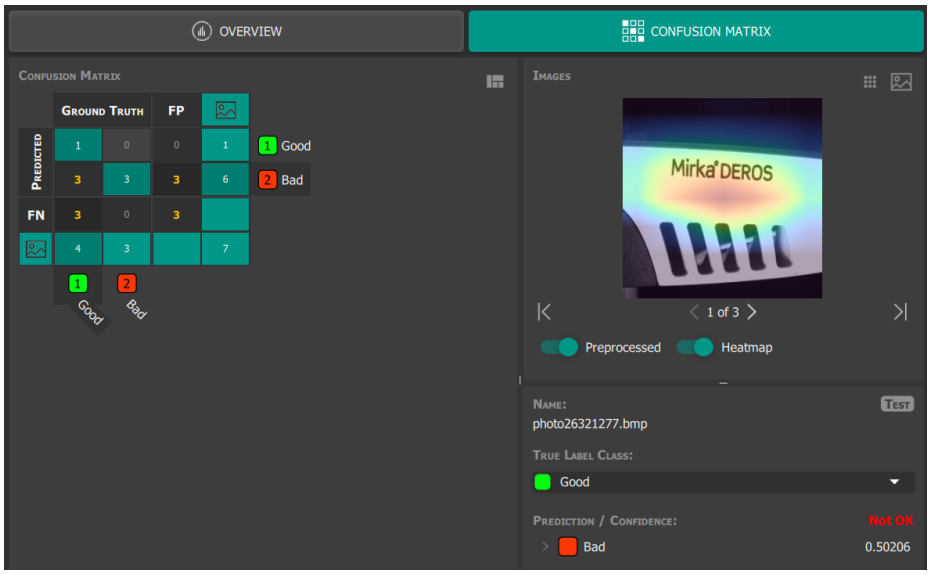


Figur 36. Produkt med godkänd text.

När det kom till MVTecs Deep Learning Tool var resultatet inte riktigt lika lovande även om det utfördes betydligt flera djupinlärningsträningar med det programmet. Deep Learning Tool fungerade ganska bra när det kom till att hitta groparna i materialet, men när den stämplade texten skulle granskas verkade det hopplöst att få programmet att förstå vad det skulle söka efter som defekter. Alla fem djupinlärningsmodeller testades men ingen av modellerna klarade av att urskilja text som blivit dubbelstämplad eller färg som runnit från stämplingen och utan att kunna specificera defekternas storlek eller dylikt så verkade det hopplöst att få programmet att hitta defekterna.

Vid testningen av Deep Learning Tool testades det att både öka och minska på mängden testbilder som användes, samt allt annat som man kunde tänkas ändra på, men inget verkade göra någon större skillnad. Deep Learning Tool skulle kanske ha gått att använda till granskning av groparna i det svarta materialet, men för att granska den stämplade texten kan det konstateras att programmet inte var tillräckligt pålitligt för det ändamålet.

I figur 37 syns ett exempel på hur programmet har hittat en defekt på produkten som det har markerat på värmekartan under den stämplade texten. Programmet hittade alltså en defekt som inte alls hade något med stämpeln att göra, djupinlärningsalgoritmen sökte alltså efter defekter på helt fel ställen och dessutom borde den inte alls ha klassificerat denna bild som en defekt.



Figur 37. En acceptabel stämpling som Deep Learning Tool klassat som en defekt.

5 Diskussion

Syftet med detta examensarbete var att studera och jämföra två olika kameraprogram som använder sig av djupinlärningsteknik och skapa sig en uppfattning om vilket av dessa program som vore bättre lämpat för kvalitetsgranskningen av en produkt som tillverkas på företaget Prevex. Som riktlinjer för examensarbetet gavs några punkter som företaget ville att jag skulle jämföra i arbetet, men upplägget av examensarbetet var annars ganska fritt. I arbetet framkommer lite fakta om djupinlärning, introduktion till företagen bakom kameraprogrammen och själva programmen, utförandet av testningen av programmen och till sist resultaten från testningen.

Under arbetets gång har jag lärt mig en hel del om djupinlärning och eftersom djupinlärning och artificiell intelligens är väldigt heta ämnen i dagens läge känns det som att denna kunskap kan komma väl till hands i framtida jobbsammanhang. Inga större utmaningar dök upp under arbetets gång, det som tog mest tid med arbetet var att lära sig hur de olika programmen fungerade.

Kvalitetsgranskning med så kallade smarta kameror var bekant för mig sedan tidigare eftersom jag har jobbat med sådana kameror, men att utföra granskningen med djupinlärningsteknik var helt nytt och ganska annorlunda jämfört med traditionella kameraprogram som används för granskning av olika produkter. Prevex hade licens till programmen från MVTec så de kunde testas i lugn och ro, men licensen till programmet från Cognex lånades från ett annat företag under en kortare tid så programmet från Cognex hann jag inte testa så länge, men jag tycker ändå att jag fick testresultat som dög från alla program.

Det som kunde ha gjorts bättre var kanske att jag skulle ha kunnat göra ännu flera tester för att kanske få ännu bättre resultat och man skulle ju alltid ha kunnat läsa på ännu mera om hur själva djupinlärningen fungerar för att ännu bättre förstå hur programmen fungerar och vad som sker i bakgrunden. Jag kunde också ha skrivit mera om teorin bakom djupinlärningen i arbetet, men eftersom teorin som blev inkluderad räcker gott och väl till för förståelsen av detta examensarbete lämnades mera djupgående teori bort. Det skulle även ha gått att utvidga detta arbete genom att inkludera program från ännu

flera tillverkare, men eftersom MVTec och Cognex var kända på företaget sedan tidigare valde jag att endast inkludera dem, dessutom kunde det ha varit svårt att få tag i licenser till andra program bara för testsyfte.

Jag anser ändå att jag lyckades med uppgiften att jämföra programmen och att man utifrån detta examensarbete kan avgöra vilket program som borde användas för granskning av produkten på företaget.

6 Källförteckning

COGNEX Company. (u.d.). Hämtat från https://www.cognex.com/en-fi/company/history

COGNEX Deep learning solutions. (u.d.). Hämtat från https://www.cognex.com/products/deep-learning

COGNEX In-sight D900 tools. (u.d.). Hämtat från https://www.cognex.com/products/deep-learning/tools

COGNEX In-Sight ViDi Help - In-Sight ViDi User Interface. (den 06 Januari 2022). Hämtat från https://support.cognex.com/docs/isvidi_150/web/EN/Help_ISViDi/Content/Topics/IDE/in-sight-vidi-ui.htm?tocpath=In-Sight%20ViDi%20User%20Interface%7C____0

Cognex VisionPro ViDi Help. (den 6 November 2020). Hämtat från https://support.cognex.com/docs/vidi_410/web/EN/vidisuite/Content/ViDi-Topics/overview/vidi-workflow.htm?tocpath=Cognex%20ViDi%20Tools%7C____1

Deep Learning Classification with the MVTec Deep Learning Tool. (den 19 Mars 2021). Hämtat från https://www.youtube.com/watch?v=7FTuOASol90 den 16 Oktober 2022

IN-SIGHT D900 VISION SYSTEM. (u.d.). Hämtat från Cognex: https://www.cognex.com/en-fi/products/deep-learning/in-sight-d900

KWH GROUP Prevex. (2019). Hämtat från https://www.kwhgroup.com/sv/annual_report/arsberattelse-2019-2/prevex/

MATHWORKS What Is Deep Learning. (u.d.). Hämtat från https://www.mathworks.com/discovery/deep-learning.html

MIRKA Mirka DEROS 550CV 125mm pölynpoisto 5,0. (u.d.). Hämtat från https://www.mirka.com/fi/Mirka-DEROS-550CV-125mm-polynpoisto-50-MID5502022/

MVTec Company. (u.d.). Hämtat från https://www.mvtec.com/company

MVTec Deep Learning Tool. (u.d.). Hämtat från https://www.mvtec.com/products/deep-learning-tool

MVTec Halcon. (u.d.). Hämtat från https://www.mvtec.com/products/halcon

MVTec Why HALCON. (u.d.). Hämtat från https://www.mvtec.com/products/halcon/product-information

PREVEX About us. (u.d.). Hämtat från https://www.prevex.com/about-us/