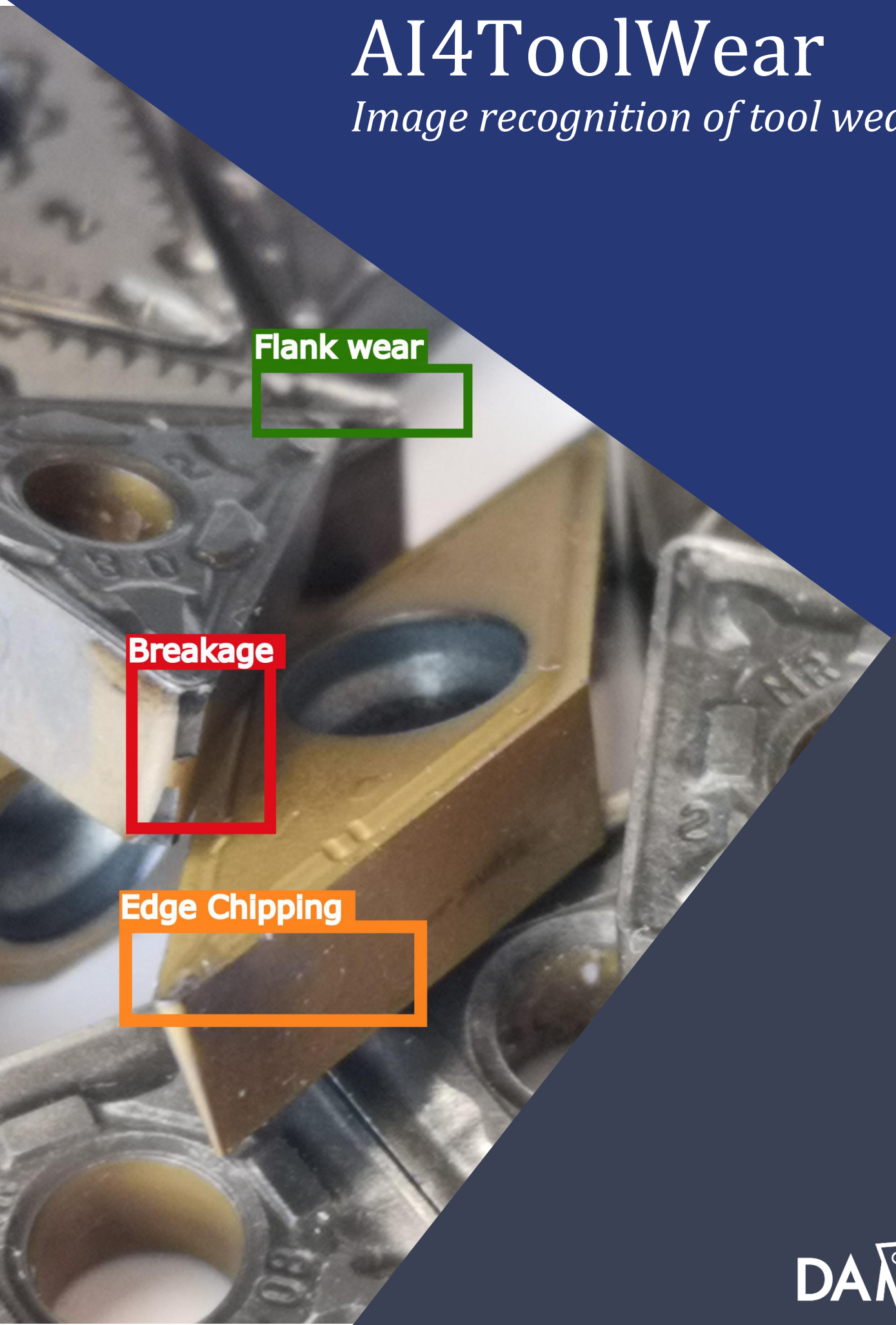
AI4ToolWear

Image recognition of tool wear



DAMRC

Table of content

This project is made in collaboration with:

Funding:

Contributors:

1 Executive Summary

The work of this report, AI4ToolWear, is built on the idea about utilising Artificial Intelligence, AI, in the machining industry. Specifically, the work focusses on the opportunity for optimizing the machining process by allowing automatic analysis of wear of cutting inserts.

The area of AI technology is huge and is rapidly developing on both a scientific and a commercial level. The State-of-the-Art AI methods and models within computer vision are investigated in this project, and an opensource framework for training is an object detection model is used. The result is a model that can analyse an image of a cutting insert and predict a boundary box around a wear type, and furthermore classify which wear type it is.

The project outcome is both a custom trained AI model and a dataset of cutting inserts, developed by DAMRC. The dataset consists of 1595 images that have been annotated with boundary boxes and wear types, namely Flank wear, Crater wear, Built-up edge, Notch wear, Edge chipping and Breakage. The equipment for capturing the images is relatively low-cost to ensure that the startup cost for using the AI model is low, as well. The developed dataset is used for training the presented AI model and can furthermore be used by DAMRC in future development of AI models.

The presented model shows good capability to detect wear types on the validation data set, used to benchmark a model during training. The model performance metrics shows that it can classify 40 % of its detections correctly, called precision, and it detects 39% of all wear instances present, called recall. Best predictions are made of the wear type Breakage, but this type is also the one mostly classified as a wrong wear type, namely Edge chipping.
The interference data, unseen images for quality check, has been used to understand the performance of how the model predicts wear on images captured by a digital microscope and images captured by a smartphone with an add-on microscope clip. The result shows that the model performs better on images from the digital microscope, this can be due to better image quality or the lack of images from the smartphone setup in the training dataset.

The results presented in this report concludes that AI has the potential to be used for automatic tool wear detection of cutting inserts. This opens for continue work both for improving the presented model and to further develop automatic detection for other types of tools. Improvements will require more annotated images of inserts and a better representation of wear types and image types. It is furthermore suggested to investigate the market to know how the automatic tool wear detection best brings value to the machining companies or other in the machining industry.

2 Introduction

A key part of machining metal parts is the cutting tool, which is used in the machine for material removal. During use, the cutting tool, which can come in many forms and types, is worn and will with time have to be replaced.
Knowing the wear state of cutting tools can provide significant insights into the effectiveness of the machining process and potentially extend the lifetime of the tool. However, inspection and analysis of cutting edges of the tool is today still performed manually and requires personal with experience in tool wear and machining. DAMRC has an interest in tool wear and is aiming to provide the industry with insights and solutions to reduce tool wear and the consequences of tool wear in the manufacturing of parts.

The idea of this project and its research is to investigate, how newest developments within artificial intelligence can be utilized in the metals machining industry, in the domain of tool wear determination.

DAMRC aims to develop a computer model that detects tool wear on inserts using artificial intelligence (AI) and image recognition. The foreseen benefits of using an AI model are gaining better consistency in tool wear analysis and opportunities to automate the machining process even further. The goal is to develop an AI model that can recognize wear comparable to the results of visual inspection performed by machinists, such as flank wear, chipping wear, flaking, etc.

If successful, DAMRC could further develop the algorithms to detect other types of tools and potentially enable the AI model to suggest changes in parameters to reduce tool wear. Additionally, the knowledge from developing a tool wear AI model, AI4ToolWear, could be used for other AI related topic.

3 Project Scope and description

This project is going to couple the technology areas of artificial intelligence and tool wear analysis and will be the foundation for automatic analysis of tool wear in metal machining processes by development of the first version of the AI4ToolWear model.

Cutting tools are manufactured in many shapes for different functionality, hence this project will focus on tool wear of cutting inserts only. Cutting inserts are used for various machining processes, like turning and milling. The insert tool type is considered well suited for training an AI model because they are manufactured to industry standards regarding dimensions and material.

The main questions to be investigated in this project are:

- Which AI methods, for image recognition, are suited for recognizing wear of tool inserts?
- How well can tool wear be detected by a custom trained AI model?
- Which equipment, i.e. microscope, can be used for detecting tool wear with AI?

Answering these questions will help create the foundation for automatic tool wear analysis in the future, and results from this project is foreseen to be used in further development of an AI4ToolWear model able to predict tool wear on various types of tools.

This report will be covering the above questions in three major sections, first a literature study for establishing the fundamental knowledge on the project topics. Second, experiments are using and investigating a custom trained AI model for detecting tool wear of cutting inserts. Thirdly, a conclusion on the project findings, together with future perspectives of an AI4ToolWear model is presented.

The success criteria of the project are:

- Establish understanding and competencies in AI technology within DAMRC
- Develop an AI model able to do automatic tool wear analysis and prove if the AI technology can be used for this use case or not.

4 Literature Study

4.1 Overview of literature study

A literature study within relevant fields of the project, AI4ToolWear, is conducted to give an understanding of the topics, the terminology, and the most recent developments within the area of tool wear and computer vision.

The literature study is reviewing relevant scientific papers to understand most recent results within use of AI for tool wear detection, and tool wear analysis in general. A practical understanding of AI and computer vision is obtained through searching and reviewing blog post, forums and training videos.

Section 4.2 and 4.3 will cover tool wear in general and what parameters are used to define tool wear, and how tool wear detection is done in the industry today. Section 4.4, 4.5 and 4.6 describes the topic of AI and computer vision and outlines the State-of-the-Art withing AI and tool wear detection. Section 4.7 will cover the opportunities of combining AI and tool wear detection. Finally, section 4.8 concludes the literature study and scopes the work of the project.

4.2 Definition of tool wear for inserts

Tool wear is of interest to the manufactures because the tool wear state affects both quality and cost of the produced parts. Change or restoring of tools with remaining tool life should be avoided, to reduce the downtime of the machine and opposite exceeding the tool life, leading to lower product quality or machine breakdown, should be avoided too. Assessment of the tool life is crucial to optimize the life time for a tool, therefore a methodology for assessment is presented by [1] Daicu, R. et al. together with a thorough description of the tool wear characteristics. These characteristics are reviewed in brief in following.

Tool wear has several causes, main causes are illustrated in figure 1, like material of specimen or due to speed of machining. Knowing the cause will help avoid or mitigate the wear. The wear can be classified as 1) Corner wear; 2) Flank wear; 3) Notch wear; 4) Crater wear. Parameters for determining severity of tool wear is listed in figure 2 and table 1. These parameters are found by skilled personal doing microscopic analysis of a tool. Most importantly is the flank wear and related parameters, V_B and V_{Bmax}, as this wear type should always appear. For uniform wear the V_B parameter is used for characterization of the wear, the admissible wear is then recommended to be V_B = 0.3 mm. In case of irregular wear, the V_{Bmax} parameter is used, and has a recommended admissible wear value of V_{Bmax} = 0.6 m.

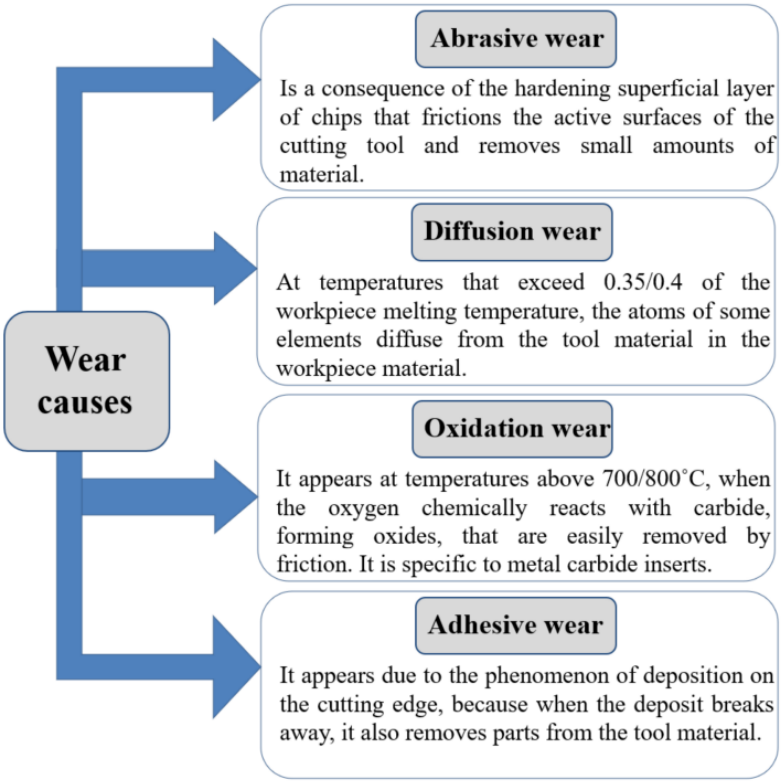


Figure 1 Main causes for tool wear, [1] Daicu, R. et al. 2022

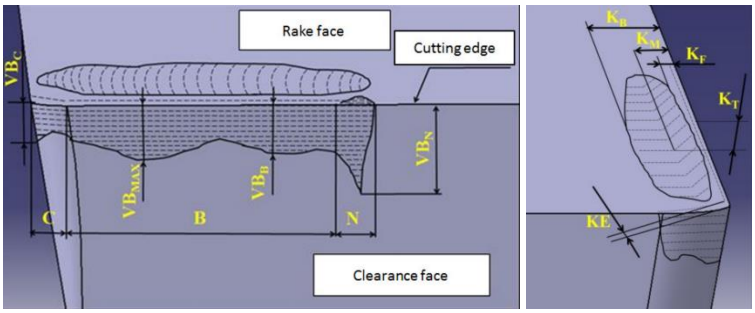


Figure 2 Illustration of tool wear for insert [1] Daicu, R. et al. 2022

Table 1 Parameters for characterization of tool wear for inserts

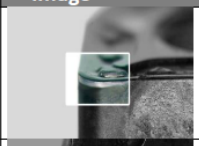
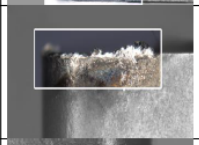
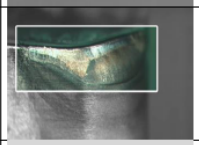
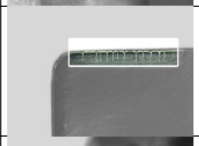
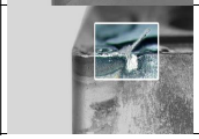
	Parameter	Description
Flank	V_B	Average width, flank wear, recommended admissible wear value VB_B=0.3 mm, for regular wear
	V_{Bmax}	Maximum width, flank wear, recommend admissible value wear V_{Bmax}=0.6mm, for irregular wear
Crater	K_T	crater depth
	K_M	Distance between the cutting edge and the deepest crater
	K_B	Distance between the cutting edge and the back crater contour
	K_F	Width of the land between the crater and the cutting edge
Corner	K_E	Radial displacement of the tool corner

The listed parameters give numerical values for tool wear, which are easy to compare between different inserts. Obtaining the numerical values still relies on the person analysing the tool wear and this can lead to differences even when analysing the same insert. These differences can come due to difficulty in assessing the transition between wear and non-wear of the tool, which impacts the values [1] Daicu, R. et al. 2022.

In addition to the numerical parameters, the visual appearance and shape of the tool wear can help determine what has caused the wear and how the wear can be prevented. Table 2 shows a list of visual appearance of tool wear on inserts and related characteristics and how to prevent the wear type. Additional information and other types of tool wear are described in Appendix A. Inspecting the visual appearance allows machinists to evaluate the tool wear and adjust the machining process accordingly, without doing a detailed analysis and calculating the wear parameters from table 1.

Table 2 Wear types and corresponding measures for inserts used for turning (DAMRC, Course compendium on Tool wear).

DAMC TOOL WEAR COUSE

Type	Image	Characteristics	Prevention of wear type
Flank wear		• Wear on the clearance, sides and surfaces of the cutting edge • Through inclusions of hard Constituents in the subject • Predictable and stable wear	• Reduce the cutting speed • Choosing an insert with stronger wear resistance • Increase feed
Crater wear		• Occurs on the rake face of the insert • Enhanced by the cutting speed = too high cutting temperature • Weakening of the cutting edge that can lead to breakage	• Reduce cutting speed and feed • Use a larger cutting angle • Choosing an insert with stronger wear resistance
Build-up Edge (BUE)		• Pressure welding of chips on the cutting edge. • For sticky materials (RS and ALU) • Due to too low cutting speed and/or an unsuitable quality of insert	• Increase the cutting speed • Choosing an insert with stronger wear resistance • Select other geometry of insert (sharp) • Finished surface (smooth)
Edge chipping		• Often by interrupted chipping • Chips are directed towards the cutting edge, this results in decomposition • Chip hammering • Overloading mechanical stresses	• Reduce cutting speed and feed • Use tougher and more stable cutting edges • Check stability
Plastic deformation		• Cutting temperature is too high at the same time as too high mechanical loads • The tool deforms when it becomes soft	• Reduce cutting speed and feeding • Reduce cutting depth • Increase coolant/lubricant supply • Choosing an insert with stronger wear resistance
Thermal Cracks		• Caused by temperature fluctuations from machining with interrupted chips/varying coolant supply • Occurs perpendicular to the cutting edge	• Reduce cutting speed and feeding • Choosing an insert with stronger wear resistance/ stable geometry of insert • Stop cooling in case of interrupted chips
Notch Wear		• Often occurs with hard-surface workpieces (forged/cast/RS) • Severe local damage to the rake and clearance face of the blade • Pressure welding of chips	• Reduce the cutting speed • Use varying cutting depths • Choosing an insert with stronger wear resistance • Use an insert with a smaller angle of engagement or a smaller corner radius

4.3 Tool wear detection and analysis in the industry

A minor market research has been made to understand how the companies work with tool wear today and which tools exists for helping analyse the tool wear of inserts.

Interviews with companies providing material for this project tells that machining companies have experienced operator using simple equipment like a loupe to evaluate the tool wear of inserts. The analysis is performed as part of the machining process setup for new parts, when the machining parameters have been determined it is expected that the tool wear remain the same between insert replacements.

A more detailed analysis is performed if the machining process is using a more expensive tool, e.g. a custom-made end mill. This analysis could be performed by the tool manufacturer, who will have high-level equipment for investigating how a tool can be renewed. This process is out of scope for this project.

Alternatives to manual analysing the tool inserts with a loupe is not many, but one example is the “Tool Wear” smartphone app from tool manufacturer, Sandvik-Coromant. The app helps the examiner capture a picture and allows comparisons with pictures of common tool wear types, for easy wear type determination. Figure 3 shows a screenshot from the app and a setup with a microscope lens clipped onto a smartphone. The app has no automatic recognition built into the app and acts only as assistance for the manual analysis.

The small market research shows two main points of interest:

1) Analysis of tool wear inserts is a used actively in the industry but is still manual and is performed with simple and low-cost equipment.
2) The manual analysis only considers types of insert wear, not severity.

A software for automatic detection of wear on insert has not been found during the research, why an AI4ToolWear model could prove beneficial in the industry. Development for other types of tools could be considered if the model shows good results for cutting inserts, as the market also needs this.

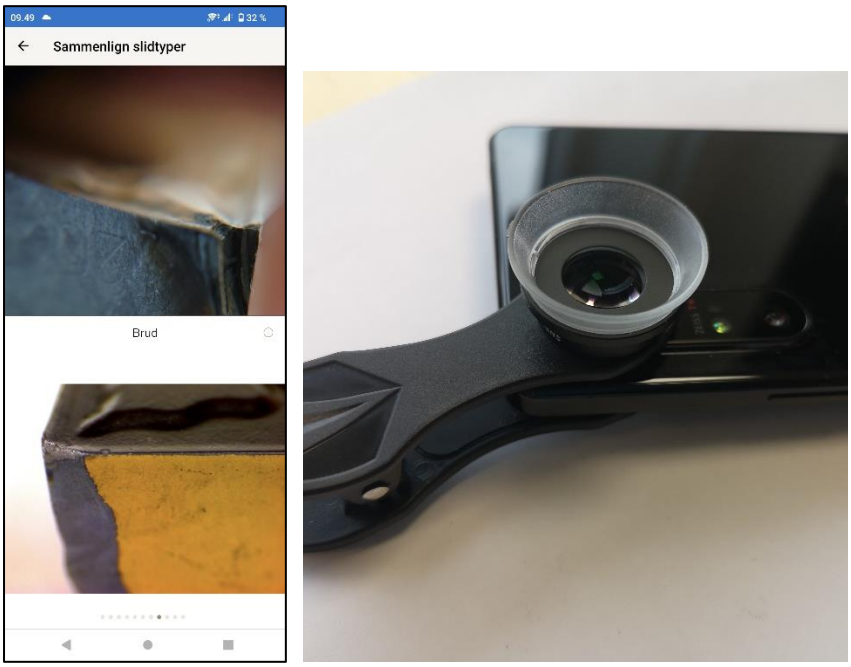


Figure 3 Left: screenshot from smartphone application for analysing tool wear ("Tool Wear" Sandvik-Coromant). Right: Optical lens clips for smartphone (Sandvik-Coromant)

4.4 AI and computer vision

Much well produced material about AI can be found on online blogs, video tutorials and books, hence this section will provide only an introductory description to AI and computer vision, to give the better understanding of the presented work in this report. Section 4.4.1 will give insight in why AI is escalating in use and as a topic in recent years, section 4.4.2 goes into the technical details which should allow better understanding of the computer vision models presented in section 4.4.3, which will be the models considered for this project.

4.4.1 General AI description

Artificial Intelligence (AI) has in the past years been a hot topic both inside and outside of research communities, likely due to the recent appearance of Large Language Models (LMM), from well renowned companies like OpenAI, Google and Meta. The capability of AI algorithms within various fields has also drastically improved and have come closer to human performance, as illustrated in Figure 4, [4] Zimmerman, A. 2023.

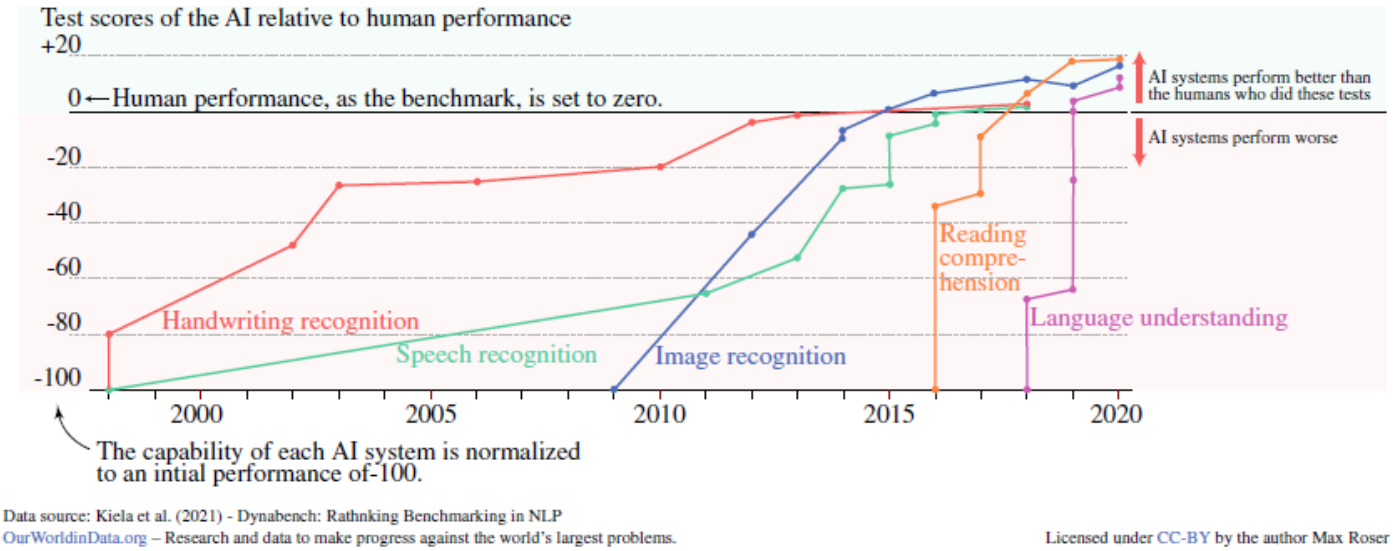


Figure 4 Illustration of AI performance compared to human performance within different algorithm types [4] Zimmerman, A. 2023

AI is the general term that covers any machine or software that behave like a human, subcategories of AI is Machine Learning (ML) and Deep Learning (DL). AI covers a broad area of research such as language generation, image generation and computer vision. The interest of this report is AI with computer vision, which covers object detection, segmentation, and image classification. Figure 5 illustrates how AI, ML and DL are linked and give a short description of each term.
The concept of AI dates to Alan Touring in 1956, but have accelerated in the past 10 years, due to the improvements of computer chip performance, hereunder graphics processing unit (GPUs), and development of large datasets for benchmarking, e.g. ImageNet, [6] en.wikipedia.org. Today most algorithms used and developed goes under the DL category, which uses neural networks.

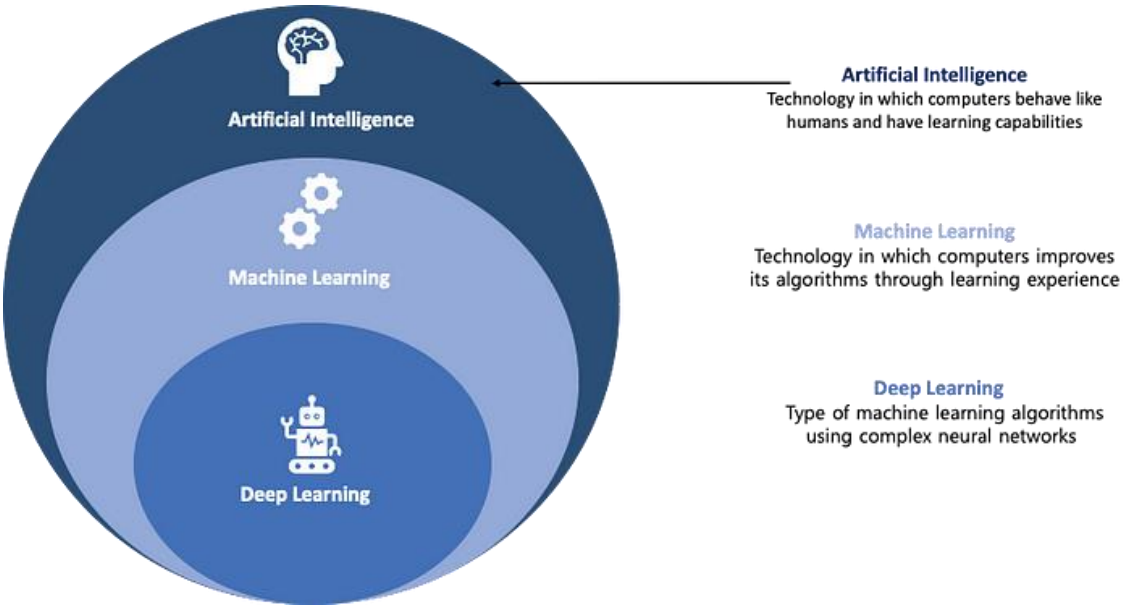


Figure 5 Illustration of AI, ML and DL, [7] dslab-global.medium.com

4.4.2 Working method of neural network for computer vision

As the name suggests AI algorithms is a computational model trying to imitate the brain, like DL using neural networks. Despite the parallels to biological neural networks, the artificial neural networks can only

be seen as oversimplified, as biological networks are still not fully known [12] Kröse and Smagt. The topic of neural networks is large and cannot be covered in full in this report, hence a simplified explanation will be given to achieve a basic understanding. As explained in [9] Baeldung, a neural network consists of artificial neurons, connected in multiple layers, forming the network. An artificial neuron is illustrated in figure 6, with input numbers, x_i, being summarized with weights, w_i, and added with a bias, b, and thereafter an activation function, ϕ(X), is used to decide whether the neuron should be activated or not. Mathematical the summation is expressed as:

$$y = \phi(\sum_{i=0}^{n}(w_i x_i) + b)$$

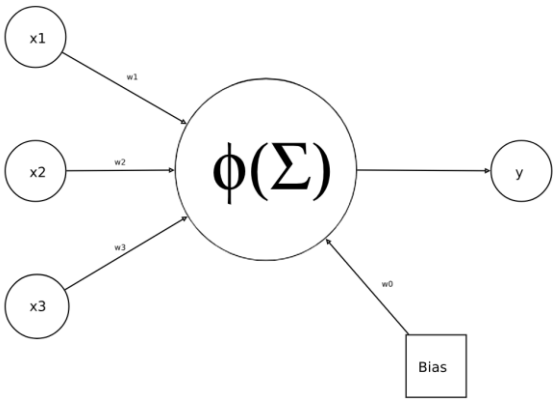


Figure 6 Schematic representation of an artificial neuron [9] Baeldung.

A DL neural network will consist of one input layer, one output layer and at least two or more hidden, as illustrated in figure 7. This is called a Fully Connected Neural network, FCN.

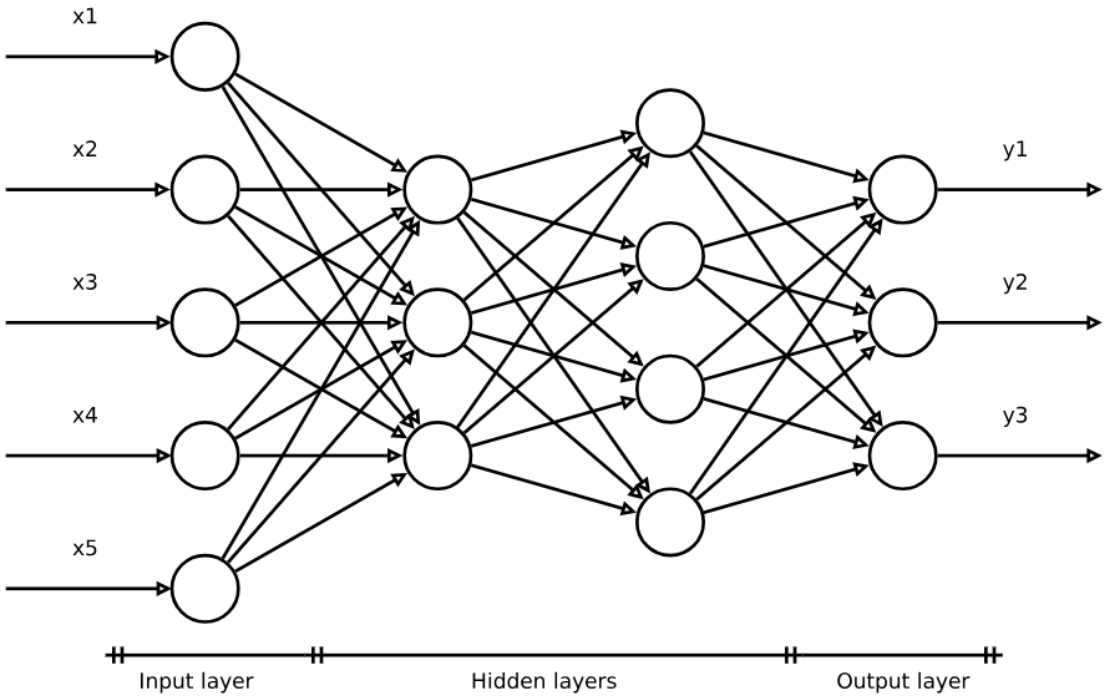


Figure 7 Fully Connected Neural network with input layer, hidden layers and output layer [9] Baeldung.

When training a neural network input numbers are processed in the network and the output is compared with the expected output, followed by correction of the weights and biases, i.e. the trainable parameters.

This training process can be done iteratively until a satisfactory output is achieved. In the case of computer vision, the use of FCN can be computational heavy. To illustrate this let's consider a 32x32 pixel RGB image being processed using an FCN having 2 hidden layers, and with four possible outcomes. The picture contains 1024 pixels and 3 numbers per pixel for the red, green and blue value, meaning the input layer will have 3072 neurons. Assuming the same number of neurons in the two hidden layers gives a total trainable parameters of 18.895.876. Calculation of the trainable parameters is as following, N_i, being number of neurons in layer *i*:

$$Parameters = (N_{in} * N_{h1} + N_{h1} * N_{h2} + N_{h2} * N_{out}) + (b_{in} + b_{h1} + b_{h2} + b_{out})$$

$$(3072 * 3072 + 3072 * 3072 + 3072 * 4) + (3072 * 3 + 4) = 18.895.876$$

To overcome this challenge a Convolutional Neural Network, CNN, can be used. The CNN type uses different methods to limit the number of trainable parameters, like Convolution Layer and Pooling. The convolution layer uses a nxn matrix to "sweep" over the input (image) matrix to extract characteristics like horizontal or vertical edges. This is illustrated in figure 8, where I is the 6x6 input matrix and K is the 3x3 matrix used for convolution, resulting in the 3x3 matrix, I*K

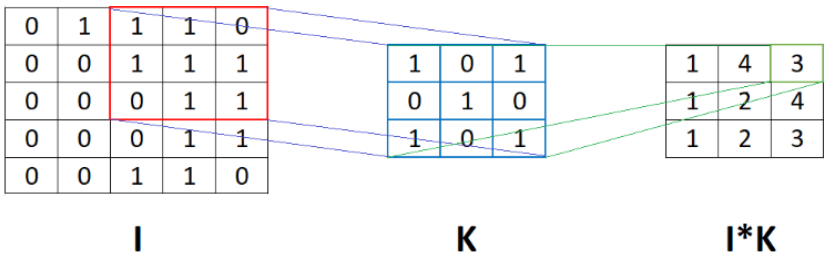


Figure 8 Convolution of 6x6 matrix to a 3x3 matrix [9] Baeldung

Pooling is an operation reducing the matrix dimension even further. E.g. Max pooling takes nxn sections of the matrix and reduces it by taking the highest number, assuming that this is the most significant from that section. The max pooling is illustrated in figure 9, with a 2x2 section. Another example of a pooling method is average pooling, which takes an average of the nxn section.

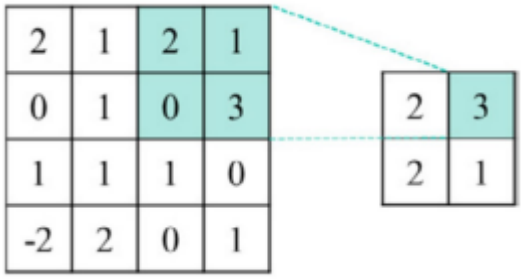


Figure 9 Max pooling operation [10] Bergs et al.

When the convolution layers and pooling have been used for processing, then a fully connected network can be used to perform the final prediction, as illustrated in figure 10, the flattening process take the multi-dimensional data and transforms it to a 1d vector, which is expected by the FCN.

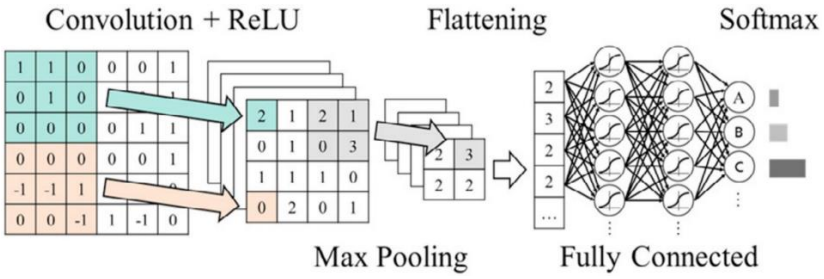


Figure 10 Illustration of a CNN with convolution layer, max pooling, flattening and fully connected network, FCN [10] Bergs et al.

4.4.3 Computer vision models

Many different architectures have been developed through time for doing different tasks within computer vision. Some of the, currently, most dominant ones within research are U-Net, YOLO (You Only Look Once) and most recently SAM (Segment Anything Model). The tree models solve different challenges, and covers the following categories within computer vision:

Image classification
The task of classifying what is seen on an image. A classification model would be able to detect one class of tool wear for each picture.

Object detection
The task of finding one or multiple objects of interest in an image. Classification of the object can be done as well. The result will be a boundary box around the detected object and a probability of the object being within a given class. Like image classification, the model will be able to detect one class per boundary box, but the benefit being that one image could have multiple objects detected, hence multiple classes detected.

Image segmentation
The task of classifying individual pixels, giving a segment of a given object. The result will be pixel-wise detection of an object, a mask, and the probability of the object being within a given class. In the context of tool wear the pixel-wise segmentation could be useful for estimating the V_B parameter, if the segmentation model can detect the flank wear accurately. This will require a calibrated setup, where the size of each pixel is known.

U-net and SAM are purely segmentation models, whereas YOLO have both Object detection, classification, and segmentation capabilities. Descriptions of the three models are given below.

U-net
The architecture was presented by [11] Ronneberger et al in 2015 and tried to challenge the need for thousands of labelled images to do successful training. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. The U-net architecture is shown in figure 11 was developed for the biomedical image segmentation but have since been used more broadly. The key advantages of the U-net model are that relatively few labelled images are needed and data augmentation, e.g. rotating or scaling existing labelled images, can be used to improve the detection performance.

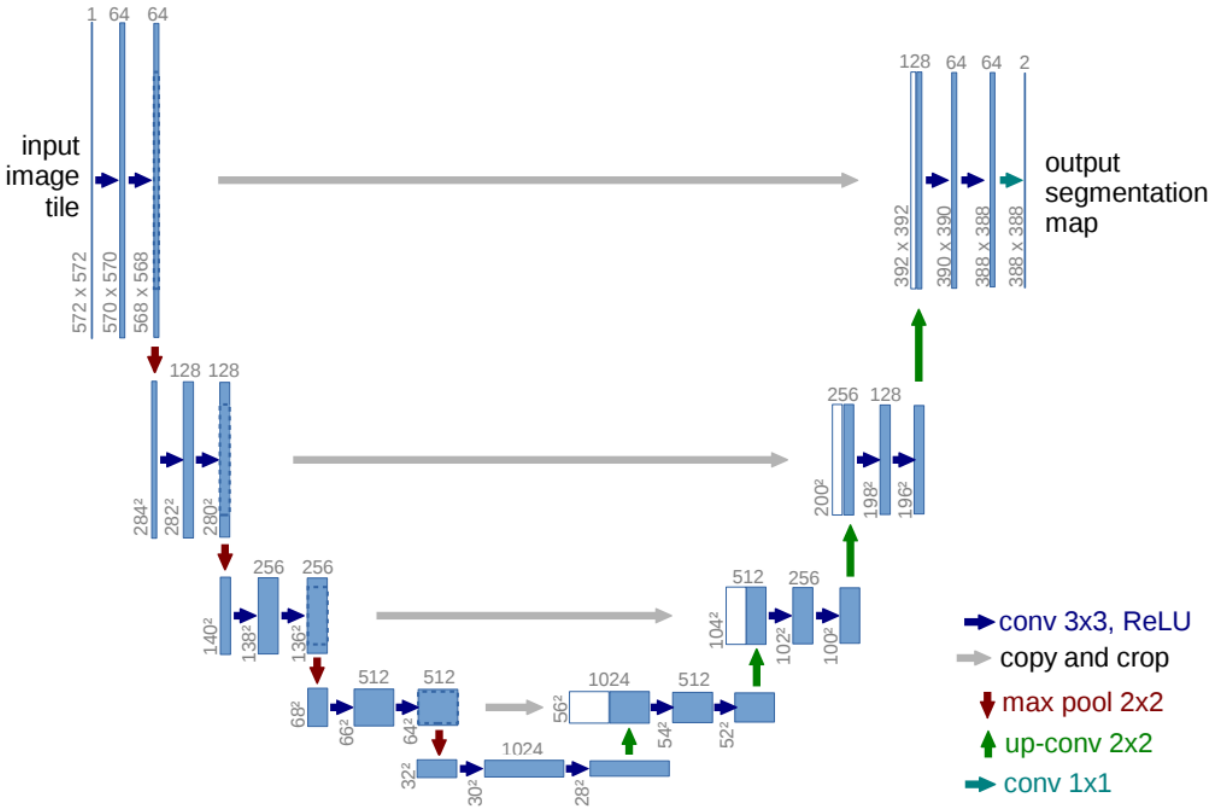


Figure 11 U-net architecture with encoding and decoding paths forming a U-shape [11] Ronneberger.

SAM

Segment Anything Model is the outcome of research from Meta, published in 2023 by [13] Kirillov, Mintun and Ravi et al. The model has been trained using the largest segmentation dataset to date, containing over 11M images with over 1 billion masks. This enables the trained model to predict masks of objects which is has not even been trained for. The model can be prompted using an API and provides a mask of an object, but it does not label the mask with any class. The model is promoted as a general segmentation model, similar to ChatGPT being a general language model.

YOLO

You Only Look Once origins from the idea of making a fast object detection model which is able to run real time on a video stream, and thereby be useful for robotics, self-driving vehicles and alike. It was presented in 2016 by [14] Redmon et. al, and was created as an open-source project. The company Ultralytics have continued the YOLO development, and presented YOLOv8 in autumn 2023. The Ultralytics version of YOLO exists both as open source and commercial license. Despite some from being fast, the Ultralytics YOLO model also provides an easy python API for training and using the model and provides good performance results when trained on the benchmark dataset COCO [21] Jocher et al.

Some of the architecture and models commonly used in recent studies are U-net, a segmentation architecture and YOLO (You Only Look Once), an object detection and classification model, and SAM (Segment Anything Model), a segmentation model trained with a huge multi labelled dataset.

4.5 Evaluation of a trained model

It is important to evaluate the performance of a model after and during a custom training, for doing this there are multiple parameters that can be used to understand if the training has been running as intended and if the trained model is trustworthy when used. Some of the parameters are described in below, to give better knowledge on interpretation of the results presented elsewhere in this report.

4.5.1 Dataset for training

A lot of data is needed when performing training of a neural network, hence it is essential to structure this data correctly. Most of the data is used for the actual determination of trainable parameters, another part is used for model validation, and yet another part is used for quality check. The split can be as following.

- Data set for training (annotated)
  - Training data: used for training the model's trainable parameters
  - Validation data: used for calculating the model's performance metrics
- Dataset for Quality Control
  - Interference data: used after training to check model performance with unseen data

4.5.2 During training

During training, it is of interest to see if the model is continuously learning and getting better at predicting the intended use case. Monitoring loss functions can help visualize if the model continues to learn and improve, if not the training should be stopped and dataset or training parameters must be improved. The concept of loss functions is giving a value for the discrepancy between the model output and ground truth, this discrepancy should become smaller, going towards zero, for every epoch trained. Different loss functions are used depending on the type of model, e.g. YOLOv8, uses Box and CLS loss for looking at discrepancies in the predicted output of coordinates of boundary boxes and classes respectively, as described in [23] Jocher, G. et al.

4.5.3 After training

After successfully training a model the evaluation of the model itself can be done. The metrics for evaluating the performance is calculated using the validation dataset. Some of these metrics that can be calculated are described by [24] Jocher, G. et al. and are described in brief here:

- **Intersection over Union (IoU)**: Used for segmentation and object detection models and gives a percentage of overlap between predicted mask or boundary box and the ground truth.

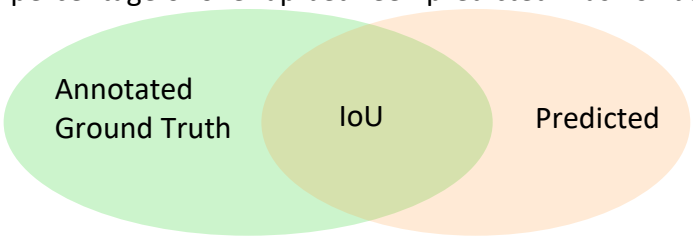


Figure 12 Illustration of IoU

- Correctness of predicted classes:
  - Ground truth (actual) vs. model output (prediction):

		Actual	
		Positive	Negative
Predicted	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Figure 13 Illustration of TP, FP, FN and TN

- **True Positive (TP)**: Refers to a predicted output being correctly predicted. I.e. a model making many TP predictions is good at finding what is being looked for, e.g. a model predicting the class Flank wear, and the annotated ground truth is Flank wear.
- **False Positive (FP)**: Refers to predicted output not being correctly predicted, either because it belongs to another class or it is just background, e.g. a model predicting the class Flank wear but the annotated ground truth being Breakage.
- **False Negative (FN)**: Refers to areas not being predicted with a class, but the annotated ground truth holds a class, e.g. the models gives no prediction where it should have predicted Flank wear. A model making many FN predictions is not good at finding what is being looked for.
- **True Negative (TN)**: Refers to areas not being predicted, i.e. background, and being background. This category is not of interest for classification models.

- **Precision**: Combines TP and FP metrics to evaluate how large a percentage of the correctly predicted classes among all predictions.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**: Combines TP and FN metrics to evaluate the percentage of correctly predicted classes with all the instances of classes that are in the annotated ground truth, and therefore should have been predicted.

$$Recall = \frac{TP}{TP + FN}$$

- **Confusion Matrix**: Combines the count of TP, TN, FP and FN in a matrix to get a visual overview of the model performance. A well performing model will have Confusion Matrix with a high count in its diagonal, indicating many TP and few FP and FN.

4.6 Tool wear detection with AI

Several researchers have studied how AI can help detecting tool wear and tool wear parameters. Some focus on indirect monitoring of the machining process to detect tool wear changes, e.g. by training a deep learning model on time series images from a dynamometer like [16] G. Martínez-Arellano et al. 2019.

For this project the direct monitoring is of interest, i.e. visual assessment of the tool, why a deeper review of four studies within this field will be given in following.

A model for automatic categorization of tool wear on insert is presented by [17] M. T. García-Ordás et al. The article proposes a computer vision model using a shape descriptor technique. The model is trained with a new insert data set of 212 images. An expert labelled the images in three categories, low, medium and high wear. The proposed shape descriptor, B-ORCHIZ, shows to outperform other shape descriptors when compared.

An automatic detection of tool wear and estimation of V_B index was presented by [18] T. Mikołajczyk et al. in 2017. This article proposes a segmentation algorithm, that can, pixelwise, detect tool wear of inserts and thereby calculate the V_B parameter by knowing the size of a pixel. Figure 14 shows an image before and after analysis. The study shows a good correlation between V_B index measured optically and automatically using the proposed algorithm on image of the cutting edge. It is noted that the light settings have a big influence on the predicted parameter and good light conditions are need, and maybe even need to be standardized, to make reliable V_B predictions.

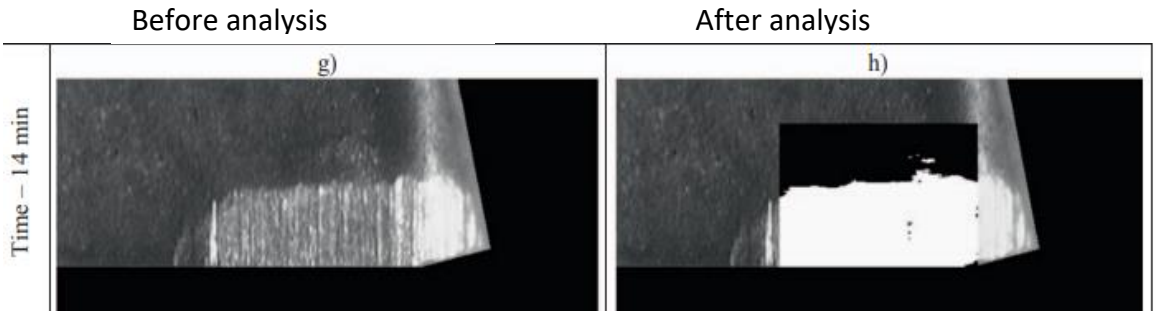


Figure 14 Example of analysis, [18] T. Mikołajczyk et al. 2017

In 2020 a two-step approach was presented by [19] T. Bergs et al. 2020. In the first step an AI model (CNN) is trained to classify the tool type (ball end mill, end mill, drills and inserts), the model achieves an accuracy of 95,6 %. The second step AI models, one for each tool type, are trained for semantic segmentation (U-net) to detect worn areas, achieving an Intersect over Union (IoU) of 0.7. The two-step process is meant for a machine integrated microscope to do automatic tool wear analysis, this process is shown in figure 15. 400 images were used in the study to train the AI models.

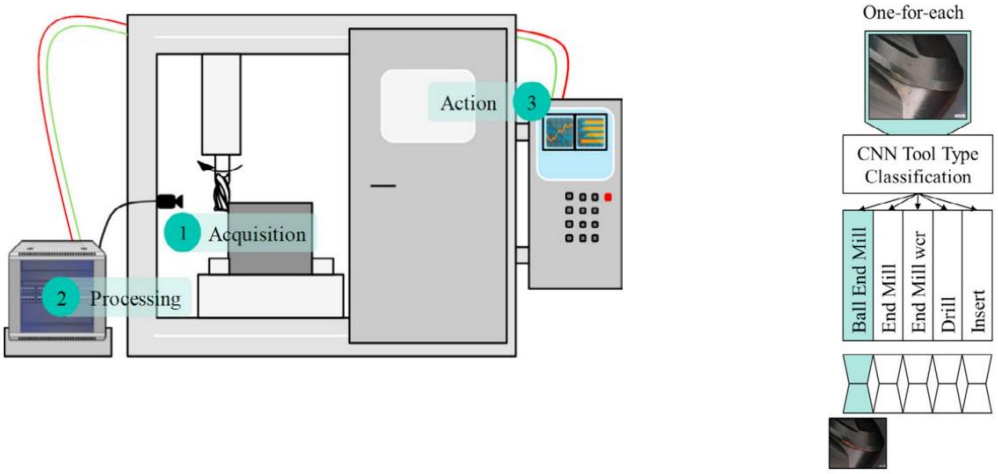


Figure 15 Illustrations showing machine integrated microscope (left), and the two-step process analysis (right), [19] T. Bergs et al. 2020.

The fourth study, [20] W. Lin-Ju et al. 2021, is combining object detection (YOLOv4) and images segmentation to detect the tool wear area of end mills. Three segmentation models were compared with

the result that U-Net obtained the best performance, compared to Autoencoder and Segnet. The idea of the combined process is reducing the need for computational power by using the efficient YOLOv4 model to detect the area of interest and then use the more computation heavy segmentation model for detailed analysis of the tool wear. An image stitching technique is used to overcome the challenged of the end mill being cylindrical, and this allow to create one large panorama image for each tool. 100 images of end mill tools were used in the study for training the AI models. Figure 16 illustrate the process from the article.

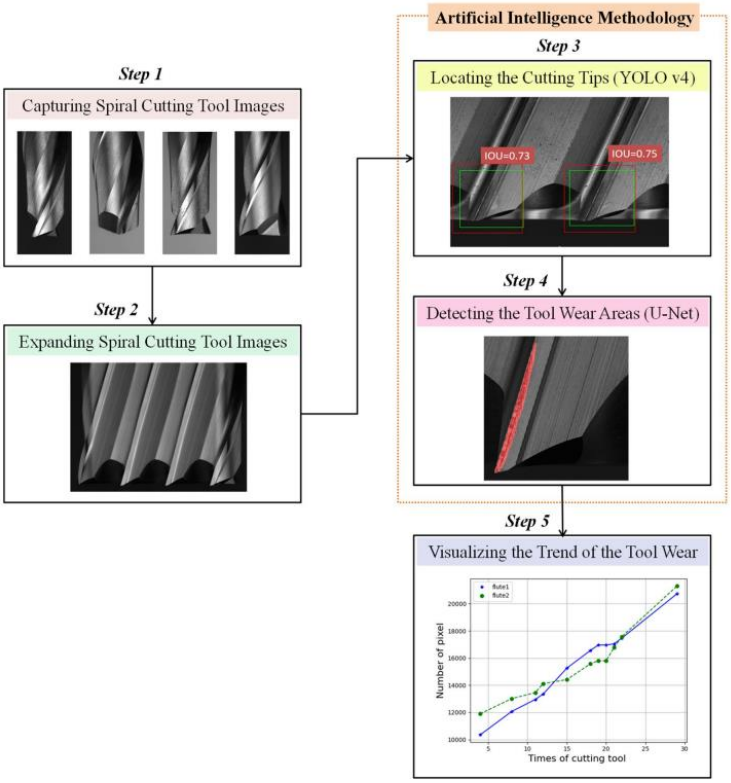


Figure 16 Illustration of the process proposed in [20] W. Lin-Ju et al. 2021

The four studies propose different approaches to solve the issue of analysing tool wear, utilizing both classifying, object detection and segmentation models. They all present trained AI models that seems to perform well, all four articles use or propose the use of controlled light settings and the same type of microscope for capturing images for training and interference (using the model for predictions).

4.7 Opportunity for improvements

Small tests are conducted to evaluate opportunities with the different AI models presented in section 4.4.3. The focus, in the tests, is also on understanding the software tools used for training and using the different models. Three models are investigated at this stage and a resume of the tests are given below:

4.7.1 ZeroCost4DL (U-net Segmentation model)

The ZeroCost4DL presented by [22] Hidalgo-Cenalmor et al. is developed to allow easy training of AI models for researcher. The project includes multiple different models, where the U-net segmentation model is of interest. The result of the evaluation test shows that segmentation of tool wear is possible using the U-Net model, but a much-improved dataset is needed to create more accurate masks. The predicted output is a masked image with coloured pixels, predicting background or tool wear area. Figure 17 shows a sample from the training process, with both annotated image (ground truth) and the predicted mask, with IoU of 0.598, meaning an overlap of 59.8 % between ground truth and predicted mask.

The benefit of predicting the tool wear pixels-wise, is the possibility of calculating wear parameters, e.g. V_B and V_{BMax}, like the method presented in [18] T. Mikołajczyk et al. A calibrated setup, with known size of the pixels, is needed to calculate the parameters. As a AI4ToolWear model, the segmentation method is not considered as good solution currently, due to the strict requirements for the microscope setup and the foreseen manual interactions needed to accurately calculate wear parameters.
Details on the model and the training process with dataset2 is given in Appendix B.

Figure 17 Sample from training process, from left: input source image, annotated mask, prediction by the model and overlay of annotated and predicted mask.

4.7.2 SAM (Pretrained segmentation model)

SAM is developed and trained with the idea of doing image segmentation of any type of objects in an image. The SAM python API allows prompting the model to narrow down which area the mask should be predicted. The prompting can be points or boxes indicating masks to include or exclude. Examples of this is given in figure 18 shows examples of point and box prompting and the resulting predictions by SAM in blue colour. The testing shows impressive prediction, considering the model has not been trained specifically for tool inserts, and most likely there has not been any tool insert images in the dataset of 11 million images. Similar to the U-Net segmentation model, SAM is not considered suitable for an AI4ToolWear model, due to the need for prompting. SAM will likely be beneficial to integrate in many software programs for analysis, and this development should be followed closely.
Details on the SAM prompting test can be seen in Appendix D.

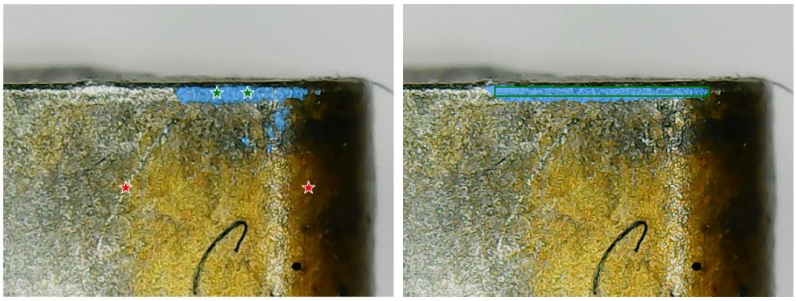


Figure 18 Left: Prompting with inclusion points, green stars, and exclusion points, red stars; Right: Prompting with box to include, green.

4.7.3 YOLOv8 (Object detection with classification)

A dataset of tool inserts, dataset3.0, have been created to try the training process of YOLOv8. YOLOv8 has different model types, where the one for object detection and classification has been used. The result of the training with dataset3.0 is a model that can predict a boundary box around a specific class, e.g. flank wear or edge chipping. Figure 19 shows an image where the four classes has been predicted and boundary boxes are put around them. An object detection model will only be able to give insights in the classes and the number of their occurrences and cannot calculate wear parameters. The benefit of object detections models is the possibility to train them variety of images (zoom level, angle, light setting), and likewise the

microscope setup is not required to be calibrated. This gives a flexibility for the machinists who must perform the analysis. Furthermore, the tool wear class is already used in the industry to optimize machining parameters. Using an object detection model could improve the efficiency and reproducibility of today's manual class predictions.
Details on the training and the dataset can be seen in Appendix F and Appendix G.
The YOLO framework from Ultralytics is a powerful tool for training a high performing image recognition AI. The framework is by default opensource software but can be integrated in commercial products for a yearly fee.

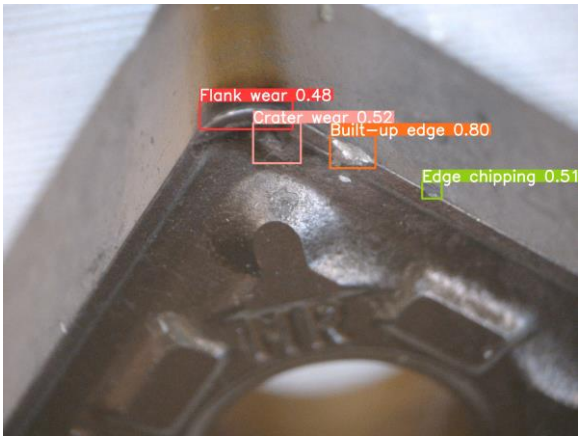


Figure 19 Image with boundary boxes around predicted classes, from trained model.

4.8 Conclusion on the Pre-Analysis and Literature Study

The pre-analysis and literature study shows that object detection and classification (utilizing YOLO) will be the best option to gain knowledge on the training procedure and obtaining results that can evaluate the AI technology. The focus should be training a model that can classify images from low-cost microscopes, to make the process assessable for most machinists. The results and knowledge from training an object detection model, is seen useful also if a segmentation model is later to be developed.

5 Experiment Design

5.1 Introduction

The experiment will investigate how well a model performs, when trained on images from low-cost, low-quality microscopes. This will give insights in how a data set is prepared, and which parameters that can improve the model performance. Furthermore, the performance of the models can be evaluated, to investigate if low-cost microscopes can be used for tool wear identification.
The result of the experiment will be a dataset of tool inserts (annotated images), and an object detection model that can detect common tool wear type.

The experiment will consist of three tasks:

1. Data set creation
2. Iteratively training and improve the object detection model

3. Evaluate the best model with low-cost microscopes

5.2 Test design/process

The test will use an interference data set from two types of microscopes, one digital microscope (Bresser DST-1028 5.1 MP) and a microscopic lens clip for smartphones (APEXEL 2 in 1 12x/24x Macro Lens). Predictions will be made with a custom trained model, which has been prior chosen from all trainings performed.

The process from the data set creation to model evaluation is described in figure 20. The arrows going backwards illustrates the iterative process done when training a model.

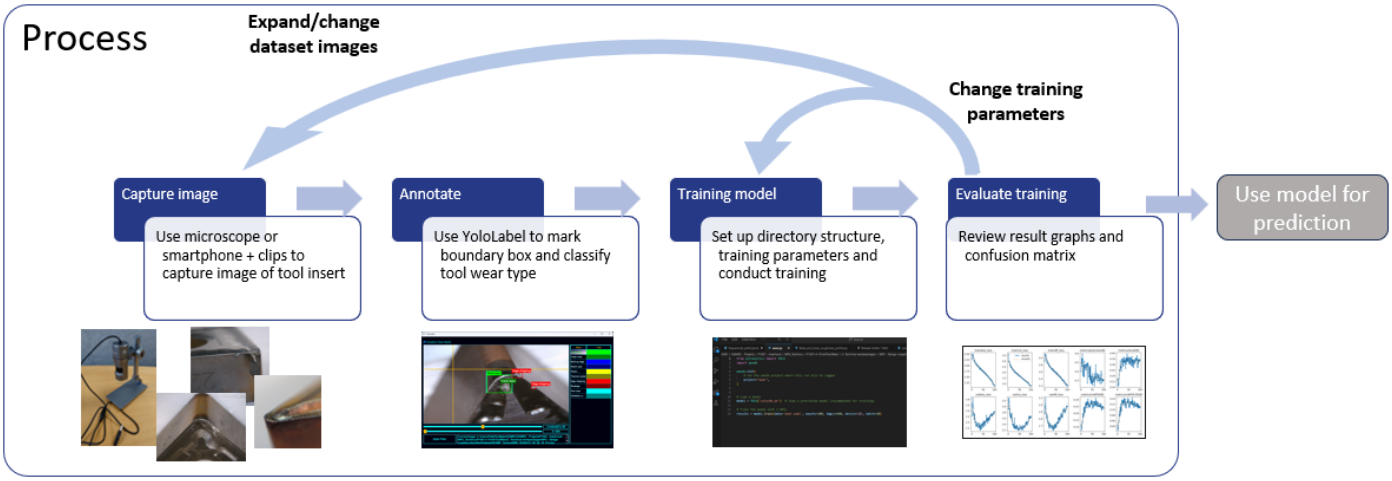


Figure 20 Process for the training procedure used for the experiment.

The trained model will be evaluated with the interference data set, and focus will be on:

- Number of errors detected/not detected (FN vs TP)
- Differences between method of image capturing (microscope clips and digital microscope)
- Model performance (confusion matrix) vs. experienced performance

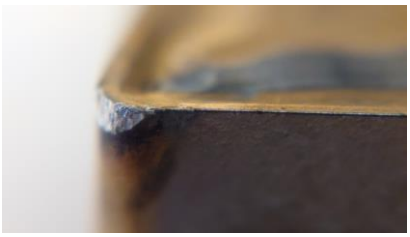
5.3 Equipment for the test

- Microscopes (1x Digital USB microscope, 1x Smartphone + add-on lens)
- PC with GPU
- Annotation software
- Python script for model training

5.3.1 Microscopes

Two different methods are used to capture images for the training data set and interference data set. Both are relatively low cost, one being an add-on lens to any smartphone giving additional 24x magnification, and the second being a Digital USB Microscope which can be added to a PC and which has an adjustable

magnification of 10x to 280x. Figure 21, show the method and the details about the setup.



Setup details
Smartphone: Xperia XQ-AT51
Lens: 70 mm
image size: 4032 x 2268 px
Lens clip: APEXEL 2in1
Magnification: 24x
Price: ~30 EUR

Setup details
Digital Microscope: Bresser DST1028 5.1
Magnification: 10x – 280x (x40-x120 used for datasets)

Image size: 2592 x 1944 px
Price: ~200 EUR

Figure 21 Top: Add-on lens for any smartphone and sample image. Bottom: Digital microscope and sample image.

The benefit of the smartphone add-on lens is the low price and easy accessibility. The tool wear analyst, e.g. a machine operator, only needs their smartphone and the add-on, to capture images almost anywhere. Experience with the add-on lens shows that it has a very small focus area and needs good light condition and a very stable hand or optimally a smartphone stand could be used.

The digital microscope has a higher price, still relatively low, but allows for a better magnification level and more static setup than the smartphone + add-on. The microscope has a built in LED light, nevertheless good light conditions are still needed.

The orientation of the tool inserts is slightly variating between the images, with the basic rule of trying to capture both Clearance face, Cutting Edge and Rake face. A sample of the captured images are seen in figure 22.

Figure 22 five images from the data set with images captures by smartphone + add-on and digital microscope.

5.3.2 PC with GPU

A PC with a graphical processing unit, is preferred when running training of AI models with large data sets. For this experiment the training has been done using:

GPU: NVIDIA Quadro P2000	• Memory 5 GB GDDR5 • NVIDIA CUDA cores: 1024

5.3.3 Annotation Software

For annotating the tool wear on the images an opensource annotation software is used, named YoloLabel. This software is built for creating annotated data sets prepared for training with the YOLO framework.

5.3.4 Python script for training model

This experiment uses the YOLO framework for Python, to train an AI model. YOLO provides different sizes of pretrained models, ranging from nano to x-large [15]. The YOLO models are trained and later used for predictions using python scripts.

5.4 Material for the test

- **Worn tool insert**
  Used tool inserts have been kindly lent from collaborating machining companies. The inserts have been used for different materials and different machining operations. This gives a variation of tool wear types and insert types in the data set, but the relatively low number will not be enough to cover all machining types and tool wear types sufficiently.
    - Used tool inserts: 214 pcs
- **Data set description**
  Multiple pictures is captured from each of the 214 tool inserts as each insert have multiple used sides and different angles and microscope is used to capture the images. In total, five datasets have been created from the images, differing the number of images used, classes of tool wear types used and annotator of the images.
  The dataset used for training the model investigated in this experiment is named Dataset5.0, consisting of 1784 images. The dataset is split in images for training the model and images for validating the model during training, with a split of training: 89%, and validation: 11%. Figure 23 shows how the instances of classes are represented in the images for training.
  More details about the datasets are found in Appendix E

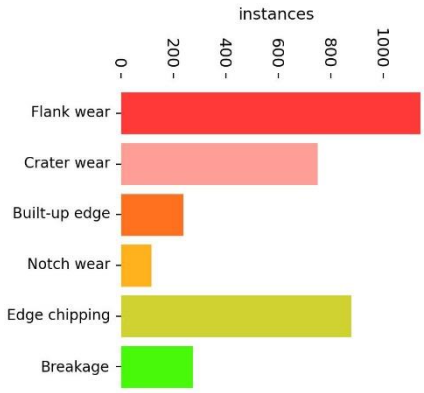


Figure 23 Distribution of classes in the 1396 images used for training.

- **Interference dataset**
  An interference dataset, unseen images of worn tool inserts, is needed for evaluation of the best performing model. The interference dataset is created using the digital microscope and the smartphone with microscope clip. All inserts used in the dataset has wear of some type. The split of images is:
    - Digital Microscope: 18 images
    - Smartphone + clip: 14 images

The primary material used for training an AI to do object detection is annotated images. The quality of this data is important, i.e. the image quality, the correctness of class annotation and boundary box annotation. Another important factor is the splitting of the dataset in images for training and images for validation. The validation images and its annotations are used for calculating the model performance parameters.

5.5 Conduction of the test

The iterative training process illustrated in figure 20 has been followed and have led to creation of the five datasets describe in previous section. In total 12 trainings have been conducted, and the 12th training, train50, have been chosen for evaluation. Train50 is using Dataset5.0 + data augmentation of Dataset5.0. An overview of all 12 trainings can be found in Appendix F.

Table 3 Specifications for the 12th training

12 – Train50 specifications	
Training time	10 hours
Dataset used	Dataset5.0 + data augmentation
No. of trainable parameters in model	25.9 million
Classes in dataset	Flank wear Crater wear Built-up edge Notch wear edge chipping Breakage

5.5.1 Evaluation of training

The results output from the YOLO framework is used to evaluate the training of train50. The results graph is shown in figure 24. The loss shows a continuous decrease in value, which indicates that the model keeps learning during the 100 epochs. The trend lines of the metrics also show mainly linear improvements, but a stagnation of the improvement is seen from approximately epoch 80 for recall and mAP50-95 parameters, indicating that the model might start overfitting. Overall, the training is learning as expected using the dataset.

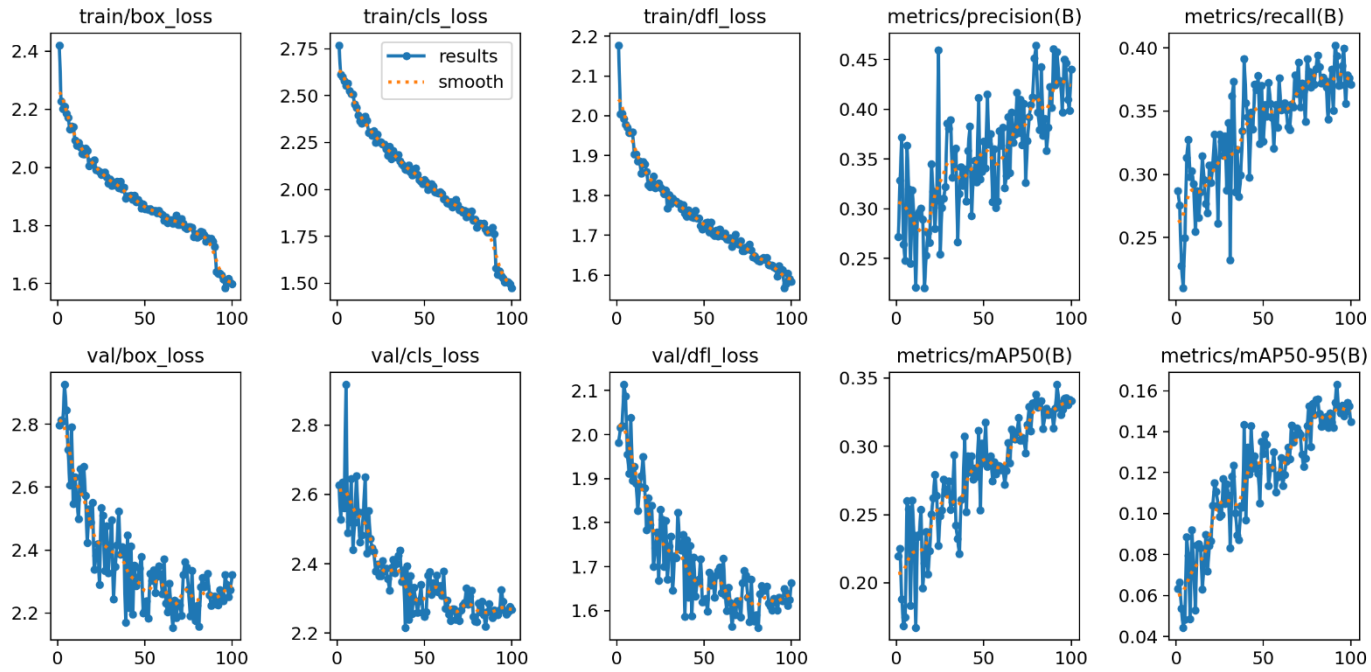


Figure 24 graphs of trends for loss parameters and metrics (precision, recall and mAP50/50-95) for train50.

5.5.2 Evaluation of model performance

The model performance metrics, in terms of well it can predict the classes, is calculated after a training using the validation dataset. Highlighted metrics for the evaluation is mAP50-95, precision, recall and the Normalized Confusion Matrix.

The Confusion matrix visually shows if the model predicts the right classes, True Positive (TP), or if it predicts the wrong class, False Positive (FP), or if a tool wear occurrence is not predicted at all, False Negative (FN). The Normalized Confusion Matrix for train50 and the validation dataset is shown in figure 25. The matrix shows that TP predictions are accounting for almost 1/3 for each class. E.g., for Flank wear, 31 % of the model's predictions are truly Flank wear, and only a total 4 % of the predictions is wrongly classified as another wear type. This leaves approx. 64 % of the flank wear unpredicted, which then falls into the Background class, i.e. no detection of tool wear.
The best prediction success is found for Breakage, with 51 % being predicted correctly, and only 8 % is seen as background, but the remaining 41% predictions are as they are wrongly predicted as Edge chipping. The high prediction rate is likely because of the distinct look of breakage making it easier for the model to detect.
For occurrences of Notch wear the predictions almost split equally between Edge chipping, Notch wear and Background. The high percentage of wrongly classified predictions, i.e. Egde chipping predicted but the classes is annotated as Breakage or Notch wear, is huge downside of the model and could lead to mistrust from the operator. The wrong predictions of Background, no tool wear, is seen less critical as this can be due to bad image quality, and the operator would try capture a new picture.

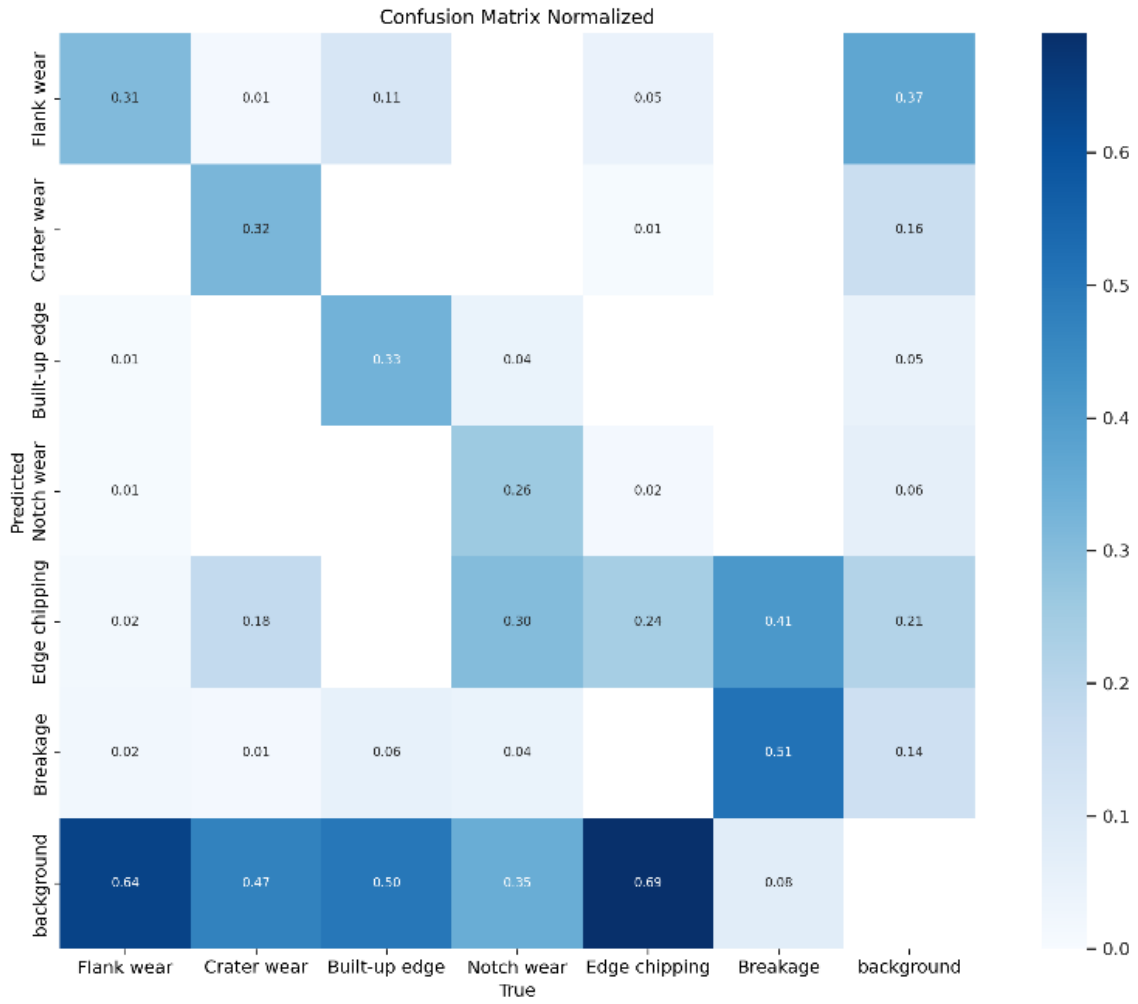


Figure 25 Normalized confusion matrix of result from validation prediction in train50.

The Confusion Matrix gives a nice compact overview of the model performance, apart from the matrix other metrics are also interesting to review, when evaluating the model performance. Table 4 lists the average and class specific metrics for the train50 model and the validation data set. Here mAP50-95 is seen as a good measure of how accurately the model can predict. Comparing the mAP50-95 of the train50 model, 0.153, with a model trained on the COCO-dataset [21] Jocher, G. et al., 0.502, the train50 model still has room for improvement.
The Precision metric for train50 tells that the model predicts correctly, TP, only 40% of the times, and opposite it predicts a wrong class, FP, over 50 % of the times. This means the operator most likely cannot trust the prediction that is given by the model.
The recall metric for train50 tells that 39% of the annotated wear in the validation dataset is predicted as the right class of tool wear, and 61% of the annotated wear is not even detected, i.e. seen as Background.

Table 4 Performance metrics of validation dataset in train50

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
All	197	456	0.403	0.394	0.323	0.153
Flank wear	197	175	0.371	0.343	0.255	0.0823
Crater wear	197	72	0.332	0.297	0.246	0.0906

Built-up edge	197	18	0.495	0.444	0.375	0.166
Notch wear	197	23	0.469	0.391	0.385	0.16
Edge chipping	197	129	0.271	0.271	0.15	00503
Breakage	197	39	0.48	0.615	0.528	0.362

5.5.3 Prediction on interference dataset

The interference data set consists of 40 images, captured by the two types of microscopes. The images are analysed with a python script using the train50 “best.pt” model. Results from the predictions are summarised in table 5 and can be found in full in Appendix G. All images in the dataset have visual tool wear, and it is expected that the detection rate will high. Comparing the two capturing methods, it interesting to note that the digital microscope both have the highest percentage of images with detections and detects multiple wear classes on more images, than images captured by the smartphone + microscope clip.

Table 5 Resume of predictions of interference dataset

Digital Microscope 18 images	Smartphone + microscope clip 22 images
Images with detection: 15 (83 %)	Images with detection: 12 (55 %)
Images with >1 detection: 6 (33 %)	Images with >1 detection: (9 %)

This difference in detections can, amongst other reasons, be caused by:

- The digital microscope allows a higher magnification level
- The setup of being more stable for the digital microscope
- The microscope clip has a small focus point, leaving most of the image blurred
- Lighting conditions are different between the setups
- All 40 images in the dataset are unique, hence difference will appear
- Dataset5.0, used for training of train50, contains to few images from smartphone + microscope clip. Digital microscope: 1396 images (88%), Smartphone + microscope clip: 189 images (12 %)

In general, the digital microscope performs best regarding the number of predictions, and it seems more difficult to capture easily predictable images using the smartphone + microscope clip. Examples of images predicted by the two methods are seen in figure 26.

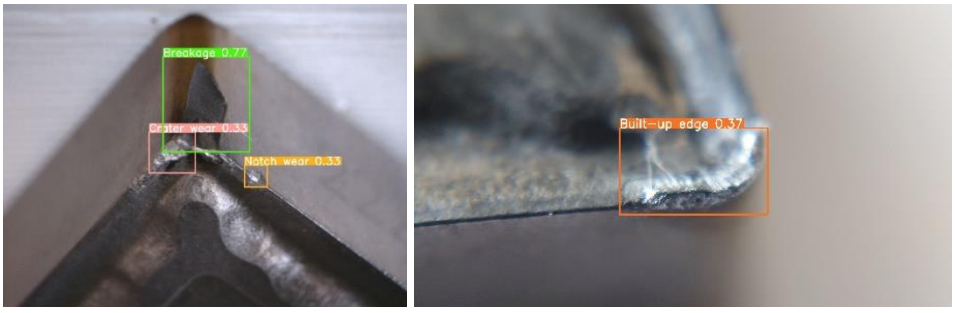


Figure 26, Predictions with train50 model, left: digital microscope image, right: Smartphone + microscope clip images.

6 Conclusion of the Test Results

The test is focusing on evaluation of a trained model for predicting tool wear, and how well this model can be used to predict tool wear using two proposed image capturing method, namely using a digital microscope or a smartphone + microscope clip. The three parts of interest in the conclusion section is 1) the created dataset, 2) the training process, 3) image capturing process.

The test evaluates a model trained using dataset5.0, which is an outcome an iterative training process. Dataset5.0 contains 1595 annotated images in total. The distribution of annotations is uneven in the data set, with most instances of Flank wear: +1100, and Edge chipping: +800. The number of tool wear classes have in dataset5.0 been limited to six classes to avoid classes with very low instances. Dataset5.0 is still considered large enough to train a model for evaluating the potential of using an AI model for tool wear analysis.

The YOLOv8 framework is used for the training process, and has proven competent to train a model, using dataset5.0. The trained model does predict tool wear classes correctly, but with a too high percentage of wrong predictions and Background-predictions. Comparing the model's performance metrics with the COCO dataset, shows significant difference in performance, which indicates room for improvement of a AI4ToolWear model, before industrial use. The training process is successfully used as an iterative process of improving the training dataset.

The final part of the test evaluates the performance of two different methods for capturing images, both relatively low-priced investments. The methods has a difference in the image quality, with the digital microscope capturing visually better images than the smartphone + microscope clip. The smartphone + microscope clip setup requires some patience for capturing images that are not blurred. The predictions of images from the two methods shows that the digital microscope can give most predictions (83 % of the images), probably due the fact that most images in Dataset5.0 is captured using this method. Both methods can predict tool wear, hence they both have potential in a later industrial trial of the AI4ToolWear model.

7 Discussion

This project gives insight in how an AI model can be trained and utilised for the tool wear analysis. There is still much work to be done before an AI4ToolWear model can be used for industrial trials and later implemented as part of the analysis of the machining process. This section will try highlight some of these working areas.

The test shows potential in using an AI model in the analysis of tool wear classes of inserts. The evaluation does show that there is still work needed in improving the dataset for training and validation. This includes getting more inserts, images of different capturing conditions, images some different capturing methods and better representation of all tool wear classes. An improved dataset should lead to an improvement in model performance considering precision, recall and mAP50-95.

The focus in the project is narrowed down to training of an object detection model, which can classify tool wear types. This model will therefore not be able to analyse the severity of the flank wear, V_B or V_{BMax}. This will require a new annotation process for training a segmentation model and furthermore require a calibrated images capturing setup, as the segmented pixels need to be correlated with actual size in mm. The class-wise predictions on its own, presented in the project, will most likely benefit the machining companies in their process, and a severity analysis could add even more value to the companies in the

effort to make data driven decisions and optimize the machining processes.
Only the YOLOv8 framework has been used for the training process and comparison to other frameworks will be beneficial to understand if the choice of framework is correct, in terms of price and performance.

This project presents two proposed methods for capturing images, which are both relatively low price. The difference in the method being how and where the capturing process can be done. Further market investigation is needed to understand which method is suiting the machining companies the best.

8 Conclusion

This project has worked on developing the DAMRC's own AI4ToolWear model, which is the idea about making the powerful AI tool available for the machining industry.

The result is a computer vision model for doing object detection and class-wise detections of six tool wear types. Flank wear, Crater wear, Built-up edge, Notch wear, Edge chipping and Breakage. The model is custom trained to detect where the tool wear area is on an image and then place a boundary box around it and classify the tool wear type. The performance metrics precision and recall are two of the parameters used for evaluating the model performance and these shows that 40 % of the model's detections are correct, and 39 % the tool wear is detected by the model. A better performance is likely needed to gain a trustworthy and valuable model, that the industry wants to use.

The success criteria for the project are seen as fulfilled as fundamental knowledge on AI has been establish along with establishing the dataset of 214 inserts and 1595 images and using these in the iterative training process. The project has also shown that the large amount of research presented in recent year within the area of AI is possible to transfer to the domain of tool wear analysis.

For the next step, it is suggested to continue the work of improving the project's AI model, meanwhile investigating how the analysing capabilities can be used by the industry, by doing user tests with relevant companies. A thoroughly market research has not been conducted as part of this project; hence this is suggested, when going forward, to understand better the opportunities for exploitation, either as a DAMRC service, tool wear analysing application or a third option.

9 Dissemination

Planned activities for dissemination of project results are:

- Making the technical report (this report) publicly available on DAMRC's website
- Including the project topic as part of DAMRC's course "tool wear"
- Create a post about the project on DAMRC's LinkedIn page
- Present the results at DI and DAU seminar at DAMRC in November 2024

10 Appendix A – Dataset2

10.1 Data set description

Dataset2 is the first data set with tool images for testing of the AI models. Dataset1 consists of non-relevant images and has been used for setting up data preparation and model trials, thus not further described.
Dataset2 consists of microscope images of tool inserts. In total the data set holds 85 images, captured with 137X magnification.

The tool inserts have been oriented perpendicular to the microscope, to allow capturing of the flank wear.

Each tool insert can have several edges used for machining; hence each insert can be represented on multiple images. In total 46 tool inserts are part of Dataset2.

The images are split into Training data (75 images) and Quality Control data (10 images).

10.2 Microscope setup

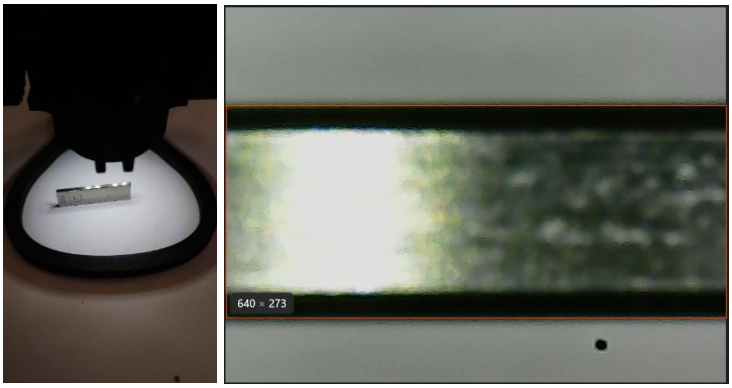
A low priced “nolabel” microscope is used to capture the images. This is chosen to resemble the image quality that likely will be sent in from various people in the machining industry.

Zoom (magnification)	137X
Resolution	0,0073 mm/px
Image size	0.3 MP (640x480)

Magnification calculation

A gage block of 2 mm is used. A picture of the gage is captured, and the pixels are counted to calculate the pixels per mm (magnification).

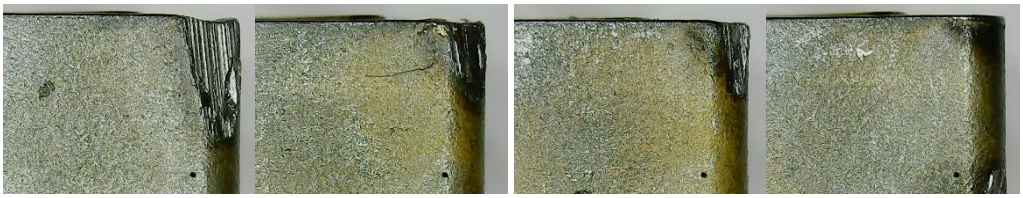
$$Magnification = \frac{273\ px}{2\ mm} = 137\frac{px}{mm}$$



10.3 Sample images

Four of the 85 images are shown in the next sections. Firstly, the original images, secondly the annotated images to be used as ground truth during training and finally a gray scale version that can be used if the given model requires it.

10.3.1 Original images – source images

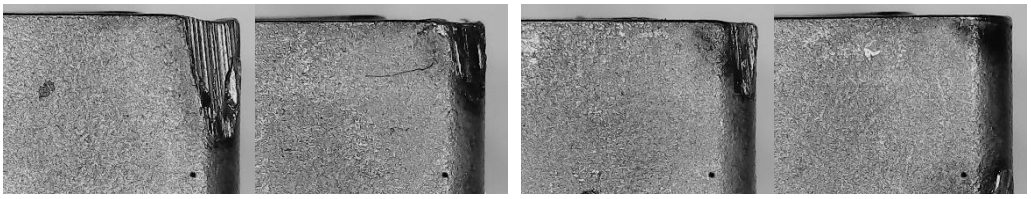


10.3.2 Annotations – Mask images

Original images are annotated using the LabelMe software by defining a polygon around the tool wear area. The polygons are used to make a mask of white and black, and saved in 8-bit TIF format. White pixels indicate the tool wear and black pixels are anything else. The annotated images are used as ground truth for the training.

10.3.3 Gray scale images – Source images

All original images are converted to grayscale 8-bit TIF format, to enable use in some AI models. Source images are used together with masked images to train the models.



11 Appendix B - ZeroCostDL4mic

11.1 Google colab

ZeroCostDL4Mic is utilizing Google Colab to share its deep learning models. Google Colab is an online tool based on Jupyter Notebook, which allows coding in notebook style and doing so in an online environment. The benefit of this is that all computer power is from a server, which allows for training of neural network on high performance GPUs. Moreover the google Colab is easy to share amongst other collaborators and each notebook creates its own environment with all needed dependencies. In short Google Colab aims to lower the entry barrier for researchers and developers doing AI modelling and training.
link: https://colab.research.google.com/

11.2 Notebook description

The ZeroCostDL4mic Google Colab notebook is a project available on
https://github.com/HenriquesLab/ZeroCostDL4Mic/wiki

The description by the creators themselves:

"ZeroCostDL4Mic is a toolbox for the training and implementation of common Deep Learning approaches to microscopy imaging."

The work origins from the research group Optical Cell Biology (Henriques Lab), placed at Instituto Gulbenkian de Ciência, Portugal (formerly placed at University College London).

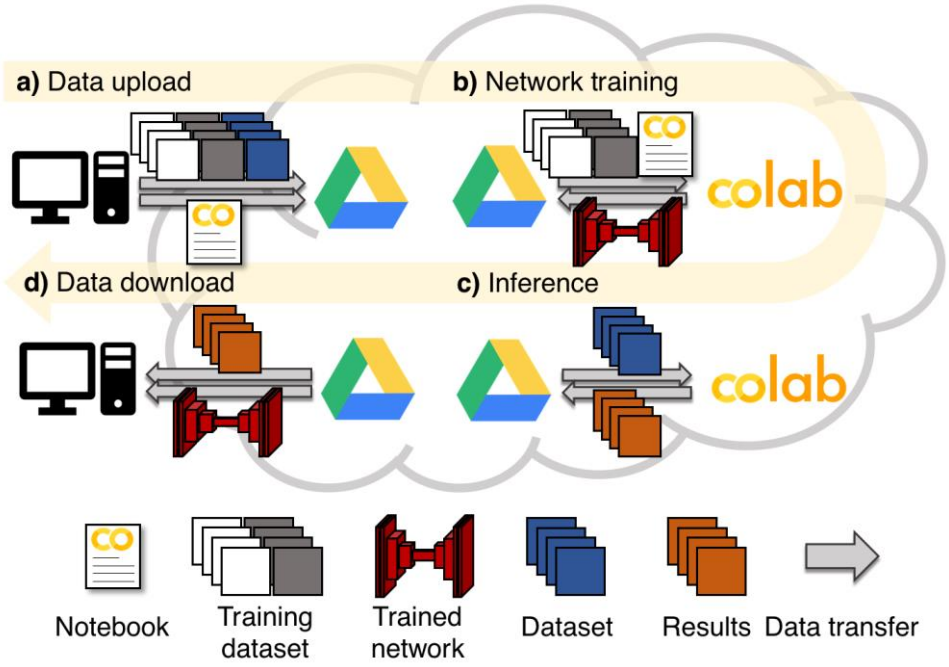


Figure 27 Overview of training process using ZeroCostDL4Microscopy

The ZeroCostDL4Mic has Google Colab notebooks for multiple deep learning models able to do segmentation of images. Figure 27, illustrates how projects is used for training and prediction.

Fully supported models, found on the Github repository:

U-Net (2D, 3D, instance segmentation, semantic segmentation)
StarDist
Noise2Void
CARE
Label free prediction (fnet)*
Object Detection (YOLOv2)
pix2pix
CycleGAN
Deep-STORM

To run the U-Net training procedure, the Google Colab notebook must be saved in a private Google Drive. The notebook describes the process and guides on the training and data set up.

11.3 Training data and procedure

The U-Net model is per default not trained, and therefore needs a dataset for training. The Data set will help the model to determine the weights that best detect the texture being looked for.

The dataset for training consists of images of tool inserts captured with a microscope. The images must be 8-bit and preferably TIF format.
Each source image (captured by microscope) should have a related mask image (truth of desired prediction, which is used for comparing)
The training dataset will need to be split in two sets, one for training procedure, Training, and the second for quality control, QC. The Training dataset is split by ZeroCostDL4Mic into images for training the network and images for validating during the training.

Suggested folder structure of data set:

- Experiment A
    - Training dataset
        - Training_source
            - img_1.tif, img_2.tif, ...
        - Training_target
            - img_1.tif, img_2.tif, ...
    - Quality control dataset
        - Training_source
            - img_1.tif, img_2.tif
        - Training_target
            - img_1.tif, img_2.tif
    - Data to be predicted
    - Results

A detailed explanation of the notebook and project is found on the U-Net Google Colab notebook. https://colab.research.google.com/github/HenriquesLab/ZeroCostDL4Mic/blob/master/Colab_notebooks/U-Net_2D_ZeroCostDL4Mic.ipynb

11.4 ZeroCostDL4mic training results with dataset2

The dataset used for this training is described in Appendix A. To perform the training of the ZeroCostDL4Mic model, the Google Colab notebook is followed.

Screenshots and descriptions from the Training, Validation and Quality Control are presented in the following.

1) Importing dataset2

The first step in the training process is the import of the source and annotated images of dataset2. Figure 28 shows an example of source images and annotated (masked) images.

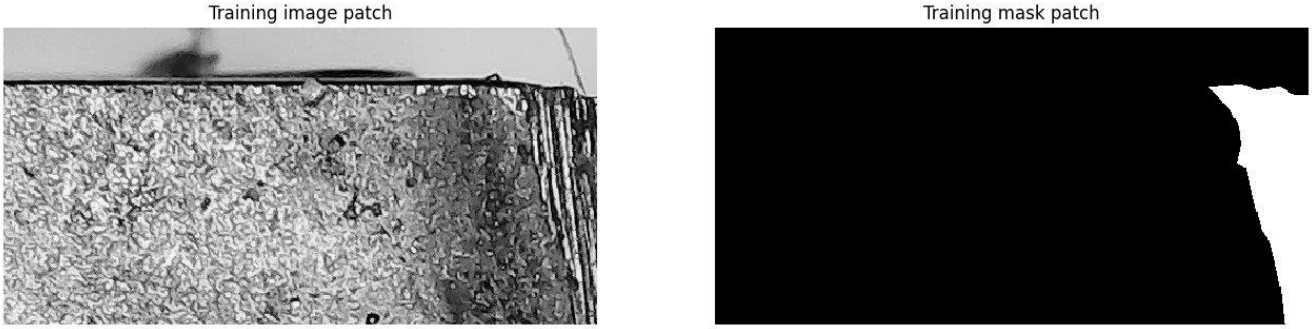


Figure 28 sample from dataset2, left: source image, right: annotated mask

2) Data Augmentation

Due to the low number of images in dataset2 it is found that data augmentation is needed to gain the best results. The default data augmentation available in ZeroCostDL4Mic is used. Figure 29 shows an example of how an images is manipulated with new angles to create more images for the training process.

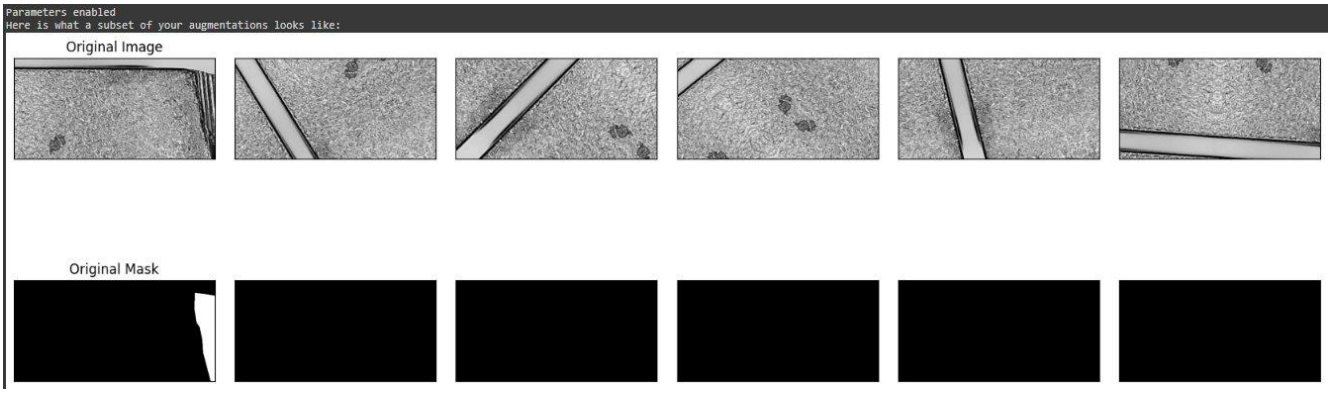


Figure 29 Data augmentation (rotation) applied on an images from dataset2

3) Training process – results

The training was conducted in 100 epochs. The graph, in figure 30, of the loss after each training epoch indicates that the trained model is training correctly, since it converges towards zero.

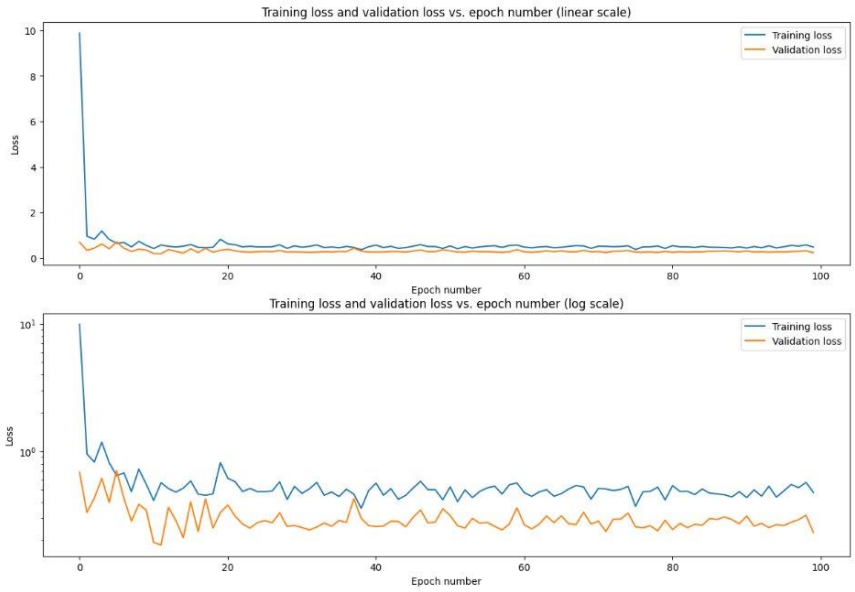


Figure 30 Linear and log representation of loss after each epoch in the training with dataset2

4) Trained model – results

Figure 31 shows one example image both as input, ground truth (annotated mask), prediction and overlay of ground truth and prediction. The examples show that the trained model does predict pixels as tool wear in the tool wear region given by the ground truth. The example also clearly shows that many pixels not being tool wear are predicted by the model to be tool wear. Therefore, the trained model needs to be drastically improved, by improving the dataset. The results also show that even though pixels are correctly predicted, there is still a need for correlating the predicted pixels with the tool wear parameter.

Figure 31 Sample from training process, from left: the input image, annotated mask, prediction from the model and rightmost overlay of annotated mask and predicted mask.

5) Quality Control – results

Images not used for the training of the model is predicted using the model, this is done to investigate the quality of the predictions. Figure 32 shows an image from the quality control images, being predicted. The thres hold is needed to get a black and white segmented image. From the predictions it seems like the model is predicting the shadow on the input image to be tool wear, which is not the case. Figure 33, shows a graph that can help choose the right threshold value, considering the Intersections over Union (IoU).

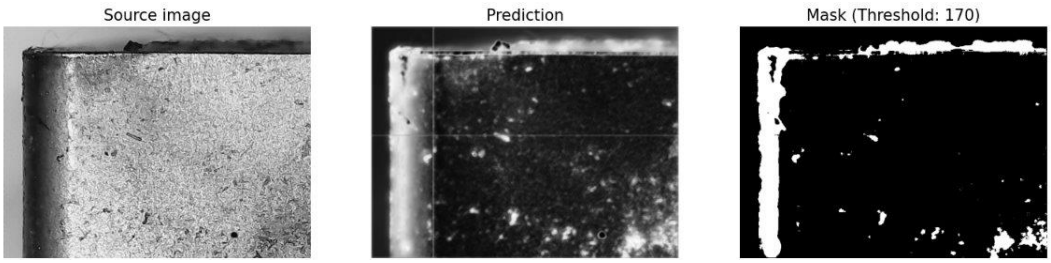


Figure 32 Quality Control sample, left: source image, middle: prediction from the model, right: prediction mask created with threshold 170

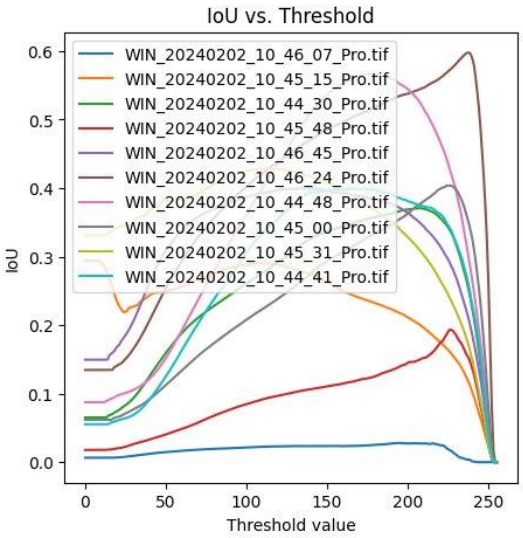


Figure 33 Graph showing IoU and Threshold relationship

6) Conclusion of training

The ZeroCostDL4Mic gives an easy way to train a model for segmentation using the U-net network. The result using the dataset2 was positive in the way that predictions could be given, but also showed the need for a better dataset. Furthermore, the predicted images, black and white pixels, do not give the final answer and additional algorithms are needed to extract and calculate the tool wear parameters. This raises the question whether segmentation is good for non-calibrated setups and even then, a segmentation model might only give the analyst a quicker way to determine the parameters but no practical knowledge for machine optimization. Knowing the tool wear area down to pixel level could still be interesting in a later stage of developing an AI4Toolwear model.

12 Appendix C – SAM – segmentation of insert image

The Segment Anything Model, SAM, is a pretrained model that is developed for creating segmented mask on any image of any object. The model has a Python API which can be prompted with points and boxes to roughly define the area of interest for creating a Mask. This prompting ability has been tested with images from dataset2, to investigate the segmentation capabilities of SAM. The following sections will show results from prompting SAM, and sample images with predicted mask, and lastly a conclusion of the potential use of SAM as part of an AI4ToolWear model.

12.1 Prompting with SAM

SAM can be prompted in different ways, most broadly an image can be input to make the model predict whatever valid segment it can find. The overall segmentation approach often creates many masks for objects and sub-objects, and is not really suited for the niche application of predicting tool wear segment. The point or box prompting can be used to overcome this challenge and to create more accurate segments.

Figure 34 shows SAM being prompted with a point, green star, defining the approximate area of the inserts flank wear. The result is three masks with different scores, of how confident SAM is in the predicted mask. It is seen that the model distinguishes fairly good between flank wear and no-wear areas of the insert, but does not include all pixels with flank wear and does include some pixels without flank wear.

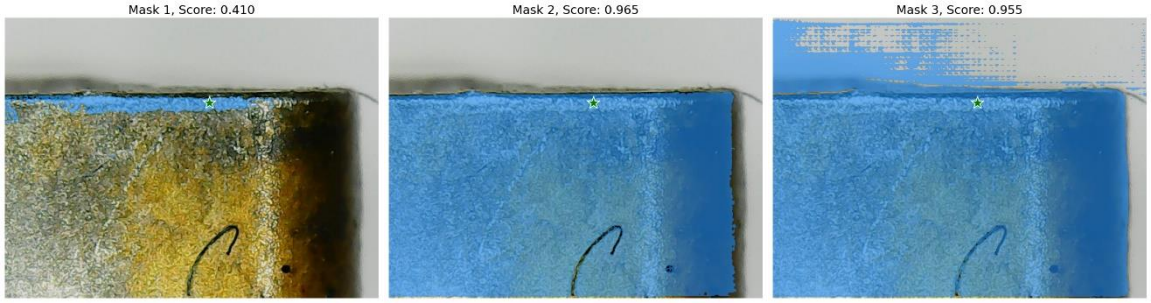


Figure 34 Prompting SAM with a point, resulting in three predicted masks.

The mask prediction can be improved by adding additional points for defining the area of interest, and furthermore points for areas not of interest can also be added. Figure 35 shows the predicted mask after prompting SAM with two points to include in the mask and two points to not include, red stars. The result is one image with a more accurate prediction of the flank wear pixels, than when using only one point for prompting. The prediction excludes better pixels that are not flank, but still leaves some flank wear pixels outside the mask.

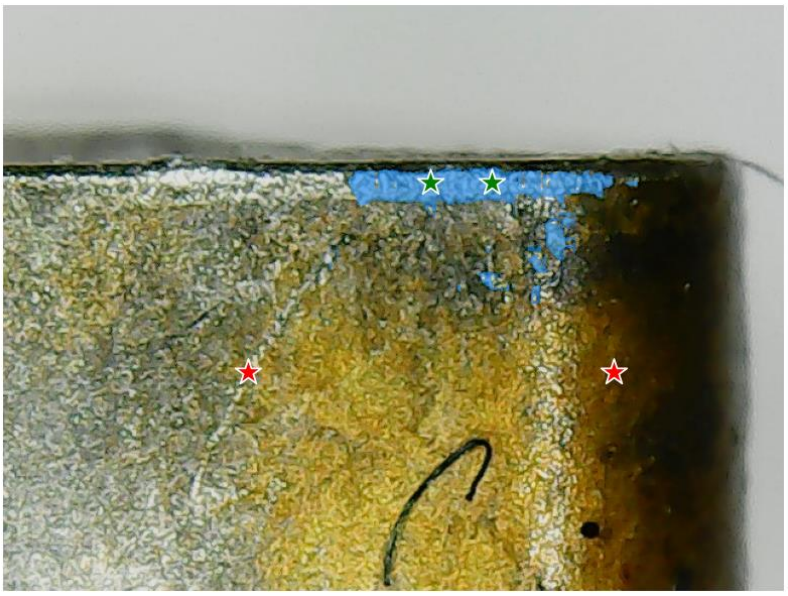


Figure 35 Prompting SAM with green stars pointing out area of interest, red start pointing out area to exclude.

Alternative to prompting with points, the model can be prompted with one ore multiple boxes of where the model should search. The points and boxes methods can also be used in combination. Figure 36 shows SAM being prompted with a box, green, which roughly defines the flank wear. The result is an image where the flank wear within the box and parts outside the box is included in predicted masks. The predicted mask from the box prompting is consider as the best prediction of the flank wear. More boxes or points could be added to create a more accurate result.

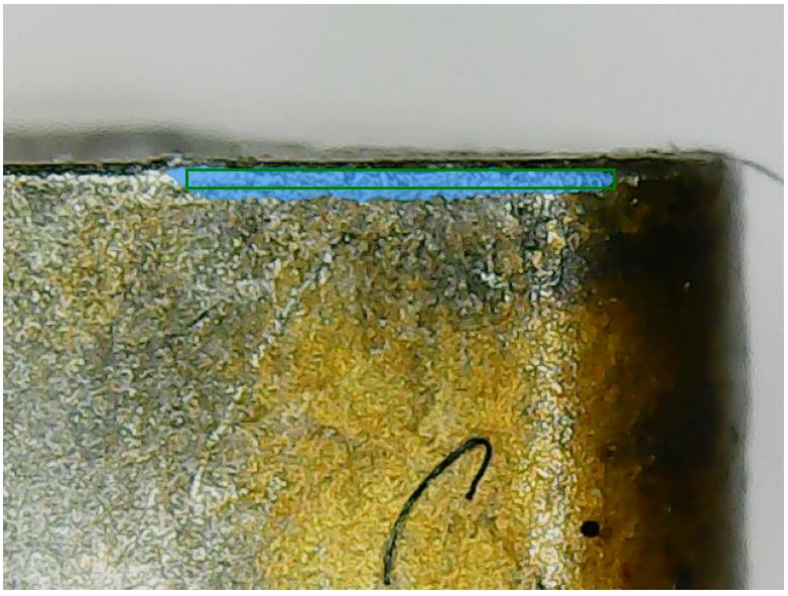


Figure 36 Prompting SAM with a box which roughly defines the flank wear area.

12.2 Conclusion of SAM used for tool wear segmentations

The prompting options of SAM shows good results for predicting the flank wear area, even though the model was not trained specifically for this, and likely have not seen a similar image during its training. The

model does not predict any class that the mask belongs to, e.g. “flank wear”, and cannot be modified to do so. SAM is seen as a very powerful tool for annotating any kind of image and could be used to improve the speed of annotation masks when creating a dataset for training a segmentation model.

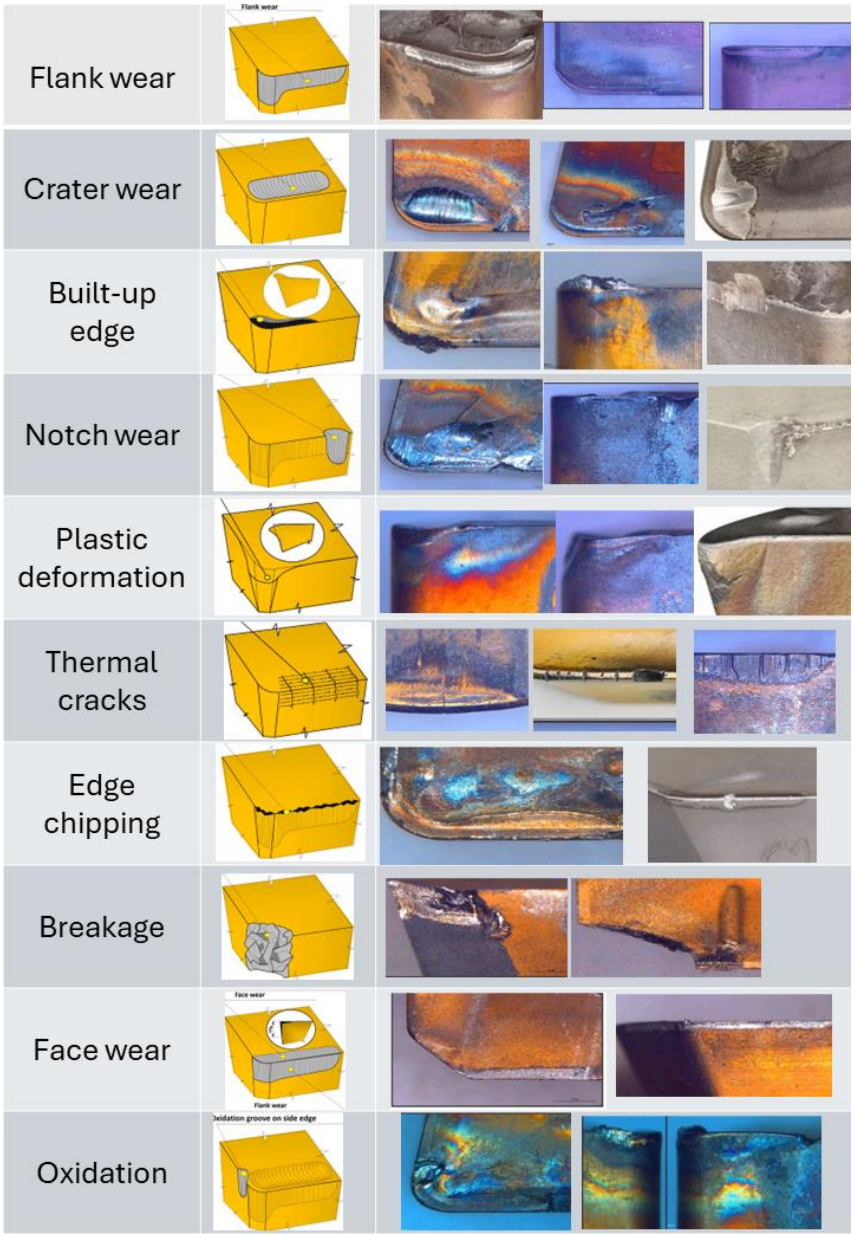
13 Appendix D – Tool wear types for dataset annotation

Below table of tool wear types have been used to annotate the "AI4toolwear" dataset. Data tool wear table is created by combining descriptions from tool manufacturers Sandvik-Coromant and Dormer Pramet, and have worked at as a reference for the annotators.

Based on:

- Sandvik-Coromant Tool wear decription: https://www.sandvik.coromant.com/en-gb/knowledge/materials/wear-on-cutting-edges (accessed 2024-05-21)
- Dormer Pramet, webinar "Insert wear": https://www.youtube.com/watch?v=xm5ISjt7ONI (accessed 2024-05-21)

13.1 Wear types



14 Appendix E – Data and datasets for training

14.1 List of data sets

Name	Description
Dataset3.0	Images of KAMF and NSM tool inserts, captured with digital microscope from Bresser, annotated by MIC.
Dataset3.1	Copy of Dataset3.0 + additional images of new angles from subset of the KAMF and NSM inserts, annotated by MIC.
Dataset4.0	Copy of images from Dataset3.1, annotated by PF
Dataset4.1	Copy of dataset4.0 + additional images captured by smartphone + Apexel 2in1 clip from subset of KAMF inserts, annotated by PF
Dataset5.0	Low occurrence tool wear types removed from annotations.

14.2 Description of data

The inserts used for creating the data sets are lent b. two machining companies, namely KAMF A/S and NSM A/S. The information given on inserts are:

- KAMF, Milling process: 43 inserts
- KAMF, Turning process: 68 inserts
- NSM, unknown process: 103 inserts
- Total: 214 inserts

Each insert has multiple used edges, which allows to capture multiple images of each inserts to create a large dataset. Number of images and distribution of annotated tool wear types are shown following subsections.

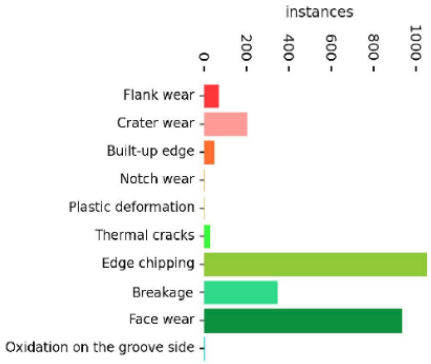
The images are captured using a digital microscope, Bresser DST-1024, and a smartphone with a lens clip, Xperia 1ii + Apexel 2in1 Macro lens.

14.2.1 Dataset3.0

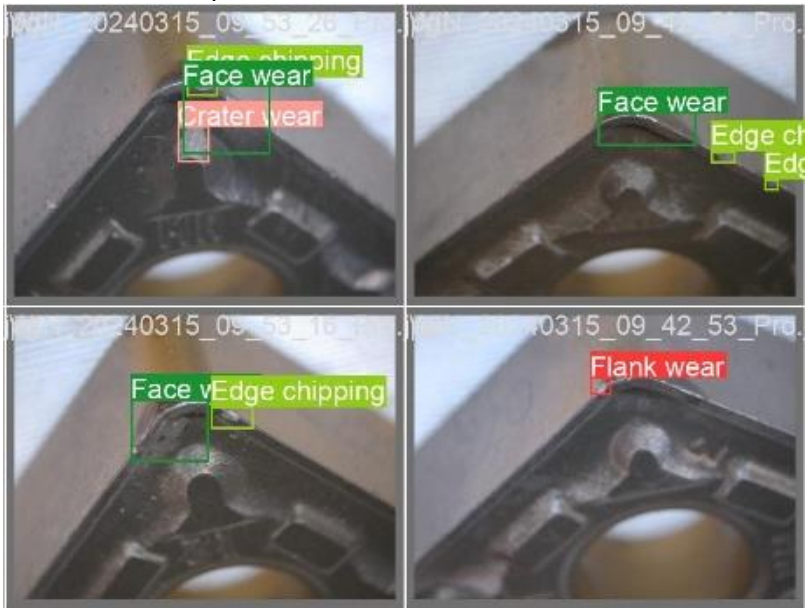
- KAMF, Milling process: 43 inserts, 192 images (train: 151, val: 41)
- KAMF, Turning process: 68 inserts, 399 images (train: 354, val: 45)
- NSM, unknown process: 103 inserts, 539 images (train: 484, val: 55)
- Total: 214 unique inserts, 1130 images (train: 88 %, val: 12 %)
- Annotator: MIC

Image samples:

Annotation resume:



Annotation sample:



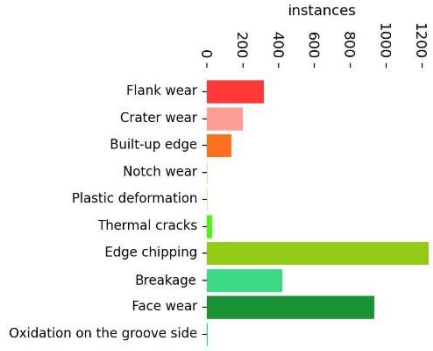
instances

14.2.2 Dataset3.1

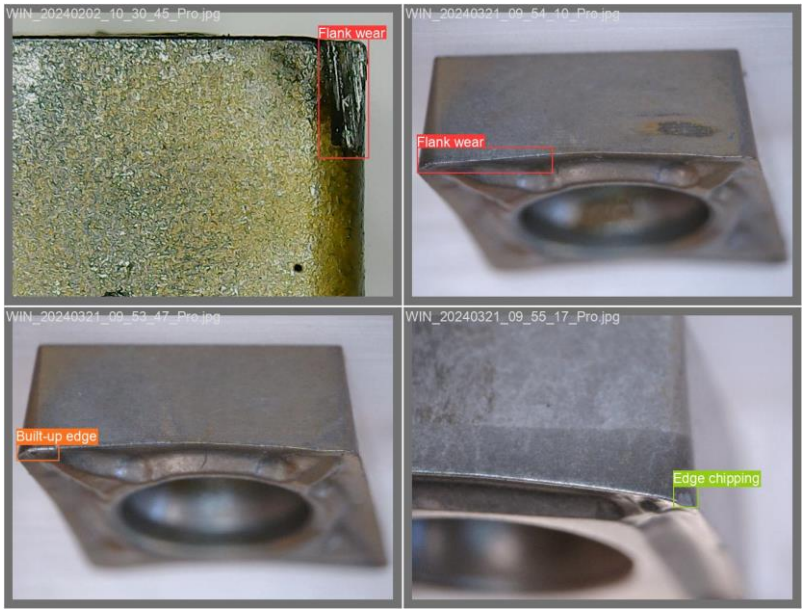
- Dataset 3.0: 214 inserts, 1130 images (train: 88 %, val: 12 %)
- KAMF turning (new angle): 68 inserts, 465 images (train 407, val: 58)
- Total: 214 unique inserts, 1595 images (train: 88 %, val: 12 %)
- Annotator: MIC

Image samples:

Annotation resume:



Annotation sample:



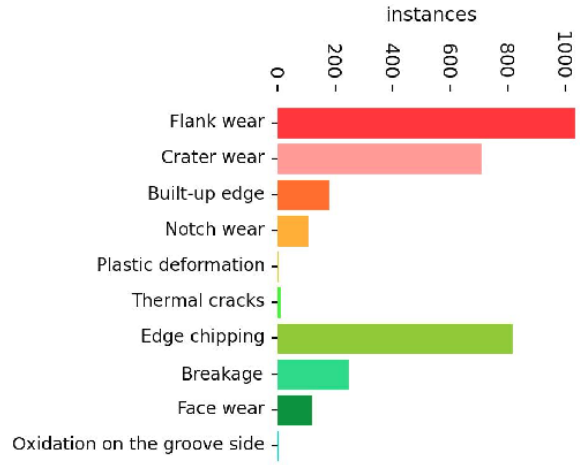
14.2.3 Dataset4

- Images from dataset3.1: 214 inserts, 1595 images (train: 88 %, val: 12 %)
- Annotator: PF

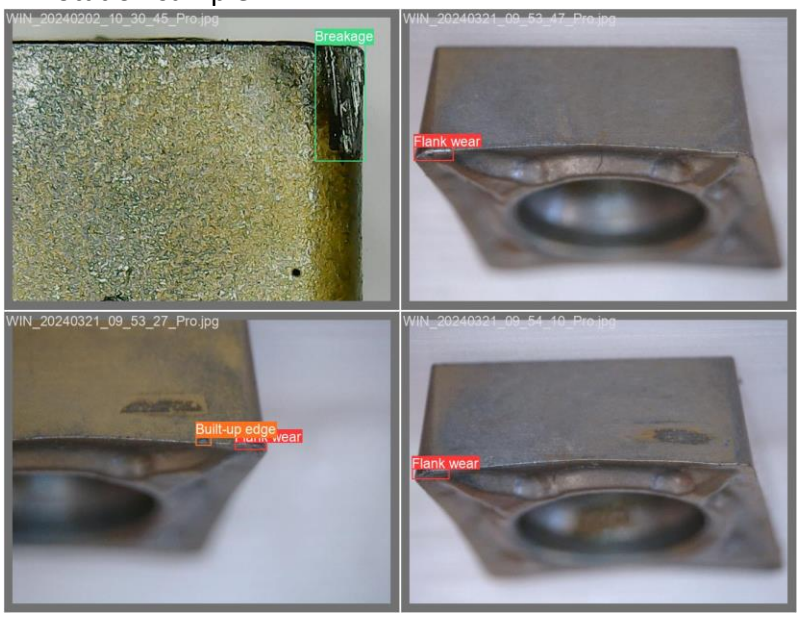
Images sample:

No new images.

Annotation resume:



Annotation sample:

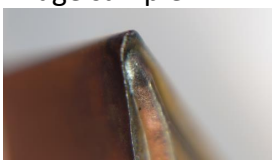


instances

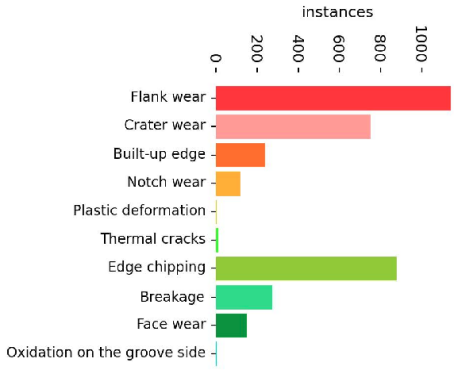
14.2.4 Dataset4.1

- Copy of dataset4.0: 214 inserts, 1595 images (train: 88 %, val: 12 %)
- Images captured using smartphone + clip: 24 inserts, 189 images
- Total: 214 unique inserts, 1784 images (train: 89 % , val: 11 %)
- Annotator: PF

Image sample:



Annotation resume:



Annotation sample:

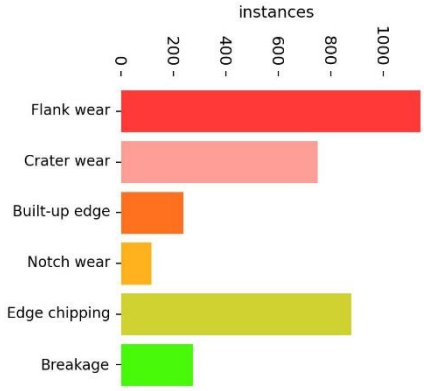
14.2.5 Dataset5.0

- Copy of dataset4.1: 214 inserts, 1595 images (train: 88 %, val: 12 %)
- Removed annotation classes:
    - Plastic deformation
    - Thermal Cracks
    - Face Wear
    - Oxidation

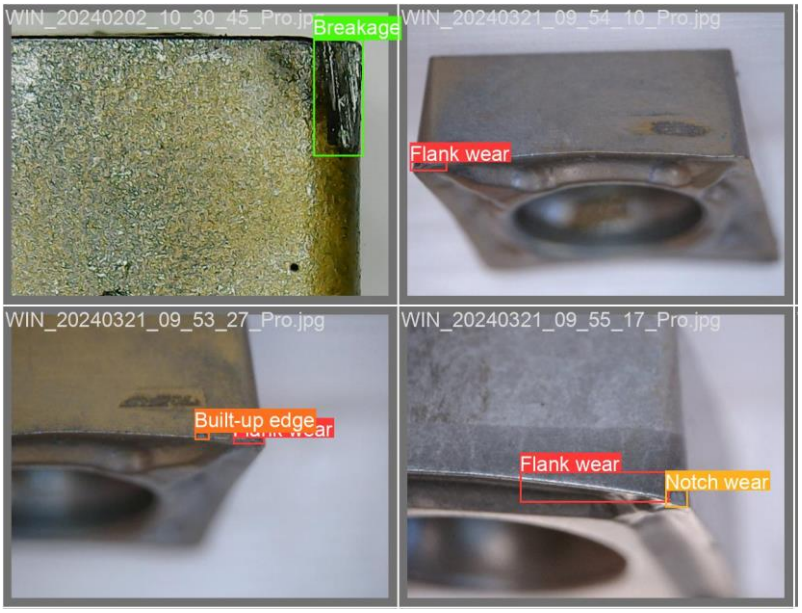
Images sample:

No new images.

Annotation resume:



Annotation sample:



15 Appendix F – Training overview

	Training name	Setup	Training hrs	Comments	Class Images Instances P R mAP50 mAP50-95)
1	train10	yolov8n.pt Dataset 3.0 Epochs: 100	1.165	mAP50-95: ~0.12 Confusion matrix not consistent in diagonal. Breakage is well predicted (0.77)	all 141 395 0.473 0.486 0.469 0.144 Flank wear 141 26 0.524 0.128 0.125 0.0377 Crater wear 141 14 0.262 0.286 0.306 0.101 Built-up edge 141 5 0.25 0.2 0.297 0.0506 Thermal cracks 141 1 0.851 1 0.995 0.0995 Edge chipping 141 195 0.342 0.451 0.332 0.111 Breakage 141 40 0.644 0.8 0.753 0.422 Face wear 141 114 0.441 0.535 0.474 0.188
2	train14	yolov8m.pt Dataset 3.0 Epochs: 100	6.229	mAP50-95: ~0.13 Confusion matrix diagonal slightly improve compared to train10. Still many False Negative	all 141 395 0.599 0.417 0.435 0.167 Flank wear 141 26 0.512 0.0412 0.0707 0.0175 Crater wear 141 14 0.471 0.256 0.242 0.0701 Built-up edge 141 5 0.527 0.2 0.256 0.1 Thermal cracks 141 1 0.844 1 0.995 0.298 Edge chipping 141 195 0.515 0.292 0.293 0.084 Breakage 141 40 0.708 0.725 0.749 0.421 Face wear 141 114 0.612 0.402 0.438 0.176
3	train16	yolov8n.pt Dataset 3.0 train10 + 50 epochs	12.599	Not showing any significant improvements in result.png, ~50 epochs seem to be enough. Overfitting might occur exceeding this number.	all 141 395 0.312 0.484 0.355 0.136 Flank wear 141 26 0.117 0.115 0.0636 0.0184 Crater wear 141 14 0.295 0.357 0.268 0.0987 Built-up edge 141 5 0.208 0.4 0.225 0.078 Thermal cracks 141 1 0.362 0.724 0.497 0.0995 Edge chipping 141 195 0.318 0.446 0.326 0.095 Breakage 141 40 0.578 0.775 0.703 0.422 Face wear 141 114 0.306 0.57 0.403 0.144

4	train17	yolov8m.pt Dataset 3.0 train14 + 100 epochs		Not showing any significant improvements in result.png, 50-100 epochs are enough. Overfitting might occur exceeding this number.	
5	train19	yolov8m.pt Dataset 3.1 Epochs: 150		mAP50-95: 0.19 More picture with flank in focus. more True Positive predictions of flank wear and improvement of confusion matrix diagonal compared to train14. But, still many false negative	all 199 480 0.679 0.429 0.469 0.19 Flank wear 199 56 0.676 0.179 0.265 0.0995 Crater wear 199 14 0.574 0.289 0.335 0.144 Built-up edge 199 19 0.621 0.259 0.261 0.0927 Thermal cracks 199 1 0.859 1 0.995 0.398 Edge chipping 199 227 0.606 0.244 0.339 0.102 Breakage 199 48 0.769 0.708 0.668 0.326 Face wear 199 115 0.645 0.322 0.421 0.17
6	train21	yolov8x.pt Dataset 3.1 Epochs: 150	34.204	mAP50-95: 0.237 Bigger model, but no huge difference from train19 in confusion matrix. Still many false negative (background detection)	all 199 480 0.435 0.517 0.486 0.237 Flank wear 199 56 0.239 0.286 0.207 0.0919 Crater wear 199 14 0.511 0.5 0.512 0.17 Built-up edge 199 19 0.375 0.158 0.202 0.108 Thermal cracks 199 1 0.302 1 0.995 0.597 Edge chipping 199 227 0.519 0.361 0.333 0.0992 Breakage 199 48 0.641 0.75 0.69 0.414 Face wear 199 115 0.459 0.565 0.459 0.183
7	train24	yolov8x.pt Dataset 4.0 Epochs: 100	22.705	mAP50-95: 0.143 New annotator (dataset4.0) compared to train21. Still many false negative (background detection). More spread result in confusion matrix. Not stable prediction with Apex 2in1 microscope.	all 197 472 0.332 0.395 0.296 0.143 Flank wear 197 175 0.326 0.371 0.26 0.0928 Crater wear 197 72 0.363 0.435 0.325 0.126 Built-up edge 197 18 0.373 0.389 0.269 0.127 Notch wear 197 23 0.487 0.522 0.478 0.245 Edge chipping 197 129 0.197 0.287 0.135 0.0507 Breakage 197 39 0.383 0.573 0.527 0.337 Face wear 197 16 0.196 0.188 0.0815 0.0214

8	train26	yolov8x.pt Dataset 4.1 Epochs: 100	25.669	mAP50-95: 0.141 Added 189 images captured with smartphone+Apexel 2in1. “manual data augmentation” Similar confusion matrix as train24, more predictions made on pictures with Apex 2in1 microscope.	all 197 472 0.472 0.288 0.31 0.141 Flank wear 197 175 0.45 0.291 0.283 0.0939 Crater wear 197 72 0.461 0.167 0.247 0.0932 Built-up edge 197 18 0.455 0.389 0.313 0.126 Notch wear 197 23 0.595 0.348 0.483 0.23 Edge chipping 197 129 0.312 0.209 0.149 0.0532 Breakage 197 39 0.648 0.487 0.545 0.369 Face wear 197 16 0.38 0.125 0.149 0.0231
9	Train38	yolov8m.pt Dataset 4.1 (Data aug.) Epochs: 50	4.964 hours	mAP50-95: 0.126 Using (yolo) data augmentation: "degree:180", "mixup:0.5"	all 197 472 0.312 0.375 0.28 0.126 Flank wear 197 175 0.376 0.314 0.232 0.0748 Crater wear 197 72 0.277 0.417 0.226 0.0749 Built-up edge 197 18 0.452 0.5 0.41 0.16 Notch wear 197 23 0.464 0.348 0.345 0.139 Edge chipping 197 129 0.28 0.302 0.159 0.0517 Breakage 197 39 0.337 0.744 0.574 0.378 Face wear 197 16 0 0 0.0171 0.00563
10	Train45	yolov8m.pt Dataset 5.0 Epochs: 50	4.952 hours	mAP50-95: 0.168 Using dataset5.0 (fewer classes). Performs class-wise and overall better than train38	all 197 456 0.401 0.395 0.347 0.168 Flank wear 197 175 0.446 0.322 0.288 0.0931 Crater wear 197 72 0.299 0.278 0.286 0.109 Built-up edge 197 18 0.381 0.389 0.259 0.126 Notch wear 197 23 0.451 0.522 0.554 0.248 Edge chipping 197 129 0.237 0.302 0.148 0.0577 Breakage 197 39 0.591 0.556 0.549 0.376
11	Train47	yolov8m.pt Dataset 5.0 (Data aug.) Epochs: 50	4.963 hours	mAP50-95: 0.157 Using (yolo) data augmentation: "degree:180", "mixup:0.5". Lower performance than train45.	all 197 456 0.331 0.429 0.328 0.157 Flank wear 197 175 0.292 0.28 0.206 0.066 Crater wear 197 72 0.362 0.444 0.323 0.117 Built-up edge 197 18 0.34 0.556 0.404 0.193 Notch wear 197 23 0.48 0.361 0.34 0.142 Edge chipping 197 129 0.225 0.14 0.145 0.051

					Breakage 197 39 0.288 0.795 0.55 0.373
12	Train50	yolov8m.pt Dataset 5.0 (Data aug.) train45(best.pt) + 100 epochs	9.924 hours.	mAP50-95: 0.157 Continuing training of train45, Using (yolo) data augmentation: "degree:180", "mixup:0.5". Lower performance than train45. Lower overall performance than train47, but few class-wise improvements, e.g. flankwear.	all 197 456 0.403 0.394 0.323 0.152 Flank wear 197 175 0.371 0.343 0.255 0.0823 Crater wear 197 72 0.332 0.297 0.246 0.0906 Built-up edge 197 18 0.495 0.444 0.375 0.166 Notch wear 197 23 0.469 0.391 0.385 0.16 Edge chipping 197 129 0.271 0.271 0.15 0.0503 Breakage 197 39 0.48 0.615 0.528 0.362

16 Appendix G - Interference data on train50 model

16.1 Data description

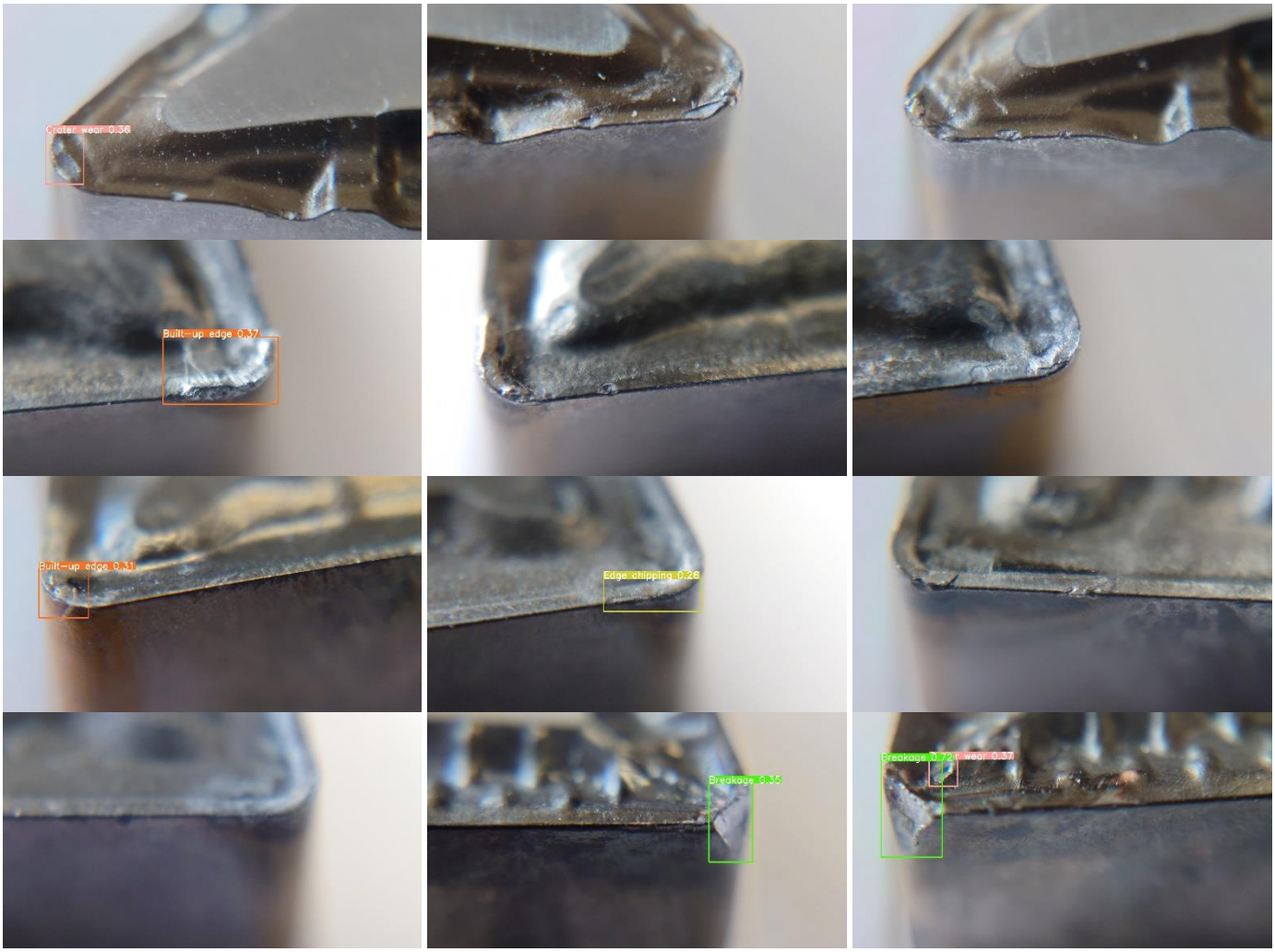
Interference data description	
Digital Microscope	Smartphone + microscope clip
Type: Bresser DST1024	Setup: Sony Xperia 1ii, 70 mm lens + Apexel 2in1 Macro clip
18 images (all with visual wear)	22 images (all with visual wear)

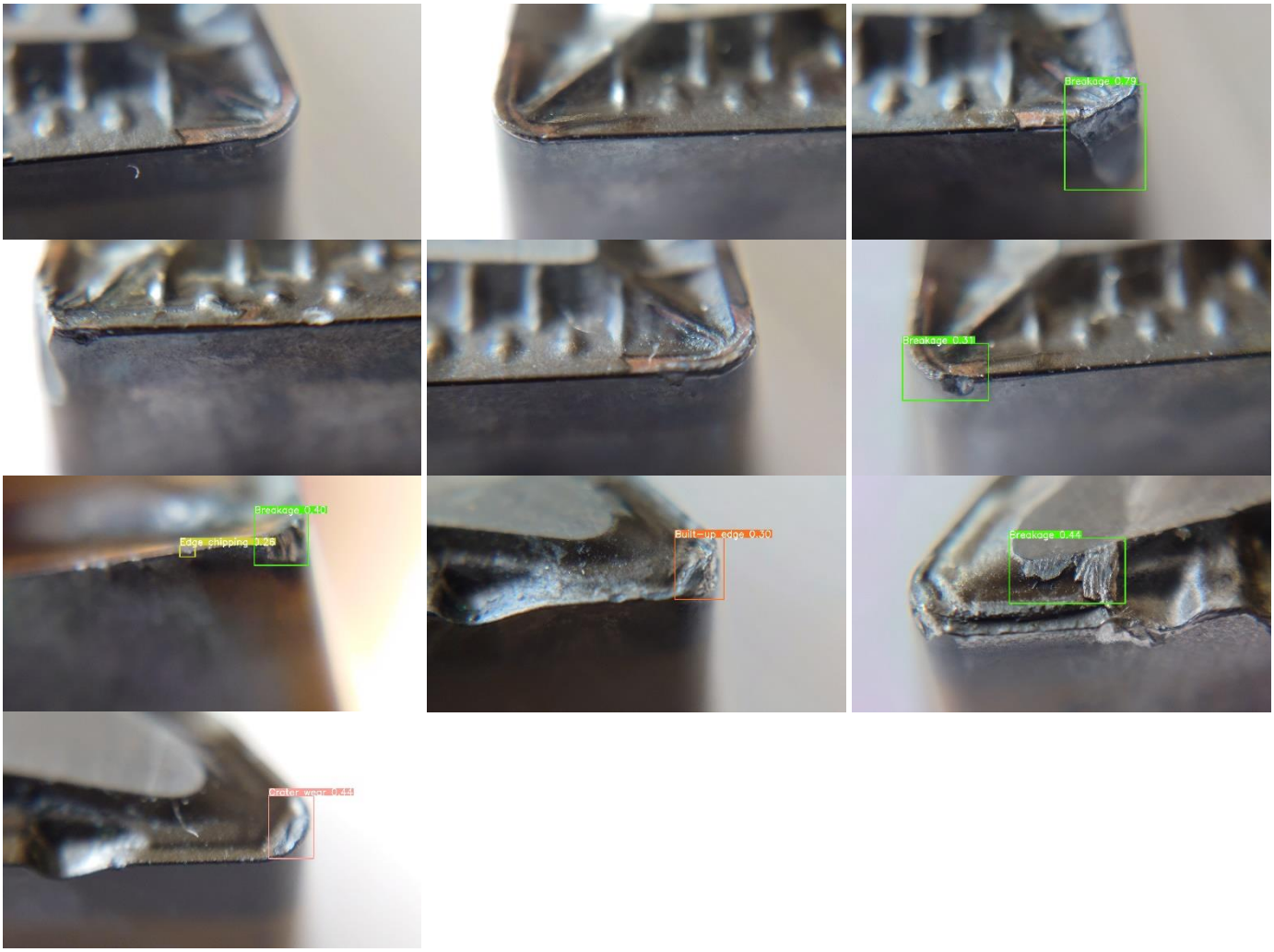
16.2 Overview of predictions

Digital Microscope	Smartphone + microscope clip
0: 480x640 1 Flank wear, 2 Built-up edges, 1: 480x640 1 Crater wear, 1 Notch wear, 1 Breakage, 2: 480x640 1 Crater wear, 3: 480x640 (no detections), 4: 480x640 1 Flank wear, 5: 480x640 2 Flank wears, 1 Built-up edge, 6: 480x640 1 Crater wear, 7: 480x640 1 Flank wear, 1 Crater wear, 1 Built-up edge, 1 Edge chipping, 8: 480x640 1 Crater wear, 1 Breakage, 9: 480x640 (no detections), 10: 480x640 1 Edge chipping, 11: 480x640 1 Flank wear, 12: 480x640 3 Edge chippings, 13: 480x640 1 Flank wear, 1 Crater wear, 1 Edge chipping, 14: 480x640 (no detections), 15: 480x640 1 Crater wear, 16: 480x640 1 Flank wear, 17: 480x640 1 Built-up edge,	0: 384x640 (no detections), 1: 384x640 1 Edge chipping, 1 Breakage, 2: 384x640 1 Built-up edge, 3: 384x640 1 Crater wear, 1 Breakage, 4: 384x640 1 Breakage, 5: 384x640 (no detections), 6: 384x640 1 Breakage, 7: 384x640 1 Breakage, 8: 384x640 1 Edge chipping, 9: 384x640 1 Breakage, 10: 384x640 (no detections), 11: 384x640 (no detections), 12: 384x640 1 Crater wear, 13: 384x640 (no detections), 14: 384x640 (no detections), 15: 384x640 (no detections), 16: 384x640 1 Crater wear, 17: 384x640 1 Built-up edge, 18: 384x640 (no detections), 19: 384x640 (no detections), 20: 384x640 1 Built-up edge, 21: 384x640 (no detections),
Images with detection: 15 (83 %)	Images with detection: 12 (55 %)
Images with >1 detection: 6 (33 %)	Images with >1 detection: (9 %)

16.3 Predictions Digital microscope

16.4 Predictions Smartphone + microscope clip




Breakage 0.79
Breakage 0.31
Breakage 0.40
Edge chipping 0.26
Built-up edge 0.30
Breakage 0.44
Crater wear 0.44

17 References

[1] Daicu, R., & Oancea, G. (2022). Methodology for measuring the cutting inserts wear. Symmetry, 14(3), 469. https://doi.org/10.3390/sym14030469

[2] Lin, W.-J., Chen, J.-W., Jhuang, J.-P., Tsai, M.-S., Hung, C.-L., Li, K.-M., & Young, H.-T. (2021). Integrating object detection and image segmentation for detecting the tool wear area on Stitched Image. Scientific Reports, 11(1). https://doi.org/10.1038/s41598-021-97610-y

[3] 230808_Compendium_MERGED_v3.pdf course compendium by DAMRC 2023

[4] Zimmerman, A. (2023). A ghostwriter for the masses: Chatgpt and the future of writing. Annals of Surgical Oncology, 30(6), 3170–3173. https://doi.org/10.1245/s10434-023-13436-0

[5] Smartphone app by Sandvik-Coromant https://www.sandvik.coromant.com/en-us/knowledge/machining-calculators-apps/tool-wear-analyzer

[6] https://en.wikipedia.org/wiki/Artificial_intelligence (accessed 240129)

[7] DSlab Global, https://dslab-global.medium.com/artificial-intelligence-430553aa775b (accessed 240129)

[8] https://www.image-net.org/ (accessed 240129)

[9] Baeldung, https://www.baeldung.com/cs/ml-nonlinear-activation-functions (accessed 240314)

[10]Bergs, T., Holst, C., Gupta, P., & Augspurger, T. (2020). Digital Image Processing with deep learning for automated cutting tool wear detection. Procedia Manufacturing, 48, 947–958. https://doi.org/10.1016/j.promfg.2020.05.134

[11] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Lecture Notes in Computer Science, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

[12] Kröse, B., & Smagt, P. van der. (1996). An introduction to neural networks (8th ed.). University of Amsterdam.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016. doi:10.1109/cvpr.2016.91

[13] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment anything. 2023 IEEE/CVF International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv51070.2023.00371

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016. doi:10.1109/cvpr.2016.91

[15] https://docs.ultralytics.com/quickstart/#conda-docker-image (accessed 240507)

[16] G. Martínez-Arellano, G. Terrazas, and S. Ratchev, “Tool wear classification using time series imaging and Deep Learning,” The International Journal of Advanced Manufacturing Technology, vol. 104, no. 9–12, pp. 3647–3662, Jul. 2019. doi:10.1007/s00170-019-04090-6

[17] M. T. García-Ordás, E. Alegre, V. González-Castro, and R. Alaiz-Rodríguez, “A computer vision approach to analyze and classify tool wear level in milling processes using shape descriptors and machine learning techniques,” The International Journal of Advanced Manufacturing Technology, vol. 90, no. 5–8, pp. 1947–1961, Oct. 2016. doi:10.1007/s00170-016-9541-0

[18] T. Mikołajczyk, K. Nowicki, A. Kłodowski, and D. Yu. Pimenov, “Neural network approach for automatic image analysis of Cutting Edge Wear,” Mechanical Systems and Signal Processing, vol. 88, pp. 100–110, May 2017. doi:10.1016/j.ymssp.2016.11.026

[19] T. Bergs, C. Holst, P. Gupta, and T. Augspurger, “Digital Image Processing with deep learning for automated cutting tool wear detection,” Procedia Manufacturing, vol. 48, pp. 947–958, 2020. doi:10.1016/j.promfg.2020.05.134

[20] W. Lin-Ju et al., “Integrating object detection and image segmentation for detecting the tool wear area on Stitched Image,” Scientific Reports, vol. 11, no. 1, Oct. 2021. doi:10.1038/s41598-021-97610-y

[21] Jocher, G. et al. https://docs.ultralytics.com/models/yolov8/#performance-metrics (accessed 240524)

[22] Hidalgo-Cenalmor, I., Pylvänäinen, J. W., Ferreira, M. G., Russell, C. T., Arganda-Carreras, I., Jacquemet, G., Henriques, R., & Gómez-de-Mariscal, E. (2023). DL4MicEverywhere: Deep Learning for Microscopy Made Flexible, Shareable, and Reproducible. https://doi.org/10.1101/2023.11.19.567606

[23] Jocher, G. et al. https://docs.ultralytics.com/modes/train/#train-settings (accessed 240606)

[24] Jocher, G. et al. https://docs.ultralytics.com/guides/yolo-performance-metrics/ (accessed 240606)