Szkoła Główna Gospodarstwa Wiejskiego
w Warszawie

Instytut Informatyki Technicznej

mgr Michał Bukowski

Zastosowanie zaawansowanych metod sztucznej inteligencji do klasyfikacji stanu zużycia narzędzia na podstawie danych obrazowych i sygnałowych pozyskanych podczas obróbki płyty wiórowej

Application of advanced artificial intelligence methods for classifying the wear condition of a tool based on image and signal data acquired during chipboard machining

Rozprawa doktorska

Doctoral thesis

Rozprawa doktorska wykonana pod kierunkiem
dra hab. inż. Jarosława Kurka, prof. SGGW
Instytut Informatyki Technicznej
Szkoła Główna Gospodarstwa Wiejskiego w Warszawie

Warszawa 2025

Oświadczenie promotora rozprawy doktorskiej

Oświadczam, że niniejsza rozprawa została przygotowana pod moim kierunkiem, i stwierdzam, że spełnia warunki do przedstawienia jej w postępowaniu o nadanie stopnia naukowego doktora.

Data 27-01-2025 Podpis promotora [illegible]

Oświadczenie autora rozprawy doktorskiej

Świadom odpowiedzialności prawnej, w tym odpowiedzialności karnej za złożenie fałszywego oświadczenia, oświadczam, że niniejsza rozprawa doktorska została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami prawa, w szczególności z ustawą z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. z dnia 28 października 2022 r., Dz.U. z 2022 r. poz. 2509 ze zm.).

Oświadczam, że przedstawiona rozprawa nie była wcześniej podstawą żadnej procedury związanej z uzyskaniem stopnia naukowego doktora.

Oświadczam ponadto, że niniejsza wersja rozprawy jest identyczna z załączoną wersją elektroniczną.

Przyjmuję do wiadomości, że rozprawa doktorska poddana zostanie procedurze antyplagiatowej.

Data 27-01 2025 Podpis autora pracy [illegible]

Spis treści

Streszczenie rozprawy doktorskiej

Niniejsza rozprawa doktorska poświęcona jest rozwojowi i zastosowaniu zaawansowanych metod sztucznej inteligencji do klasyfikacji stanu zużycia narzędzi, w szczególności wierteł podczas wiercenia oraz ostrza tnącego w procesie frezowania, na podstawie danych obrazowych i sygnałowych uzyskanych podczas obróbki materiałów. Głównym celem pracy jest opracowanie skutecznych modeli i algorytmów sztucznej inteligencji umożliwiających dokładną i wiarygodną ocenę stanu zużycia narzędzi, co ma kluczowe znaczenie dla efektywności i ekonomii produkcji przemysłowej.

W pracy podkreślono znaczenie monitorowania zużycia narzędzi w kontekście Przemysłu 4.0, gdzie integracja czujników i analiza danych w czasie rzeczywistym wspierają strategię predykcyjnej konserwacji, zwiększając żywotność narzędzi i efektywność linii produkcyjnych. Efekty te mają pozytywny wpływ na środowisko, redukując liczbę odpadów, zużycie energii i koszty produkcji.

Badania przeprowadzone w ramach rozprawy doktorskiej wykazały skuteczność technik sztucznej inteligencji, takich jak głębokie sieci konwolucyjne, sieci syjamskie, transformery wizyjne oraz algorytm XGBoost, w zadaniu klasyfikacji stanu zużycia narzędzi. W pracy przedstawiono również innowacyjne podejścia, takie jak wprowadzenie głębokiej architektury w modelach sieci kapsułowych wykorzystujących mechanizm uwagi, wykorzystanie niestandardowych funkcji strat oraz integracja oceny pewności decyzji z modelami sztucznej inteligencji, co poprawia dokładność i interpretowalność klasyfikacji.

Praca doktorska wnosi istotny wkład w rozwój dyscypliny informatyka techniczna i telekomunikacja, a w szczególności obszaru sztucznej inteligencji i jej zastosowań w przemyśle, oferując nowe perspektywy i rozwiązania w obszarze bezinwazyjnego monitorowania stanu zużycia narzędzi w procesach wiercenia oraz frezowania, co ma bezpośrednie implikacje dla przemysłu produkcyjnego.

W skład rozprawy wchodzi 6 artykułów naukowych, których kopie zostały zamieszczone w rozdziale 10, a ich zbiorcze opracowanie w rozdziałach 7-8.

Dissertation Abstract

This doctoral dissertation focuses on the development and application of advanced artificial intelligence methods for the classification of tool wear, specifically drill during drilling and cutting edges during milling, based on image and signal data obtained during material processing. The central aim of the work is to develop effective artificial intelligence models and algorithms that enable accurate and reliable assessment of tool wear, which is critical for the efficiency and economics of industrial production.

The dissertation highlights the importance of tool wear monitoring in the context of Industry 4.0, where the integration of sensors and real-time data analysis supports predictive maintenance strategies, increasing tool lifespan and production line efficiency. These effects positively impact the environment by reducing waste, energy consumption, and production costs.

The research conducted as part of this dissertation demonstrated the effectiveness of artificial intelligence techniques, such as deep convolutional networks, Siamese networks, vision transformers, and the XGBoost algorithm, in the task of classifying tool wear. The dissertation also introduces innovative approaches, including the implementation of deep architectures in capsule networks utilizing attention mechanisms, the use of custom loss functions, and the integration of decision confidence assessment into AI models, improving both classification accuracy and interpretability.

This doctoral dissertation makes a significant contribution to the development of artificial intelligence and its applications in industry, offering new perspectives and solutions in the field of non-invasive tool wear monitoring during drilling and milling processes. These advancements have direct implications for industrial manufacturing.

This doctoral dissertation makes a significant contribution to the development of the discipline of information and communication technology, particularly in the field of artificial intelligence and its applications in industry. It offers new perspectives and solutions in the area of non-invasive tool wear monitoring during drilling and milling processes, with direct implications for industrial manufacturing.

The dissertation comprises six scientific papers, copies of which are included in chapter 10, with their collective analysis presented in chapters 7-8.

1. Wprowadzenie

Rozwój technologii i dążenie do optymalizacji procesów produkcyjnych wymuszają na przemyśle wprowadzanie ciągłych innowacji i nowych rozwiązań. Jednym z kluczowych elementów oddziałujących na efektywność produkcji jest stan narzędzi, których zużycie wpływa bezpośrednio na jakość wytwarzanych produktów i ekonomię procesów. W tym kontekście monitorowanie i diagnostyka stanu zużycia narzędzi są postrzegane jako nieodzowny element nowoczesnych systemów produkcyjnych, wpisujący się w paradygmat Przemysłu 4.0, w którym główną rolę odgrywa gromadzenie i analiza danych.

W niniejszej pracy doktorskiej skupiono się na zastosowaniu zaawansowanych metod sztucznej inteligencji do klasyfikacji stanu zużycia narzędzi w procesach wiercenia oraz frezowania. Wydajne modele predykcyjne umożliwiające precyzyjną i wiarygodną ocenę stanu narzędzi zostały opracowane przy wykorzystaniu zaawansowanych architektur, takich jak głębokie sieci neuronowe, transformery wizyjne oraz inne algorytmy uczenia maszynowego. Dzięki zastosowaniu nowoczesnych technik sztucznej inteligencji możliwe są zwiększenie jakości produkcji, redukcja przestojów oraz optymalizacja kosztów związanych z wymianą i konserwacją narzędzi.

Głównym celem pracy były rozwój i walidacja modeli sztucznej inteligencji, które, bazując na danych obrazowych i sygnałowych pochodzących z procesów wiercenia i frezowania, zostały wykorzystane do efektywnej klasyfikacji stanu zużycia narzędzi. W pracy sformułowano hipotezy badawcze, koncentrujące się na potencjale różnych architektur sieci neuronowych i algorytmów uczenia maszynowego w kontekście zadania klasyfikacji zużycia narzędzi. Poprzez wnikliwą analizę i eksperymenty podjęto działania zmierzające do osiągnięcia zarówno teoretycznego wkładu w rozwój dyscypliny informatyka techniczna i telekomunikacja, w szczególności w obszarze sztucznej inteligencji, jak i praktycznych implikacji dla przemysłu, dzięki umożliwieniu bardziej zrównoważonego i efektywnego wykorzystania narzędzi produkcyjnych.

2. Przegląd aktualnego stanu wiedzy

2.1. Znaczenie monitorowania narzędzi

Monitorowanie stanu zużycia narzędzi zostało uznane za kluczowy element w przemyśle produkcyjnym, mający bezpośredni wpływ na efektywność, jakość i opłacalność procesów produkcyjnych [1, 12, 16, 18–20]. Narzędzia stosowane w operacjach obróbki, takie jak wiertła, ostrza tnące frezarki, podlegają zużyciu wskutek mechanicznych i cieplnych obciążeń napotykanych podczas pracy. Możliwość dokładnej oceny stanu zużycia tych narzędzi została uznana za kluczową dla utrzymania optymalnych warunków cięcia, zapewnienia produkcji części wysokiej jakości oraz minimalizacji przestojów. Zaawansowane systemy monitorowania zużycia narzędzi pozwalają producentom na wymianę lub remont narzędzi w najbardziej odpowiednim momencie i uniknięcie tym samym przedwczesnej wymiany, która prowadzi do niepotrzebnych kosztów, lub nadmiernego zużycia, które mogłoby zagrażać jakości produktu i potencjalnie uszkodzić maszyny.

Inteligentna produkcja i technologie Przemysłu 4.0 spowodowały podniesienie znaczenia monitorowania zużycia narzędzi. Integracja czujników i analizy danych w czasie rzeczywistym z systemami produkcyjnymi umożliwia ciągłą ocenę stanu narzędzi, ułatwiając realizację strategii predykcyjnej konserwacji. Wpływa to na zwiększenie żywotności narzędzi poprzez zapobieganie ich nadmiernemu wykorzystaniu oraz wspiera automatyzację procesów produkcyjnych. W konsekwencji poprawia się efektywność linii produkcyjnych, a także zdolność do przestrzegania ścisłych tolerancji i specyfikacji, które często są kluczowe w takich branżach jak lotnicza, motoryzacyjna czy produkcja urządzeń medycznych.

Nie mogą zostać też pominięte takie aspekty utrzymania narzędzi jak wpływ na środowisko i zrównoważony rozwój. Efektywne wykorzystanie narzędzi i terminowa konserwacja redukują odpady i zużycie energii związane z procesem produkcyjnym. Optymalizując żywotność narzędzi i redukując częstotliwość ich wymiany, firmy mogą zmniejszyć swój ślad węglowy i wspierać stosowanie zrównoważonych praktyk produkcyjnych. Ponadto dane zebrane dzięki monitorowaniu zużycia narzędzi mogą być wykorzystane do udoskonalania i poprawy procesów produkcyjnych, prowadząc do dalszego wzrostu efektywności i redukcji odpadów materiałowych.

2.2. Sztuczna inteligencja jako skuteczne narzędzie w monitorowaniu zużycia narzędzi

Integracja metod sztucznej inteligencji w monitorowaniu i optymalizacji sprzętu przemysłowego stanowi znaczący postęp w strategiach produkcji i utrzymania [11, 17]. Badania prowadzone od końca lat 90. [8] wykazały skuteczność technik sztucznej inteligencji, takich jak LSTM, Bi-LSTM, ANNs i klasyfikatory SVM, w przewidywaniu awarii sprzętu w różnych branżach. Adaptacja tych technologii ułatwia głębsze zrozumienie wydajności sprzętu, umożliwiając utrzymanie predykcyjne i redukując czas przestoju.

Zastosowanie metod sztucznej inteligencji rozciąga się na optymalizację sprzętu obrotowego w przemyśle [9] oraz rozwój systemów monitorowania zdalnego, które włączają technologie internetu rzeczy (IoT) [14]. Porównując różne metody sztucznej inteligencji i ich skuteczność w odniesieniu do danych przemysłowych, badacze mają na celu dobór odpowiednich technik do konkretnych scenariuszy i tworzenie tym samym specjalnych rozwiązań konkretnych problemów przemysłowych.

Pojawienie się Przemysłu 4.0 jeszcze bardziej podkreśliło rolę technik sztucznej inteligencji w zwiększaniu ogólnej efektywności wyposażenia [2]. Biorąc pod uwagę konkretnie przypadki użycia oraz projekty badawcze, transformacyjny potencjał sztucznej inteligencji w kontroli procesów przemysłowych i optymalizacji jest oczywisty, co oznacza nową erę zarządzania produkcją i procesami.

3. Spis publikacji wchodzących w skład rozprawy doktorskiej

W tabeli 3.1 przedstawiono zestawienie publikacji, które wchodzą w skład niniejszej rozprawy doktorskiej. Spis ten obejmuje publikacje uporządkowane chronologicznie, co odzwierciedla postępy badawcze oraz wkład autora w rozwój dyscypliny informatyka techniczna i telekomunikacja. Każda pozycja w tabeli zawiera istotne informacje, takie jak rok publikacji, liczbę przyznanych punktów MNiSW/MEiN, *impact factor* (IF) oraz pełne dane dotyczące publikacji wraz z odnośnikiem do pełnego tekstu na podstawie DOI.

Sumarycznie przedstawione publikacje świadczą o intensywnym i jednotematycznym wkładzie autora w rozwój metod sztucznej inteligencji w zastosowaniu do klasyfikacji stanu zużycia narzędzi. Ich znaczący wpływ oraz wysoki poziom merytoryczny został potwierdzony przez przyznanie odpowiedniej liczby punktów oraz wysokie wartości *impact factor*, co świadczy o uznaniu w środowisku naukowym oraz potencjale aplikacyjnym przedstawionych badań.

Tabela 3.1. Chronologiczny spis publikacji wchodzących w skład rozprawy doktorskiej

#	Rok	Punkty	IF	Tytuł publikacji
1	2020	100	3,576	Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M., Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978, `https://doi.org/10.3390/s20236978`
2	2021	100	3,847	Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A., Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834, `https://doi.org/10.3390/s21113834`
3	2023	100	1,200	Bukowski, M.; Antoniuk, I; Kurek, J., Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification. Bulletin of the Polish Academy of Sciences, Technical Sciences 2023, 71, 1–10, `https://doi.org/10.24425/bpasts.2023.147338`
4	2024	70	0,7	Bukowski, M.; Antoniuk, I.; Szymanowski, K.; Krupa, A.; Kurek, J., Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024, 34(3), 41-60, https://doi.org/10.37190/ord240303
5	2024	70	0	Bukowski, M.; Jegorowa, A.; Kurek, J., A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard. Przeglad Elektrotechniczny 2024, 5, https://doi.org/10.15199/48.2024.05.52
6	2024	100	3,900	Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A., Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092, https://doi.org/10.3390/s24041092
Suma		**540**	**13,223**	

4. Cel pracy oraz hipotezy badawcze

4.1. Cel pracy

Celem głównym pracy był rozwój zaawansowanych metod sztucznej inteligencji w celu klasyfikacji stanu zużycia narzędzi (wiertła i ostrza tnącego frezarki), na podstawie danych obrazowych uzyskanych podczas wiercenia w płycie wiórowej oraz danych sygnałowych pozyskanych podczas frezowania tejże płyty. Celem autora pracy było opracowanie skutecznych modeli i algorytmów, które pozwolą na dokładną i wiarygodną ocenę stanu zużycia narzędzi wykorzystywanych w procesach produkcyjnych, co ma kluczowe znaczenie dla efektywności i ekonomii produkcji przemysłowej.

4.2. Hipotezy badawcze

Wnikliwa analiza stanu wiedzy pozwoliła na postawienie 6 głównych hipotez badawczych:

H1: Zastosowanie sieci syjamskich oraz transformerów wizyjnych (ang. *vision transformers* – ViT) w klasyfikacji zużycia narzędzi może pozwolić na osiągnięcie wysokiej dokładności klasyfikacji, przewyższając tradycyjne metody, przy jednoczesnym zminimalizowaniu błędów klasyfikacji pomiędzy krytycznymi klasami.

H2: Integracja mechanizmów oceny pewności decyzji z modelami sztucznej inteligencji może zwiększyć zarówno dokładność, jak i interpretowalność klasyfikacji zużycia narzędzi, nawet w przypadku ograniczonej dostępności danych, bez konieczności modyfikowania procesu uczenia ani architektury modelu.

H3: Zastosowanie mechanizmu uwagi w sieciach kapsułowych może pozwolić na pogłębienie ich architektury, przy jednoczesnym zachowaniu możliwości treningu na standardowych stacjach roboczych.

H4: Architektura CNN z wieloma wejściami może efektywniej klasyfikować stan zużycia narzędzi w procesie frezowania poprzez lepsze wykorzystanie złożonych sygnałów czasowych.

H5: Wykorzystanie obrazów skalogramów jako danych wejściowych w sieciach CNN może znacząco zwiększyć dokładność rozpoznawania stanu narzędzi poprzez lepsze odwzorowanie charakterystyk sygnałów czasowo-częstotliwościowych.

H6: Zastosowanie niestandardowych funkcji strat w algorytmie XGBoost może skutecznie rozwiązać problem niezbalansowania klas oraz zwiększyć dokładność klasyfikacji krytycznych stanów zużycia narzędzi, co jest kluczowe dla zapewnienia ciągłości i wysokiej jakości produkcji w przemyśle meblarskim.

5. Indywidualny wkład współautorów w powstanie publikacji wchodzących w skład rozprawy doktorskiej

W rozdziale tym zawarto zbiorcze zestawienie indywidualnego udziału każdego ze współautorów w publikacjach, które wchodzą w skład niniejszej rozprawy.

5.1. Publikacja 1: Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes

Tabela 5.1. Procentowy udział współautorów w publikacji „Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes”

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Jarosław Kurek	20%	20	0,7152	• koncepcja • metodologia • nadzór • administracja • walidacja	[illegible]
2	Izabella Antoniuk	10%	10	0,3576	• koncepcja (współpraca) • metodologia (współpraca) • pierwszy szkic • edycja i korekta	[illegible]
3	Bartosz Świderski	10%	10	0,3576	• koncepcja (współpraca) • metodologia (współpraca) • wizualizacja	[illegible]
4	Albina Jegorowa	10%	10	0,3576	• koncepcja (współpraca) • metodologia (współpraca) • akwizycja danych • zarządzanie danymi	[illegible]
5	Michał Bukowski	50%	50	1,788	• koncepcja (współpraca) • metodologia (współpraca) • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja	[illegible]
	Suma	**100%**	**100**	**3,576**	---	---

5.2. Publikacja 2: Decision Confidence Assessment in Multi-Class Classification

Tabela 5.2. Procentowy udział współautorów w publikacji „Decision Confidence Assessment in Multi-Class Classification"

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Michał Bukowski	70%	70	2,6929	• koncepcja • metodologia • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja	[illegible]
2	Jarosław Kurek	10%	10	0,3847	• koncepcja (współpraca) • metodologia (współpraca) • nadzór • administracja • walidacja	[illegible]
3	Izabella Antoniuk	10%	10	0,3847	• koncepcja (współpraca) • metodologia (współpraca) • pierwszy szkic • edycja i korekta	[illegible]
4	Albina Jegorowa	10%	10	0,3847	• koncepcja (współpraca) • metodologia (współpraca) • akwizycja danych • zarządzanie danymi	[illegible]
	Suma	**100%**	**100**	**3,847**	---	---

5.3. Publikacja 3: Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification

Tabela 5.3. Procentowy udział współautorów w publikacji „Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification"

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Michał Bukowski	80%	80	0,96	• koncepcja • metodologia • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja • pozyskanie i utrzymanie danych	[illegible]
2	Izabella Antoniuk	10%	10	0,12	• pierwszy szkic • edycja i korekta	[illegible]
3	Jarosław Kurek	10%	10	0,12	• nadzór • administracja • walidacja	[illegible]
	Suma	**100%**	**100**	**1,200**	---	---

5.4. Publikacja 4: Multiple input CNN architecture for tool state recognition in the milling process based on time series signals

Tabela 5.4. Procentowy udział współautorów w publikacji „Multiple input CNN architecture for tool state recognition in the milling process based on time series signals"

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Michał Bukowski	60%	42	0,42	• koncepcja • metodologia • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja	[illegible]
2	Izabella Antoniuk	10%	7	0,07	• pierwszy szkic	[illegible]
3	Karol Szymanowski	10%	7	0,07	• akwizycja danych • zarządzanie danymi	[illegible]
4	Artur Krupa	10%	7	0,07	• edycja i korekta	[illegible]
5	Jarosław Kurek	10%	7	0,07	• nadzór • administracja • walidacja	[illegible]
	Suma	**100%**	**70**	**0,7**	---	---

5.5. Publikacja 5: A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard

Tabela 5.5. Procentowy udział współautorów w publikacji „A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard"

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Michał Bukowski	80%	56	0	• koncepcja • metodologia • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja • edycja i korekta	[illegible]
2	Albina Jegorowa	10%	7	0	• akwizycja danych • zarządzanie danymi	[illegible]
3	Jarosław Kurek	10%	7	0	• pierwszy szkic • walidacja • nadzór • administracja	[illegible]
	Suma	**100%**	**70**	**0**	---	---

5.6. Publikacja 6: Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard

Tabela 5.6. Procentowy udział współautorów w publikacji „Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard"

#	Autor	Udział	Punkty	IF	Opis udziału	Podpis
1	Michał Bukowski	70%	70	2,73	• koncepcja • metodologia • badanie (ang. *research*) • analiza formalna • implementacja • wizualizacja • edycja i korekta	[illegible]
2	Jarosław Kurek	10%	10	0,39	• pierwszy szkic • nadzór • administracja	[illegible]
3	Bartosz Świderski	10%	10	0,39	• walidacja	[illegible]
4	Albina Jegorowa	10%	10	0,39	• akwizycja danych • zarządzanie danymi	A. Jegorowa
	Suma	**100%**	**100**	**3,90**	---	---

5.7. Bibliometria autora wyliczona z uwzględnieniem procentowego wkładu w publikacje wchodzące w skład niniejszej rozprawy doktorskiej

Tabela 5.7. Bibliometria autora wyliczona w uwzględnieniem procentowego wkładu w publikacje wchodzące w skład niniejszej rozprawy doktorskiej

#	Tytuł publikacji	Udział	Punkty	IF
1	Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes	50%	50	1,788
2	Decision Confidence Assessment in Multi-Class Classification	70%	70	2,6929
3	Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification	80%	80	0,96
4	Multiple Input CNN Architecture for Tool State Recognition in Milling Process Based on Time Series Signals	60%	42	0,42
5	A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard	80%	56	0
6	Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard	70%	70	2,73
	Średnia / Suma	**68%**	**368**	**8,5909**

6. Sylwetka i dorobek autora

Autor niniejszej rozprawy urodził się w 1992 roku w Warszawie. W 2015 roku ukończył licencjat na Wydziale Matematyki, Informatyki i Mechaniki Uniwersytetu Warszawskiego na kierunku informatyka. Tytuł pracy licencjackiej to „Granica między frontendem a backendem w czasach JavaScriptu na podstawie aplikacji urlopowej". Następnie w 2017 roku ukończył studia magisterskie na tym samym wydziale. Tytuł pracy magisterskiej to „Analysis and modeling of terrorist networks". W 2020 roku rozpoczął pracę badawczą nad zaawansowanymi metodami sztucznej inteligencji w zastosowaniu do klasyfikacji stanu zużycia narzędzia. W 2024 roku rozpoczął pracę w Katedrze Sztucznej Inteligencji, Instytucie Informatyki Technicznej Szkoły Głównej Gospodarstwa Wiejskiego (SGGW) jako asystent badawczo-dydaktyczny.

Autor jest zarejestrowany w kilku kluczowych bazach danych naukowych, co ułatwia dostęp do jego publikacji i śledzenie jego wkładu w naukę. Poniżej przedstawiono identyfikatory autora do poszczególnych baz:

1. ORCID: 0000-0003-1567-879X
2. Google Scholar ID: kX4tob4AAAAJ
3. Scopus Author ID: 57220550822
4. Web of Science ResearcherID: JZD-9561-2024

Tabela 6.1 prezentuje bibliometryczny przegląd dorobku naukowego autora, ujmując kluczowe wskaźniki jego aktywności badawczej w różnych bazach danych naukowych. Tabela zawiera zbiór danych odnoszących się do liczby publikacji, liczby cytowań oraz indeksu Hirscha (ang. *h-index*), które zostały zgromadzone na podstawie trzech renomowanych źródeł: Web of Science (WoS), Scopus oraz Google Scholar.

W pierwszej kolumnie tabeli wymieniono nazwy baz danych, w których zarejestrowano dorobek naukowy autora. Kolejna kolumna przedstawia liczbę publikacji autora odnotowanych w każdej z wymienionych baz, co pozwala na ocenę jego produktywności badawczej. Trzecia kolumna zawiera informacje o liczbie cytowań wszystkich artykułów autora, co jest istotnym wskaźnikiem rozpoznawalności oraz wpływu jego pracy na rozwój dyscypliny informatyka techniczna i telekomunikacja. W ostatniej kolumnie zaprezentowano indeks Hirscha, który stanowi zbalansowaną miarę produktywności i wpływu naukowca, uwzględniającą jednocześnie liczbę publikacji i ich cytowalność.

Analiza zawarta w tabeli 6.1 pozwala na wielowymiarową ocenę wkładu autora niniejszej rozprawy w rozwój nauki, ukazując nie tylko ilość jego pracy, ale także jej jakość i znaczenie. Przedstawione wskaźniki świadczą o solidnym dorobku naukowym i uznaniu w środowisku akademickim, co stanowi podstawę do dalszej dyskusji i analizy w kontekście tego rozdziału.

Tabela 6.1. Bibliometria Autora

Baza	Liczba publikacji	Liczba cytowań	Indeks Hirscha
Web of Science (WoS)	13	72	5
Scopus	13	82	6
Google Scholar	15	96	6

W poniższej tabeli 6.2 przedstawiono szczegółową analizę dorobku naukowego autora w postaci spisu publikacji. Tabela skupia się na chronologicznym zestawieniu 13 artykułów naukowych opublikowanych w latach 2020-2024, ukazując ewolucję tematyki badawczej autora oraz jego wkład w rozwój dyscypliny informatyki technicznej i telekomunikacji, ze szczególnym uwzględnieniem zastosowań w przemyśle.

Początkowe publikacje z 2020 roku, takie jak artykuły opublikowane w czasopiśmie „BioResources" oraz w „Sensors", koncentrują się na monitorowaniu stanu narzędzi (wierteł) poprzez analizę obrazów otworów wykonanych w płytach wiórowych. W tych pracach eksplorowane są efektywne metody przetwarzania obrazów oraz sieci neuronowe, takie jak sieci syjamskie, w celu rozpoznania zużycia wiertła, co wskazuje na innowacyjne podejście do diagnozowania narzędzi w przemyśle drzewnym.

W 2021 roku badania skupiały się na zaawansowanej klasyfikacji i ocenie pewności decyzji w systemach wieloklasowych, co jest odzwierciedlone w artykule opublikowanym w „Sensors". Autor kontynuuje swoje zainteresowania związane z głębokim uczeniem, jak widać w publikacji w „Wood Science and Technology", gdzie zastosowano metody głębokiego uczenia do klasyfikacji zużycia wiertła na podstawie obrazów otworów.

W roku 2022 prowadzono dalsze badania w zakresie monitorowania stanu narzędzi, tym razem z wykorzystaniem automatycznej, sygnałowej oceny stanu narzędzi, jak opisano w „BioResources". Ta praca rozszerza metodologię o analizę sygnałów, co pokazuje ewolucję podejścia badawczego autora ku bardziej zintegrowanym rozwiązaniom.

Publikacje z lat 2023-2024 koncentrują się na zaawansowanych metodach ekstrakcji cech z obrazów oraz na innowacyjnych architekturach sieci neuronowych, takich jak sieci kapsułowe i transformery wizualne (ViT), do klasyfikacji obrazów. Prace te, opublikowane w „Bulletin of the Polish Academy of Sciences Technical Sciences", „Sensors", „Forests", „Operations Research and Decisions" i „Przeglądzie Elektrotechnicznym", prezentują nowatorskie podejścia do analizy obrazów w kontekście diagnostyki zużycia narzędzi, co świadczy o głębokim zrozumieniu i kreatywnym zastosowaniu najnowszych technologii w dziedzinie uczenia maszynowego.

Podsumowując, tabela 6.2, z chronologicznym spisem wszystkich publikacji autora, prezentuje jego znaczący wkład w badania w dyscyplinie informatyka techniczna i telekomunikacja. Sumaryczna liczba punktów przyznanych publikacjom autora niniejszej rozprawy wynosi 1340. Jest to wskaźnik kwantyfikujący duży wkład autora w dyscyplinę informatyka techniczna i telekomunikacja. Sumaryczny *impact factor* (IF) dla wszystkich publikacji (dla których IF był dostępny) wynosi 32,635. Ten wskaźnik jest sumą indywidualnych wartości *impact factor* dla poszczególnych czasopism, w których opublikowano prace, co odzwierciedla wpływ i prestiż tych publikacji w środowisku naukowym.

Warto zauważyć, że nie wszystkie publikacje mają przypisany *impact factor*, co jest typowe dla artykułów opublikowanych w czasopismach, które nie są indeksowane w bazach danych IF. Pomimo tego wysoki sumaryczny IF w połączeniu z imponującą liczbą punktów świadczy o znaczącym wkładzie autora niniejszej rozprawy w naukę, zwłaszcza w dyscyplinę informatyka techniczna i telekomunikacja oraz jej zastosowań w praktycznych aspektach przemysłu.

Tabela 6.2. Chronologiczny spis wszystkich publikacji autora

#	Rok	Punkty	IF	Tytuł publikacji
1	2020	100	1,614	Jegorowa, A.; Antoniuk, I.; Kurek, J.; Bukowski, M.; Dołowa, W.; Czarniak, P. (2020). Time-efficient approach to drill condition monitoring based on images of holes drilled in melamine faced chipboard. BioRes 2000, 15(4), 9611-9624.
2	2020	100	3,576	Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M, Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978, https://doi.org/10.3390/s20236978
3	2021	100	3,847	Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A, Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834, https://doi.org/10.3390/s21113834
4	2021	200	2,898	Jegorowa, A.; Kurek, J.; Antoniuk, I.; Dołowa, W.; Bukowski, M.; Czarniak, P., Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. Wood Sci Technol 2021, 55, 271-293, https://doi.org/10.1007/s00226-020-01245-7

Kontynuacja na następnej stronie

Tabela 6.2 – kontynuacja z poprzedniej strony

#	Rok	Punkty	IF	Tytuł publikacji
5	2022	100	1,500	Świderski, B.; Antoniuk, I.; Kurek, J.; Bukowski, M.; Górski, J.; Jegorowa, A., Tool condition monitoring for the chipboard drilling process using automatic, signal-based tool state evaluation. BioResources 2022, 17(3), 5349-5371.
6	2023	100	3,900	Antoniuk, I.; Kurek, J.; Krupa, A.; Wieczorek, G.; Bukowski, M.; Kruk, M.; Jegorowa, A., Advanced Feature Extraction Methods from Images of Drillings in Melamine Faced Chipboard for Automatic Diagnosis of Drill Wear. Sensors 2023, 23, 1109, https://doi.org/10.3390/s23031109
7	2023	100	1,200	Bukowski, M.; Antoniuk, I.; Kurek, J., Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification. Bulletin of the Polish Academy of Sciences Technical Sciences 2023, 71(6), e147338. https://doi.org/10.24425/bpasts.2023.147338
8	2023	100	2,900	Jegorowa, A.; Kurek, J.; Antoniuk, I.; Krupa, A.; Wieczorek, G.; Świderski, B.; Bukowski, M.; Kruk, M., Automatic Estimation of Drill Wear Based on Images of Holes Drilled in Melamine Faced Chipboard with Machine Learning Algorithms. Forests 2023, 14, 205, https://doi.org/10.3390/f14020205
9	2023	100	3,900	Kurek, J.; Krupa, A.; Antoniuk, I.; Akhmet, A.; Abdiomar, U.; Bukowski, M.; Szymanowski, K., Improved Drill State Recognition during Milling Process Using Artificial Intelligence. Sensors 2023, 23, 448, https://doi.org/10.3390/s23010448
10	2024	70	0,7	Bukowski, M.; Antoniuk, I.; Szymanowski, K.; Krupa, A.; Kurek, J., Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024, 34(3), 41-60, https://doi.org/10.37190/ord240303
11	2024	70	0	Bukowski, M.; Jegorowa, A.; Kurek, J., A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard. Przeglad Elektrotechniczny 2024, 5.

Kontynuacja na następnej stronie

Tabela 6.2 – kontynuacja z poprzedniej strony

#	Rok	Punkty	IF	Tytuł publikacji
12	2024	100	3,900	Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A., Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092. https://doi.org/10.3390/s24041092
13	2024	100	2,700	Kurek, J.; Latkowski, T.; Bukowski, M.; Świderski, B.; Łępicki, M.; Baranik, G.; Nowak, B.; Zakowicz, R.; Dobrakowski, Ł., Zero-Shot Recommendation AI Models for Efficient Job–Candidate Matching in Recruitment Process. Appl Sci 2024, 14, 2601, https://doi.org/10.3390/app14062601
Razem		**1340**	**32,635**	

7. Omówienie i dyskusja wyników

Czwarta rewolucja przemysłowa [10] zainicjowała transformację w sektorze produkcyjnym i przemysłowym, wprowadzając zaawansowane technologie cyfrowe, takie jak sztuczna inteligencja (AI), internet rzeczy (IoT), robotyka oraz druk 3D. Te innowacje przynoszą bezprecedensowe zmiany w sposobie projektowania, produkcji, operacji i serwisowania produktów oraz w zarządzaniu łańcuchem dostaw.

Sztuczna inteligencja, będąc kluczowym komponentem tej rewolucji, umożliwia maszynom analizowanie ogromnych ilości danych w celu optymalizacji procesów, przewidywania awarii i automatyzacji decyzji. Dzięki temu przemysł staje się bardziej elastyczny, efektywny i zorientowany na potrzeby klienta.

Wpływ tych technologii na obecny stan przemysłu jest ogromny. Przedsiębiorstwa, które adaptują się do tych zmian, zyskują znaczącą przewagę konkurencyjną, poprzez zwiększenie wydajności, redukcję kosztów operacyjnych i lepsze zrozumienie potrzeb swoich klientów.

Wprowadzenie metod sztucznej inteligencji do przemysłu wiąże się z szeregiem wyzwań, które przedsiębiorstwa muszą pokonać, aby w pełni wykorzystać potencjał tej technologii. Jednym z takich wyzwań jest koszt sensorów i innych urządzeń niezbędnych do gromadzenia danych, które sztuczna inteligencja może analizować. Wysokie inwestycje początkowe mogą stanowić barierę, zwłaszcza dla mniejszych przedsiębiorstw, które dysponują ograniczonymi zasobami finansowymi. Kolejnym problemem jest kwestia interpretacji wyników generowanych przez algorytmy sztucznej inteligencji. Pomimo że metody sztucznej inteligencji mogą przetwarzać i analizować ogromne ilości danych z niezwykłą precyzją, wyniki te nie zawsze są łatwe do zrozumienia dla człowieka, co może prowadzić do trudności w podejmowaniu na ich podstawie właściwych decyzji biznesowych. Dodatkowo istnieje obawa wśród pracowników, że automatyzacja i robotyzacja spowodują zmniejszenie liczby ludzkich miejsc pracy, co prowadzi do strachu przed utratą zatrudnienia i oporu przed wdrażaniem nowych technologii. W niniejszej pracy pokazano, jak poradzić sobie z wyżej wymienionymi wyzwaniami.

Wykorzystanie architektury sieci syjamskich w kontekście klasyfikacji zużycia wiertła dla przemysłu meblarskiego [15] jest przykładem skutecznego wdrożenia zaawansowanych architektur sieci neuronowych do rozpoznawania stanu narzędzi. Podejście to wykorzystuje mocne strony sieci syjamskich, znanych ze swej zdolności do uczenia się na podstawie danych porównawczych, co czyni je szczególnie odpowiednimi do zadań, w których kluczowe jest rozróżnienie między bardzo podobnymi klasami. Ta architektura sieci wyróżnia się skalowalnością i wszechstronnością, oferując elastyczność adaptacji do nowych warunków za pomocą uczenia transferowego (ang. *transfer learning*). Oznacza to, że model można łatwo przeszkolić na nowo, aby rozpoznawał zużycie w różnych narzędziach lub materiałach bez konieczności rozpoczynania budowy modelu od zera, co stanowi dodatkową zaletę w procesie implementacji tego rozwiązania w procesie produkcyjnym. Model pracuje jedynie na zdjęciach odwiertów, co pozwala łatwo wdrożyć go w środowisku produkcyjnym, bez ponoszenia dużych kosztów. Ponadto integracja technik specyficznych dla problemu, takich jak analiza sekwencji pomiarów w celu eliminacji obserwacji odstających, zapobiega kosztownym pomyłkom między krytycznymi klasami (w tym przypadku między klasą czerwoną i zieloną), osiągając liczbę 37 krytycznych pomyłek na 8526 próbek. Wprowadzenie parametru okna pozwala podnieść metrykę dokładności (ang. *accuracy*) do 82% dla sieci syjamskich w stosunku do bazowych 69%, osiąganych poprzednimi metodami, które w dodatku generalizują się na nowe klasy i wymagają drogiego procesu przetrenowywania przy każdej większej zmianie. Wyniki te potwierdzają pierwszą hipotezę badawczą, wykazując, że zastosowanie architektury sieci syjamskich umożliwia osiągnięcie wysokiej dokładności klasyfikacji zużycia narzędzi, jednocześnie minimalizując błędy między krytycznymi klasami. Wprowadzanie technik dostosowanych do konkretnego problemu produkcyjnego, takich jak analiza sekwencyjna, skutkuje jedynie 0,44% błędów między krytycznymi klasami, a także zwiększa dokładność klasyfikacji o 13 punktów procentowych względem tradycyjnych rozwiązań.

Jak wspomniano we wcześniejszych rozdziałach, nie zawsze istnieje możliwość zintegrowania danych z wielu sensorów, szczególnie na początkowych etapach wdrożenia technik sztucznej inteligencji w procesach produkcyjnych. Jak pokazano w [13], kiedy do dyspozycji są tylko zdjęcia odwiertów, dokładność może spaść do 69,78%. Badania przeprowadzone przez autora [6] doprowadziły do wypracowania metodologii zwiększenia dokładności i interpretowalności zadań klasyfikacyjnych poprzez integrację oceny pewności decyzji w procesie. Podejście to odbiega od tradycyjnej, sztywnej klasyfikacji, która przypisuje jedną klasę każdej instancji problemu, poprzez ocenę poziomu pewności dla każdej decyzji klasyfikacyjnej. Metoda ta dostarcza mechanizm do określenia, czy próbka powinna być automatycznie sklasyfikowana, czy też należy podać ją do manualnej rewizji przez eksperta. System ten nie tylko ma na celu zmniejszenie obciążenia pracą ekspertów ludzkich przez filtrowanie przypadków, w których klasyfikator jest wystarczająco pewny, ale także zapewnia, że bardziej złożone lub niejednoznaczne przypadki będą analizowane przez ekspertów. Taki hybrydowy model współpracy maszyny z człowiekiem wykorzystuje mocne strony zarówno systemów automatycznych, jak i ekspertyzy ludzkiej, prowadząc do usprawnień w ogólnej dokładności i niezawodności procesu klasyfikacji. Co więcej, prezentowana technika, przez efektywne wykorzystanie metryk pewności, oferuje obiecującą drogę do udoskonalenia aplikacji uczenia maszynowego, wspomagając procesy decyzyjne i sprzyjając produktywnej współpracy między systemami automatycznymi a operatorami ludzkimi. Odrzucanie niepewnych przypadków poprzez poddanie ich ludzkiej ocenie powoduje, że pozostawienie 30% problematycznych próbek ocenie ludzkiej pozwala zwiększyć dokładność do 74%, zwiększając zaufanie do całego systemu. Przeprowadzone badania pozytywnie weryfikują drugą hipotezę badawczą, pokazując, że integracja oceny pewności decyzji z modelami sztucznej inteligencji skutkuje poprawą dokładności klasyfikacji zużycia narzędzi, bez modyfikacji samego modelu decyzyjnego. Poprzez dostrojenie parametrów systemu oceny pewności można jednocześnie zwiększyć dokładność modelu, jego interpretowalność, a także zaufanie do całego systemu poprzez połączenie wiedzy eksperckiej pracowników oraz dowolnej architektury modelu sztucznej inteligencji.

Autor pracy badał również inne obiecujące architektury, które można zastosować do problemu oceny jakości narzędzi. Jedna z architektur, sieci kapsułowe, została zaprojektowana w celu przeciwdziałania istniejącym ograniczeniom klasycznych sieci konwolucyjnych (CNN). Główną zaletą sieci kapsułowych jest ich zdolność do zachowania informacji o przestrzennej hierarchii między cechami, co oznacza, że mogą one lepiej rozumieć złożone przypadki danych wejściowych. To sprawia, że są bardziej efektywne w zadaniach wymagających rozpoznania wzorców z uwzględnieniem perspektywy i orientacji obiektów, co było wyzwaniem dla tradycyjnych CNN. Mimo tych zalet zastosowanie sieci kapsułowych do problemu oceny jakości wierteł poprzez ocenę zdjęć odwiertów nie przyniosło spodziewanych rezultatów, pozwalając osiągnąć jedynie 60% dokładności (ang. *accuracy*). Tak niski wynik może wynikać z wyzwań związanych z właściwym dostosowaniem architektury sieci kapsułowych do specyfiki danych dotyczących zużycia wierteł, a mianowicie zdjęcia odwiertów nie posiadają odpowiednich cech przestrzennych, których rozpoznawanie i umiejscowienie jest zaletą modelu sieci kapsułowych. Jednakże badania autora nad sieciami kapsułowymi doprowadziły do ich wykorzystania w innym obszarze, a mianowicie w problemie rozpoznawania japońskiej hiragany [3]. W tym kontekście zastosowanie zmodyfikowanej wersji sieci kapsułowych ze zwiększoną liczbą warstw pozwoliło osiągnąć dokładność zbliżoną do najlepszych dostępnych modeli, przy użyciu tylko niewielkiego ułamka parametrów i mocy obliczeniowej, które są obecne w innych modelach. To pokazuje, że mimo pewnych ograniczeń w konkretnych zastosowaniach – sieci kapsułowe mogą oferować znaczące korzyści, zwłaszcza w zadaniach związanych z rozpoznawaniem wzorców w złożonych strukturach danych, i warto próbować je stosować w problemach oceny stanu narzędzi, gdzie do dyspozycji są zdjęcia zawierające cechy przestrzenne, w których ich położenie względem siebie stanowi istotę problemu.

Wyniki te potwierdzają trzecią hipotezę badawczą, wykazując, że mechanizm uwagi zastosowany do sieci kapsułowych pozwala na wykorzystanie tego modelu na niewyspecjalizowanych stacjach roboczych. Szybkie prototypowanie rozwiązań pozwala na lepszą współpracę między badaczami a przemysłem, dzięki czemu można zainteresować potencjalnego odbiorcę rozwiązania działającym systemem, który nie wymaga zastosowań chmury obliczeniowej ani drogich i nie zawsze dostępnych kart graficznych.

Integracja danych z wielu czujników z konwolucyjnymi sieciami neuronowymi (CNN) stanowi znaczący postęp w monitorowaniu i utrzymaniu stanu narzędzi w środowiskach przemysłowych. To innowacyjne podejście wykorzystuje architekturę modelu CNN z 11 wejściami [7], dostosowanego do rozpoznawania stanu zużycia narzędzi w procesie frezowania. Poprzez przekształcanie sygnałów czasowych z różnych czujników na obrazy skalogramów za pomocą ciągłej transformacji falkowej – CNN jest w stanie wydobyć szczegółowe, subtelne cechy, które dokładnie reprezentują stan narzędzi w procesie frezowania. Ta metodyka wykazała wyjątkową skuteczność, osiągając dokładność na poziomie 96,00% w klasyfikacji zużycia narzędzi. Taki system nie tylko poprawia efektywność konserwacji narzędzi, ale także znacząco redukuje koszty operacyjne i zapewnia jakość końcowego produktu poprzez minimalizację błędów w kategoryzacji zużycia narzędzi. Podejście to wymaga jednak zastosowania wielu sensorów, które wiążą się z komplikacją procesu i wzrostem kosztów.

Przedstawiony rezultat pozytywnie weryfikuje czwartą i piątą hipotezę badawczą, prezentując wysoką dokładność modelu wykorzystującego złożone czujniki do badania stanu narzędzia. Sygnały czasowe poddane ciągłej transformacji falkowej są przykładem procesu inżynierii i ekstrakcji cech, który łączy zalety architektury sieci konwolucyjnej, jak i eksperckiej wiedzy badacza w dziedzinie, którą modeluje. Pokazuje to szerokie spektrum zastosowań różnych modeli w zależności od dostępnego budżetu i obecnych na miejscu czujników,

Najlepszym modelem oceniającym stan wiertła okazał się model oparty na architekturze Vision Transformer [5]. Osiągnął on, bez wykorzystywania dodatkowych technik, dokładność 71,14% co jest wynikiem o 1,36 p.p. lepszym niż najlepszy dotychczasowy model. W porównaniu z innymi architekturami, takimi jak sieci kapsułowe, bez problemu poradził on sobie ze specyfiką zbioru danych. Architektura ta wymaga jednak długiego czasu uczenia, mimo wykorzystania sieci już wstępnie wytrenowanej. Proces dostrajania (ang. *fine-tuningu*) wymagał aż 8000 epok, co skutkowało wieloma godzinami obliczeń. Różne architektury użyte do problemu pokazują, z jakimi kompromisami należy sobie radzić w procesie wdrażania technik sztucznej inteligencji do oceny jakości narzędzi.

Ten rezultat jest kolejnym potwierdzeniem pierwszej hipotezy badawczej i dobitnie pokazuje szybki rozwój w dziedzinie architektur sieci neuronowych, który nastąpił w ostatnich 10 latach.

Mimo osiągnięcia zadowalającej dokładności dla wybranego zbioru czujników nie wszystkie błędy klasyfikacji należy traktować z równą uwagą. W niniejszej pracy w celu oceny jakości stanu zużycia wiertła stosuje się 3 klasy: zieloną, żółtą oraz czerwoną. W wyżej wymienionych modelach starano się zminimalizować liczbę krytycznych błędów między czerwoną a zieloną klasą, mogących mieć najpoważniejsze konsekwencje w procesie produkcyjnym. Specjalnie do tego celu można zaprojektować funkcję straty [4], która skupia się na penalizowaniu takich błędów.

Techniki dobierania do problemu odpowiedniej funkcji straty potwierdzają szóstą hipotezę badawczą. Problematyka predykcyjnej konserwacji wymusza na twórcach rozwiązań dokładną analizę problemu i zastosowanie szytych na miarę rozwiązań, poczynając od doboru danych, ekstrakcji cech, architektury modelu, a kończąc na wyborze odpowiedniej funkcji straty.

Integracja danych z różnych czujników i zastosowanie technik sztucznej inteligencji stanowią postęp w monitorowaniu i utrzymaniu stanu narzędzi w przemyśle. Podejście oparte na modelach sieci konwolucyjnej, wykorzystujące obrazy skalogramów uzyskane z sygnałów czasowych z wielu czujników, pozwala na osiągnięcie dokładności klasyfikacji zużycia narzędzi na poziomie 96%. Mimo wyzwań związanych ze złożonością procesu i kosztami czujników metoda ta zapewnia skuteczną i efektywną konserwację narzędzi. Nowoczesne architektury takie jak sieci syjamskie wykazały swoją skuteczność i elastyczność w adaptacji do różnych warunków, natomiast sieci kapsułowe, mimo swoich potencjalnych zalet, nie sprawdziły się w ocenie jakości wierteł, osiągając tylko 60% dokładności. Badania nad sieciami kapsułowymi doprowadziły jednak do znaczących sukcesów w innych obszarach, takich jak rozpoznawanie japońskiej hiragany, pokazując wartość dalszego eksplorowania tych technologii. Transformery wizyjne wykazały najwyższą skuteczność w klasyfikacji zdjęć odwiertów, osiągając dokładność 71,14%. Alternatywne podejścia, takie jak integracja oceny pewności decyzji, oferują możliwość poprawy dokładności i interpretowalności klasyfikacji nawet przy ograniczonych danych – gdzie zastosowanie tego procesu umożliwiło zwiększenie dokładności modelu z 70% do 74% bez ingerencji w proces uczenia ani architekturę.

Autor uważa, że najważniejszymi osiągnięciami oryginalnymi rozprawy są:

1. Opracowanie metod identyfikacji stopnia zużycia narzędzi z wykorzystaniem algorytmów sztucznej inteligencji na podstawie danych wizyjnych i sygnałowych, umożliwiających osiągnięcie wysokiej dokładności klasyfikacji.

2. Skuteczne zastosowanie zaawansowanych architektur sieci neuronowych, takich jak sieci syjamskie, sieci kapsułowe oraz transformatory wizyjne (ang. *vision transformers*), które mogą kompensować brak dostępu do zaawansowanych czujników monitorujących i zapewniają najwyższą dokładność klasyfikacji zużycia narzędzi.

3. Implementacja i optymalizacja głębokich sieci konwolucyjnych (CNN), w tym modeli z wieloma wejściami wykorzystujących obrazy skalogramów dla lepszej klasyfikacji stanu zużycia narzędzi w procesach obróbki skrawaniem.

4. Wykorzystanie technik uczenia transferowego (ang. *transfer learning*) do poprawy dokładności modelu bazującego na zdjęciach odwiertów, z osiągnięciem najlepszych wyników w ocenie jakości wierteł.

5. Integracja mechanizmów oceny pewności decyzji z modelami sztucznej inteligencji, co podnosi zarówno dokładność, jak i interpretowalność wyników oraz ułatwia integrację maszyny i człowieka w procesie oceny stanu narzędzi.

6. Opracowanie i adaptacja niestandardowych funkcji straty dla algorytmu XGBoost, umożliwiających eliminację problemu nierównowagi klas, redukcję najbardziej kosztownych błędów klasyfikacji oraz skuteczną identyfikację krytycznych stanów zużycia narzędzi, co ma kluczowe znaczenie dla zapewnienia ciągłości i jakości procesów produkcyjnych.

8. Podsumowanie i wnioski

W ramach niniejszej pracy doktorskiej przeprowadzono szczegółową analizę oraz rozwinięto metody sztucznej inteligencji ukierunkowane na identyfikację stopnia zużycia narzędzi przemysłowych. Głównym celem badań było opracowanie wydajnych algorytmów i modeli, które przyczynią się do zwiększenia niezawodności i efektywności procesów produkcyjnych.

Podsumowując, udało się osiągnąć następujące cele:

- Rozwinięto i zaimplementowano zaawansowane techniki przetwarzania danych wizyjnych oraz sygnałowych, co pozwoliło na precyzyjną identyfikację stanów zużycia narzędzi.
- Skutecznie zastosowano głębokie sieci neuronowe, w tym sieci konwolucyjne i transformatory wizyjne, co znacząco poprawiło dokładność klasyfikacji zużycia narzędzi.
- Opracowano nowatorskie funkcje straty oraz techniki uczenia maszynowego, które skutecznie radzą sobie z problemem nierównowagi klas w zbiorach danych.
- Wprowadzono innowacyjne podejścia w zakresie oceny pewności decyzji podejmowanych przez algorytmy, co zwiększa ich wiarygodność i użyteczność w praktycznych zastosowaniach.

Wnioski wynikające z przeprowadzonych badań podkreślają znaczenie ciągłego rozwoju i integracji nowych technologii sztucznej inteligencji w przemyśle. Praca ta stanowi wkład w rozwój metod klasyfikacyjnych, które mogą przyczynić się do optymalizacji procesów produkcyjnych, redukcji przestojów oraz zwiększenia ogólnej efektywności produkcji, a tym samym znaczący wkład w dyscyplinę informatyka techniczna i telekomunikacje.

W przyszłości planowane są dalsze badania nad optymalizacją opracowanych algorytmów oraz ich adaptacją do innych zastosowań w różnych sektorach przemysłu, co może otworzyć nowe perspektywy dla zastosowań sztucznej inteligencji w automatyzacji i monitoringu procesów produkcyjnych.

Bibliografia

[1] Sanket N. Bhagat and S.L. Nalbalwar. Labview based tool condition monitoring and control for cnc lathe based on parameter analysis. page 1386 – 1388, 2017.

[2] Francesc Bonada, LLuis Echeverria, Xavier Domingo Albin, and Gabriel Anzaldi Varas. *AI for Improving the Overall Equipment Efficiency in Manufacturing Industry*. 03 2020.

[3] Michał Bukowski, Izabella Antoniuk, and Jarosław Kurek. Improved efficient capsule network for kuzushiji-mnist benchmark dataset classification. *Bulletin of the Polish Academy of Sciences Technical Sciences*, 71(6):e147338, 2023.

[4] Michał Bukowski, Jarosław Kurek, and Izabella Antoniuk. Custom loss functions in xgboost algorithm for enhanced critical error mitigation in drill-wear analysis of melamine faced chipboard. *Sensors*, 1, 2024.

[5] Michał Bukowski, Jarosław Kurek, and Izabella Antoniuk. A novel approach using vision transformers (vit) for classification of holes drilled in melamine faced chipboard. *Przegląd Elektrotechniczny*, 1, 2024.

[6] Michał Bukowski, Jarosław Kurek, Izabella Antoniuk, and Albina Jegorowa. Decision confidence assessment in multi-class classification. *Sensors*, 21(11), 2021.

[7] Michał Bukowski, Jarosław Kurek, and Antoniuk Izabella. Multiple input cnn architecture for tool state recognition in milling process based on time series signals. *Przegląd Elektrotechniczny*, 1, 2024.

[8] Mukunda K Das and Krishnan Rangarajan. Performance monitoring and failure prediction of industrial equipments using artificial intelligence and machine learning methods: A survey. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pages 595–602, 2020.

[9] Mukunda K Das, Krishnan Rangarajan, and Venkatakrishna Tirumala. Performance monitoring of industrial rotary equipment using ai/ml techniques. In *2020 Science and Artificial Intelligence conference (S.A.I.ence)*, pages 37–40, 2020.

[10] Lucreția Dogaru. The main goals of the fourth industrial revolution. renewable energy perspectives. *Procedia Manufacturing*, 46:397–401, 2020. 13th International Conference Interdisciplinarity in Engineering, INTER-ENG 2019, 3–4 October 2019, Targu Mures, Romania.

[11] Milad Elyasi, Simon Thevenin, and Audrey Cerqueus. Use of ai in assembly line design and worker and equipment management: review and future directions. *Flexible Services and Manufacturing Journal*, 2024.

[12] German Herrera-Granados, Takashi Misaka, Jonny Herwan, Hitoshi Komoto, and Yoshiyuki Furukawa. An experimental study of multi-sensor tool wear monitoring and its application to predictive maintenance. *International Journal of Advanced Manufacturing Technology*, 133(7-8):3415 – 3433, 2024.

[13] Albina Jegorowa, Jarosław Kurek, Izabella Antoniuk, Wioleta Dołowa, Michał Bukowski, and Paweł Czarniak. Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. *Wood Sci. Technol.*, 55(1):271–293, jan 2021.

[14] Xu Kun, Zhiliang Wang, Ziang Zhou, and Wang Qi. Design of industrial internet of things system based on machine learning and artificial intelligence technology. *Journal of Intelligent and Fuzzy Systems*, 40:2601–2611, 02 2021.

[15] Jarosław Kurek, Izabella Antoniuk, Bartosz Świderski, Albina Jegorowa, and Michał Bukowski. Application of siamese networks to the recognition of the drill wear state based on images of drilled holes. *Sensors*, 20(23), 2020.

[16] Soham Mehta, Rajput Anurag Singh, Yugalkishore Mohata, and M.B. Kiran. Measurement and analysis of tool wear using vision system. page 45 – 49, 2019.

[17] Goga Alexandru Silviu. The use of artificial intelligence in industrial management. *Mechanisms and Machine Science*, 174 MMS:819 – 828, 2025.

[18] Pan Wang, Liaomo Zheng, and Mengjia Lian. Intelligent cutter state detection platform based on edge-enhanced fcn. page 854 – 857, 2023.

[19] Xitong Wu, Guohe Li, Zhihua Shao, Weijun Liu, and Ganzhong Ma. Advances in research on tool wear online monitoring method. *Recent Patents on Engineering*, 18(6):124 – 139, 2024.

[20] Chang'an Zhou, Kai Guo, and Jie Sun. Sound singularity analysis for milling tool condition monitoring towards sustainable manufacturing. *Mechanical Systems and Signal Processing*, 157, 2021.

Kopie publikacji wchodzących w skład rozprawy doktorskiej

10.1. Publikacja 1

PUBLIKACJA 1

Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes

Jarosław Kurek, Izabella Antoniuk, Bartosz Świderski, Albina Jegorowa, Michał Bukowski

Sensors 2020, 20, 6978

https://doi.org/10.3390/s20236978

Punktacja: 100, IF: 3,576

sensors

Letter

Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes

Jarosław Kurek [1,*], Izabella Antoniuk [1], Bartosz Świderski [1], Albina Jegorowa [2] and Michał Bukowski [3]

1 Institute of Information Technology, Warsaw University of Life Sciences, Nowoursynowska 159, 02-776 Warsaw, Poland; izabella_antoniuk@sggw.edu.pl (I.A.); Bartosz_Swiderski@sggw.edu.pl (B.S.)
2 Institute of Wood Sciences and Furniture, Warsaw University of Life Sciences, Nowoursynowska 159, 02-776 Warsaw, Poland; albina_jegorowa@sggw.edu.pl
3 Department of Artificial Intelligence, Institute of Information Technology, Warsaw University of Technology, 02-776 Warsaw, Poland; michal.bukowski@buksoft.pl
* Correspondence: jaroslaw_kurek@sggw.edu.pl; Tel.: +48-505-482-708

Received: 6 November 2020; Accepted: 3 December 2020; Published: 6 December 2020

Abstract: In this article, a Siamese network is applied to the drill wear classification problem. For furniture companies, one of the main problems that occurs during the production process is finding the exact moment when the drill should be replaced. When the drill is not sharp enough, it can result in a poor quality product and therefore generate some financial loss for the company. In various approaches to this problem, usually, three classes are considered: green for a drill that is sharp, red for the opposite, and yellow for a tool that is suspected of being worn out, requiring additional evaluation by a human expert. In the above problem, it is especially important that the green and the red classes not be mistaken, since such errors have the highest probability of generating financial loss for the manufacturer. Most of the solutions analysing this problem are too complex, requiring specialized equipment, high financial investment, or both, without guaranteeing that the obtained results will be satisfactory. In the approach presented in this paper, images of drilled holes are used as the training data for the Siamese network. The presented solution is much simpler in terms of the data collection methodology, does not require a large financial investment for the initial equipment, and can accurately qualify drill wear based on the chosen input. It also takes into consideration additional manufacturer requirements, like no green-red misclassifications, that are usually omitted in existing solutions.

Keywords: Siamese network; contrastive loss function; convolutional neural networks; deep learning; tool condition monitoring

1. Introduction

Drill wear state recognition belongs to the larger group of problems called tool condition monitoring, which deals with the evaluation of different machine parts' condition, as well as determining how long they can be used in the production process. Depending on the properties of each tool, as well as the requirements of the final product, different signals can be recorded and later tested using various methods, to obtain the final evaluation. Quite a few procedures in this direction also deal with the main topic of this paper, which is drill wear state recognition. From the manufacturer's point of view, when the drill starts to become dull, it should be replaced as quickly as possible. Extending the use time of such a tool can result in poor product quality and therefore generate financial loss for the company. Manual evaluation of the drill state is possible and was initially done during the production process, but this is very time consuming, resulting in the prolongation of the entire procedure. A faster and more automated approach was needed, which resulted in extensive research on this subject. For example, one of the existing solutions focuses on measuring tool wear using two approaches: conventional methods and estimation with a customized software combining artificial neural networks and flank wear image recognition [1]. In this article, the authors noted that the automatic solution, while achieving slightly worse results, was still within the same range, and they presented some additional advantages, such as the lower cost.

Existing solutions vary greatly in their approach, especially the data collection methodology. As is often the case, especially when it comes to the usage of specialized equipment, the most visible advancements have been made in medicine. For example, in [2], the authors showed a miniaturized version of a device used for wireless intraoral force monitoring, which is capable of collecting and transmitting the required signals. In this field as well, image recognition and processing methodologies are used more often than in the wood industry. In [3], the authors automatically classified histological images using a histological ontology, and with the introduced improvements, they were able to recognize epithelial tissue, which was previously impossible. In the case of the wood industry, a significant number of solutions are sensor based, without including the possible advantages that the images of the checked element in the monitoring process can bring. As presented in [4], for a solution to find application in a commercial system, it usually needs to meet a series of different properties, dependent on the manufacturer's requirements, while overall, solution complexity is not desirable. An additional solution requirement is an approach with the main focus on evaluating the tool state without disrupting the actual manufacturing process [5]. Originally, different sensors were used to measure such signals as feed force, noise, vibrations, acoustic emission, cutting torque, and others [6]. Such a solution, apart from the different devices used to collect the initial signals, also requires numerous preprocessing stages before obtaining samples that can be used. At each point, additional evaluation is required, to check if the chosen sensors are appropriate for the current work environment, if the registered signals are the correct ones, and if possibly the best diagnostic features are generated from them. Finally, from the initial set of features, a subset needs to be chosen for building the final classification model. What needs to be noted is that in the case of such solutions, not only the initial setup is complicated and costly from the manufacturer's point of view, but additionally, an error at any key point can result in the final solution being unsatisfactory (in the case of choosing the wrong signal selection, apart from the resulting time loss, any costs for the required sensors should be treated as a financial loss). There are also a few different solutions that present an interesting approach to this problem. In [7], the authors used a large number of signals, extracted both from the signal and frequency domains, along with their wavelet coefficients, and evaluated them automatically to check how relevant they were to the presented problem based on the chosen set of properties. After the initial selection of the features used to estimate the tool wear, based on the final accuracy, the sensor and signal usability were evaluated. Similarly, in [8], various signals were registered over a wide range of cutting conditions, including such elements as thrust force and torque. In this case, the authors used a back-propagation neural network to predict the flank wear of a drill bit. In the solution presented in [9] as well, multiple sensors were used to collect the data, which were later integrated through a fuzzy logic model. Such a model is a combination of artificial neural networks and fuzzy logic and is capable of self-organization, self-adjustment, as well as learning from experience, and according to the authors, it was able to significantly increase the accuracy of tool wear estimation when compared to conventional approaches. In [10], the k-nearest neighbours algorithm was used in order to classify the drill condition into one of three initial classes. What is worth noting is that even while using diverse signals and different features, those solutions still obtained an accuracy below the 90% threshold. They also did not take into account the additional manufacturer requirements, such as minimizing the misclassification rate between the crucial red and green classes. The overall uncertainty of such solutions, when combined with the complicated setup and high initial costs, usually renders them inapplicable to real work environments.

The solution presented in this paper takes into account different approaches to deep learning in general (for elements such as the overall methods and applications [11], different architectures with AI applications as their main focus [12], or concerning deep learning in neural networks in general [13]), as well as the authors' previous works on this subject [14–17], along with a few crucial observations that were made during that process. Firstly, instead of a set of specialized sensors, images of drilled holes are used as the input data for the presented algorithms. Since in that case, the only external piece of equipment required during the data collection process is a simple camera, the work needed at this stage is greatly minimized, and the initial costs are reduced. Different approaches to this subject were tested, using solutions based mainly on convolutional neural networks (CNNs; which do not require specialized diagnostic features and are currently considered some of top solutions when it comes to image recognition [18]). Initially, experiments were performed on limited set of data [14], with an additional approach checking the influence of the artificial data extension for the case with two classes [15]. Those first approaches proved to be quite promising, and in the following works, the data augmentation technique was combined with the transfer learning methodology [16]. In this case, the results were better than with much more complicated setups, using various sensors to collect the related signals, with the accuracy exceeding 93%. Another solution [17] this time incorporating classifier ensemble with different pretrained networks further improved the overall classification rate and obtained over 95% accuracy (including approaches attempting to beat the ImageNet large-scale visual recognition challenge [19], using solutions such as deep convolutional neural networks [20] or other pretrained networks, like AlexNet (available online) [21]).

While the above approaches showed that a high accuracy rate is possible even with a much simpler setup than the initial, sensor based solutions related to this subject, they still failed to incorporate the additional manufacturer requirement regarding the green-red misclassification rate. Usually, three classes are considered for this problem: the green class, describing a drill that is in good shape and that can be further used in the production process, the yellow class, for tools that are suspected of being worn, which requires additional evaluation by human expert, and finally, the red class, which describes the dull elements that should be replaced immediately, since their further use in the production process can result in poor product quality and financial loss. From the manufacturer's point of view, a clear distinction between the red and green class is far more important than the other classifications. In the case of the yellow class in the standard approach, it will still be evaluated by a human expert, but if the green tool is classified as red, it will be discarded, while in the opposite case, a dull drill will be further used in the production process, resulting in holes with chipped edges. Enforcing this requirement has a higher impact on the overall solution adaptation to the actual work environment than the high overall accuracy.

The solution presented in this paper focuses on those aspects. The same type of input data are used as in the previous approaches, with only images of drilled holes being used as the training samples, without any additional signals. Furthermore, it is more focused on the green-red misclassification requirement. The Siamese network is used to build the main classifier, in order to ensure the best possible classification. This type of network was initially introduced for the face recognition problem and with CNN as its base and can be easily adjusted to drill wear state evaluation. The application of such a solution to the problem of drill wear classification was not previously studied, but the solution has obtained promising results. What is more, the presented method is highly adjustable to the changing parameters of the work environment, including changes to the samples' characteristics.

This paper is organized as follows. The data collection methodology, as well as the obtained dataset are described in Section 2. Section 3 outlines the data preprocessing used to balance the dataset, as well as additional, required operations. Siamese networks in general, as well as the presented solution specifically are described in Section 4. Section 5 presents the results obtained during the experiments and a discussion about their quality in relation to previous solutions and possible areas for future work. Conclusions follow in Section 6.

2. Dataset

The images used as the input for the presented algorithm were made using a Nikon D810 (Nikon Corporation, Shinagawa, Tokyo, Japan) single-lens reflex digital camera with a 35.9 × 24.0 mm CMOS image sensor. The entire process was performed in cooperation with the Institute of Wood Sciences and Furniture at Warsaw University of Life Sciences, Poland. For test purposes, a standard CNC vertical machine centre (Busellato Jet 100, Thiene, Italy) was used. Drilling was performed on a standard, melamine-faced chipboard (Kronopol U 511 SM; Swiss Krono Sp. z o. o., Żary, Poland), which is typically used in the furniture industry. The dimensions of the test piece were 300 × 35 × 18 mm. A regular, Faba WP-01 double-blade drill for through drilling (Baboszewo, Poland) equipped with a tungsten carbide tip was used. The drill's overall length was 70 mm, with a shank length equal to 25 mm, a flute length of 40 mm, a shank diameter of 10 mm, and a 12 mm drill diameter. The clearance angle on the drill face was −15.45 degrees. The rake angle was equal to 0 degrees, and the helix angle for the tool used was 15 degrees. The images of the equipment used are presented in Figure 1.

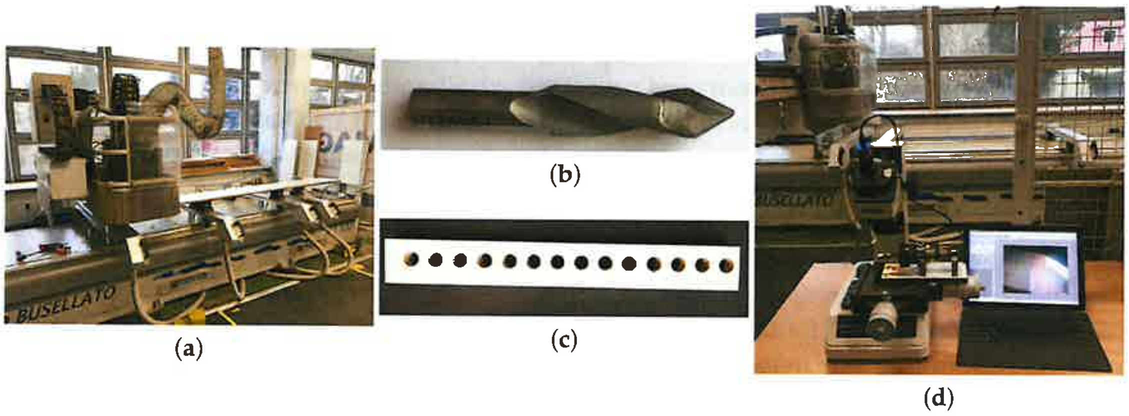


Figure 1. Equipment setup used during the experiments:Busellato Jet 100 machine centre (**a**), 12 mm Faba WP-01 drill (**b**), Kronopol U 511 SM melamine-faced chipboard sample (**c**) and workshop microscope TM–505, Mitutoyo (**d**).

The data set used in the current experiments was similar to that in the previous works [6,14–17], which was used both in the case of training the CNN network from scratch, as well as using transfer learning. It consisted of 5 image sets showing drilled holes, where each collection represented a separate tool. Samples were stored in the order in which they were made, showing the gradual wear of the drill and its effects on the hole edges. A total of 8526 images were taken, from which 3780 represented the green class, 2800 the yellow class, and 1946 the red class.

Usually, three classes are used in drill wear state recognition: red, green, and yellow. In this case, the obtained samples were divided and labelled manually, using the drill wear rate. For the manual evaluation of the drill state, external corner wear (W(mm)) was adopted as the main condition indicator and was periodically monitored using a standard workshop microscope (TM–505; Mitutoyo, Kawasaki, Japan). Based on the obtained values, three classes for drill wear were selected: green for $W < 0.2$ mm, yellow for W in the range between 0.2 and 0.35 mm, and red for $W > 0.35$ mm. Those classes were also used for the drill wear definition in the current, automated approach. In the presented case, the yellow class was used mainly as a buffer for the manufacturer. In the case of the furniture industry, depending on the type of elements produced, different hole qualities can be acceptable. In this case, depending on the manufacturer's preferences, the yellow class can later be assigned either to the green or red classes in the final production, hence expanding the overall method's customizability. Example images representing different drill wear classes are presented in Figure 2. Images presenting drills at different wear stages are presented on Figure 3. Because of that division, there were some examples when the drill was reaching the yellow or red state, and hence, they were hard to classify; therefore, it was expected that especially with the additional requirements, the accuracy rate would not reach the 90% threshold.

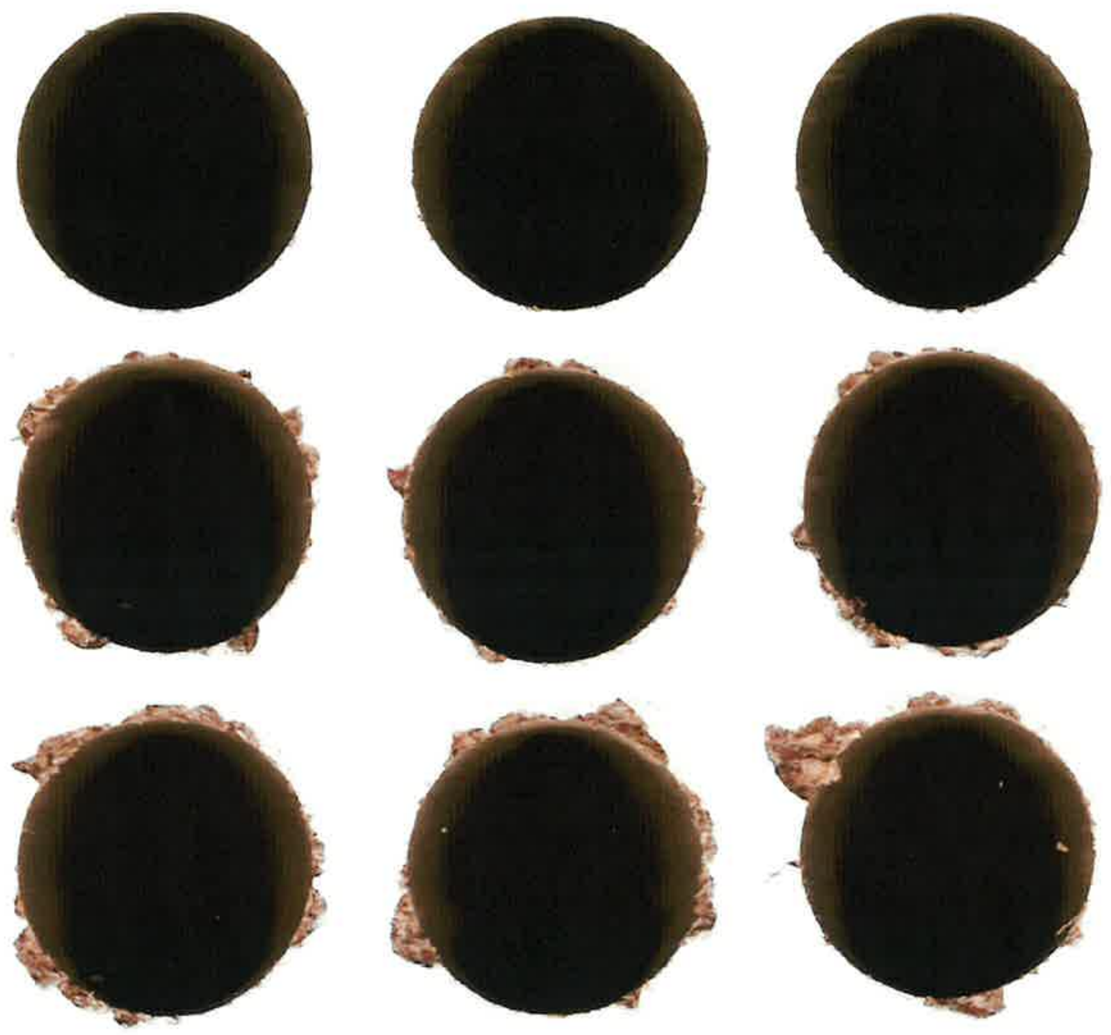
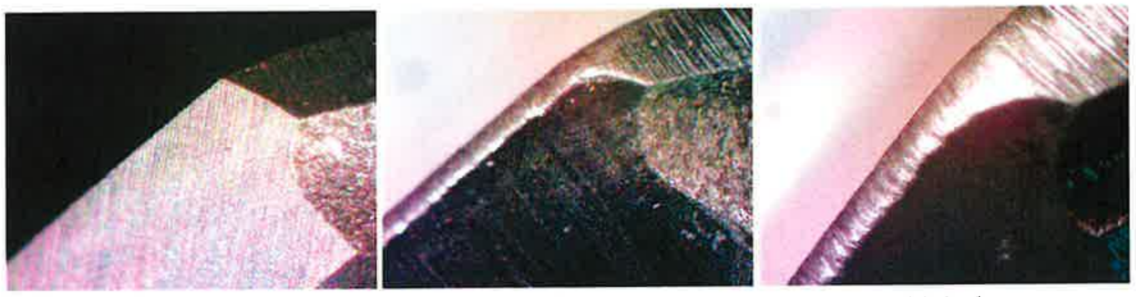


Figure 2. Example images representing holes made by drills with different wear classifications: green (**top**), yellow (**middle**), and red (**bottom**).



(**a**) Green (**b**) Yellow (**c**) Red

Figure 3. Images showing the wear of the outer drill corner for different classes: Green (**a**), Yellow (**b**) and Red (**c**). The microscope optical magnification equals 30×, while the total magnification for the monitor was 852.

3. Data Preprocessing

The original data set contained significantly more examples for the green class than the remaining two (yellow and red). Since CNN was used as a base network for the presented solution, it was not desirable for the data set to be imbalanced in such a way (it was important for the training process that each class be equally represented). To correct this, data augmentation methodologies were used, to ensure an even representation of each class (at this point, a simple rotation by 180 degrees was applied to the images, until the samples in each set reached a count equal to the best represented green class). Table 1 outlines the initial quantities for samples representing each class before and after data augmentation.

With the data balanced, the training process should not favour any of the classes. Initial operations performed on the data samples also included resizing each of the images on the fly to a size equal to 64 × 64 × 3 pixels. The training input was also normalized by dividing each value by 255, to ensure that they were in the (0, 1) range. Since 5-fold cross-validation was used, the input data were split between the 5 folds, and each of them was additionally divided into two subsets, the first for training and the second one for validation. The structure of each fold is shown in Figure 4.

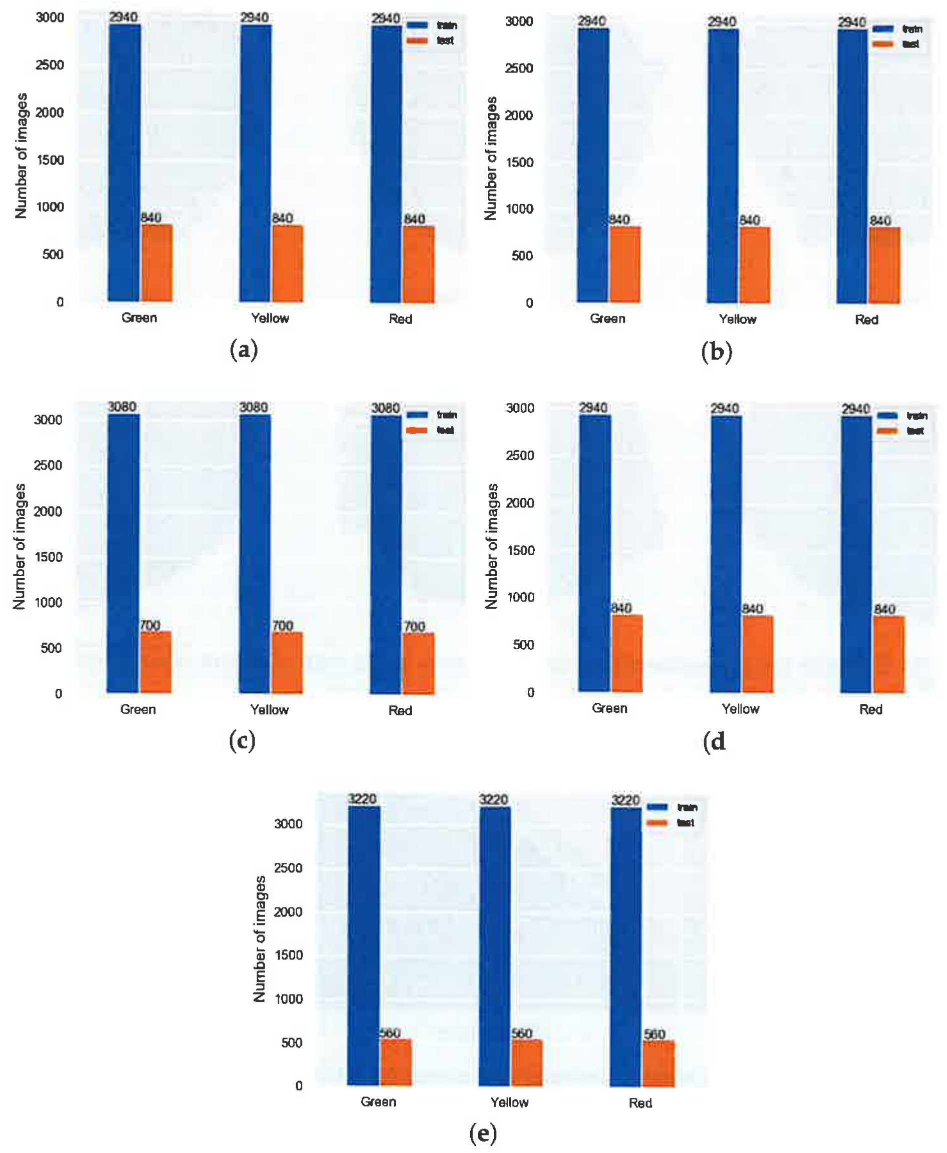


Figure 4. Fold data split during the training process using 5-fold cross-validation. The subsequent images represent the structure of: Fold 1 (**a**), Fold 2 (**b**), Fold 3 (**c**), Fold 4 (**d**), Fold 5 (**e**).

Table 1. Sample counts for each class before and after data augmentation. Values in each cell are presented in the following order: green, yellow, red.

Set Number	Original	Augmented
1	840/420/406	840/840/840
2	840/700/280	840/840/840
3	700/560/420	700/700/700
4	840/560/280	840/840/840
5	560/560/560	560/560/560

4. Siamese Network for Drill Wear Classification

4.1. Siamese Network Architecture

Siamese networks are novel algorithms used in image recognition. The first approaches with this type of procedure focused specifically on face recognition. One of the first applications was a verification system for identifying workers in a building. When it comes to this problem in general, there are two main areas to consider: verification of whether a person in the current image is one stored in a database under a specified ID and recognizing if the person from the input image is one of those stored in the original database. Especially in systems with large amounts of users (i.e., gates used to restrict access to certain building parts), accuracy is a very important factor. While having a 99% recognition rate might be acceptable for other applications, it is not the case here. Even if such a system has a 1% error rate, with 100 people in the database, the possibility of not recognizing the current person correctly is still quite high. Additionally, for most cases with face recognition, the algorithm needs to be able to recognize the person while using a single image (the one-shot learning problem). Using CNN for such an approach is not good enough, since firstly, the amount of training data is minimal, and secondly, each time a new person is added to the system, the network would require retraining. This is where the approach used in Siamese networks has the advantage.

Firstly, for the face verification problem, instead of learning to recognize each person separately, the network learns a similarity function (or the difference between the image in the database and the currently presented image). In that case, if the difference between two images is greater than the set threshold, the person would be classified as different, and as the same in case of this value being below that threshold. In general, it can be described as:

$$\begin{aligned} &\text{If } x^{(i)} = x^{(j)} \text{ than } \left\| f\left(x^{(i)}\right) - f\left(x^{(j)}\right) \right\|^2 \text{ is small} \\ &\text{If } x^{(i)} \neq x^{(j)} \text{ than } \left\| f\left(x^{(i)}\right) - f\left(x^{(j)}\right) \right\|^2 \text{ is large} \end{aligned} \tag{1}$$

where:

- $x^{(i)}$, $x^{(j)}$: images representing sample from the database and the sample that is checked for similarity.
- f($x^{(i)}$), f($x^{(j)}$): functions describing each input image for distance measuring.

To be able to calculate this distance between the input images, both of them are encoded using identical CNN networks. They are then represented as feature vectors instead of the usual classification. In general, what is done at this point is that instead of using the final classifier from the CNN (or other network), the entire process stops at one of the embedding layers (or features derived from the original image). By using this approach, two different, comparable encodings of the images can be obtained, and the distance between them can be measured (i.e., using a dense layer, with 128 parameters as a vector used to compare the two images). The idea of first launching two identical CNN networks to produce feature vectors and secondly using those vectors to calculate the difference measure between images is the basis of the Siamese network architecture [22].

4.2. *Contrastive Loss Function*

The Siamese network is a good example of a solution that can distinguish between instances of different classes and specifically determine if the image that is provided as the input is the same as the one representing the original class. In terms of face recognition, it would determine if the same person is in the picture. In the case of the solution presented here, with some additional modifications, it should point to which drill wear class the provided example belongs.

To train networks used for such recognition, a few steps are required, as well as a definition of the function used to distinguish between positive and negative examples of each class. The presented solution needs to be able to do two things: recognize the same class in two different images and notice that the presented class is different than the one to which it is compared. To achieve that, the following images are required: the first one, containing the element representing a single class (called the anchor and denoted as A), the positive image example, containing the same class (positive, denoted as P), and the negative image, with a different class (negative, denoted as N). When the distance between those images is calculated (from the obtained image representation), the ideal outcome would produce results for which the distance between the images containing the element with the same class is lower than in the case of the images containing elements representing different classes. What is more important for this approach to work accurately is that this distance needs to be significant, reaching at least some predefined margin (i.e., if the difference between A and P is smaller by only 0.01 with regard to distance between A and N, it might not be enough). The above relation can be described using the following equations:

$$\begin{gathered} \left\| f\left(x_i^a\right) - f\left(x_i^p\right) \right\|_2^2 + \alpha < \left\| f\left(x_i^a\right) - f\left(x_i^n\right) \right\|_2^2 \\ \forall \left(f\left(x_i^a\right), f\left(x_i^p\right), f\left(x_i^n\right) \right) \in \tau \end{gathered} \tag{2}$$

where:

- αis a margin that is enforced between positive and negative pairs
- τis the set of all possible triplets in the image set and has cardinality N.

At this point, the loss that needs to be minimized (generally described as the contrastive loss function [23]) has the form in Equation (3).

$$L = \sum_i^N \left[\left\| f\left(x_i^a\right) - f\left(x_i^p\right) \right\|_2^2 - \left\| f\left(x_i^a\right) - f\left(x_i^n\right) \right\|_2^2 + \alpha \right]_+ \tag{3}$$

The margin parameter (α) is a hyperparameter that needs to be manually adjusted to each classification problem. In the case of the topic described in this paper, one type of positive pair (with label y = 1) and two types of negative pairs were used in that process (the first type containing a yellow class example and the second the red class, both with label y = 0). Figure 5 contains examples of each type of pair used for training. The method used during that process is presented in Algorithm 1 and was introduced in [24].

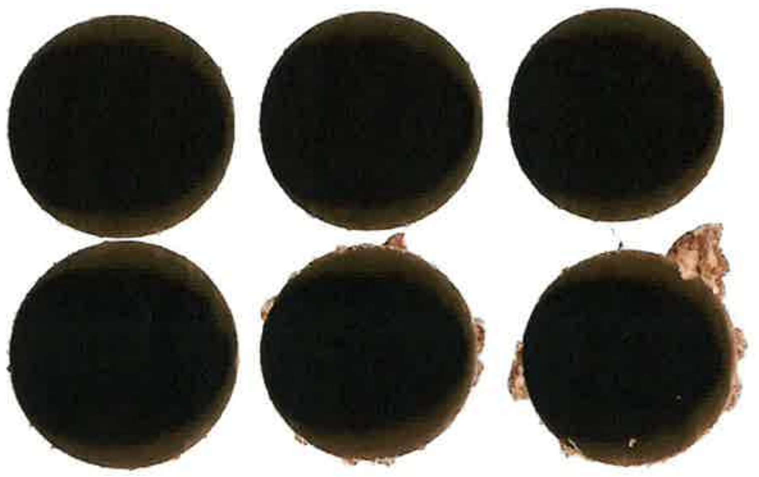

(a) (b) (c)

Figure 5. Examples of pairs used for training. The first pair - (a) - is a positive example, with label = 1; the following pairs are negative examples with label = 0, with green-yellow (b) and green-red (c) case. Anchors are presented at the top of each pair, while the checked image is at the bottom.

Algorithm 1: Siamese network training.

```
Step 1: Generating the training set
for each input sample $X_i$ do
  Using knowledge about the data set to find a set of samples such that each sample $X_j$ is similar to $X_i$.
  Pair sample $X_i$ with all the other training samples, and label pairs as $Y_{ij} = 0$ if the samples are similar, $Y_{ij} = 1$ otherwise
end for
Combine all the pairs to form the labelled training set.
Step 2: Training
while Convergence not reached do
  for each pair $(X_i, X_j)$ do
    if $Y_{ij} = 0$, update W to decrease $D_W = \left\|f(x_i) - f(x_j)\right\|_2^2$
    if $Y_{ij} = 1$, update W to increase $D_W = \left\|f(x_i) - f(x_j)\right\|_2^2$
  end for
end while
```

4.3. Siamese Network for Drill State Recognition

Since Siamese networks were originally for the face recognition problem, using a similar approach for the drill wear evaluation considered in this work, some adjustments were required. During the training process, first, the set of examples was created, where each example would contain two samples: anchor (A, to which the second sample will be compared, to determine if it is the same or different, corresponding to the picture of the person from the database in the face recognition problem) and either a positive (P) or a negative (N) example. In this case, instead of a single image that can either be the same or different (as with original application of this solution), a total of three classes were considered. First would be the positive example, with the same class as the anchor. In this case, two negative examples can be generated, containing either of the remaining classes.

The approach first divides the entire data set into positive and negative pairs. In the case of negative examples, to increase the diversity of the training set, the class is randomly chosen from the two that are different than the one to which the anchor belongs. Each of the initial images will generate two pairs used for training: one positive, where the anchor is paired with the image of the same class, and one negative, in which the anchor will be paired with an example from a different class. To calculate the distance between the images in consecutive samples, the contrastive loss function is used (see [24]).

As a base network for the learning process, CNN is used, with three convolutional layers. The detailed structure of the model prepared for the problem chosen as a main topic of this paper is outlined in Listing 1. The Siamese network uses two identical CNN networks to generate the parameters for each of the images. The general structure of such a network is presented in Figure 6. The full algorithm that was applied to drill wear recognition is presented in Algorithm 2.

Algorithm 2: Network training algorithm used for the drill wear recognition problem.

```
Create positive and negative pairs:
for each of initial D = 3 classes do
  for all images in each class do
    A = current image with index = i
    P = next image from the same class with index = i + 1
    PairPositive = (A, P) with label = 1
    Randomly choose one of the remaining classes (different than the current one)
    Choose negative image N with index = i
    PairNegative = (A, N) with label = 0
  end for
end for
Calculate the contrastive loss function:
for each created pair do
  Contrastive loss = $\frac{1}{m}\sum_{(i,j))}^{m/2} y_{i,j}D_{i,j}^2 + (1 - y_{i,j})\left[\alpha - D_{i,j}\right]_+^2$
end for
Return classification
```

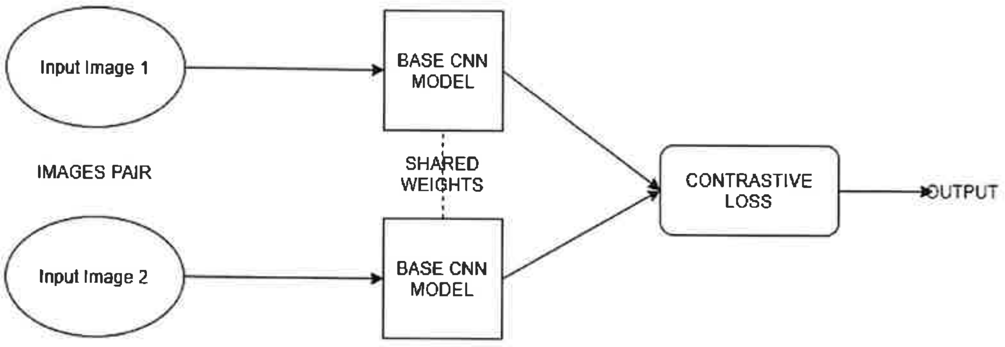


Figure 6. Outline of the Siamese network model used in the current experiments.

Listing 1: List of layers used in the CNN model.

```
Model: ''CNN''
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
conv2d_1 (Conv2D) (None, 62, 62, 64) 1792
_________________________________________________________________
activation_1 (Activation) (None, 62, 62, 64) 0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 64) 0
_________________________________________________________________
dropout_1 (Dropout) (None, 31, 31, 64) 0
_________________________________________________________________
conv2d_2 (Conv2D) (None, 29, 29, 64) 36,928
_________________________________________________________________
activation_2 (Activation) (None, 29, 29, 64) 0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 64) 0
_________________________________________________________________
dropout_2 (Dropout) (None, 14, 14, 64) 0
_________________________________________________________________
conv2d_3 (Conv2D) (None, 12, 12, 32) 18,464
_________________________________________________________________
activation_3 (Activation) (None, 12, 12, 32) 0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 32) 0
_________________________________________________________________
dropout_3 (Dropout) (None, 6, 6, 32) 0
_________________________________________________________________
flatten_1 (Flatten) (None, 1152) 0
_________________________________________________________________
dense_1 (Dense) (None, 128) 147,584
_________________________________________________________________
dropout_4 (Dropout) (None, 128) 0
_________________________________________________________________
dense_2 (Dense) (None, 50) 6450
=================================================================
Total params: 211,218
Trainable params: 211,218
Non trainable params: 0
_________________________________________________________________
```

5. Results and Discussion

During previous experiments [14–17], different approaches were tested and evaluated. While the obtained results were high in terms of overall accuracy, the entire solution still required improvement, when it comes to additional manufacturer requirements for the desired output. The main parameter concerned the number of errors between the red and green classes, which needed to be minimized. In this case, even a high overall accuracy would not be enough to compensate for this defect.

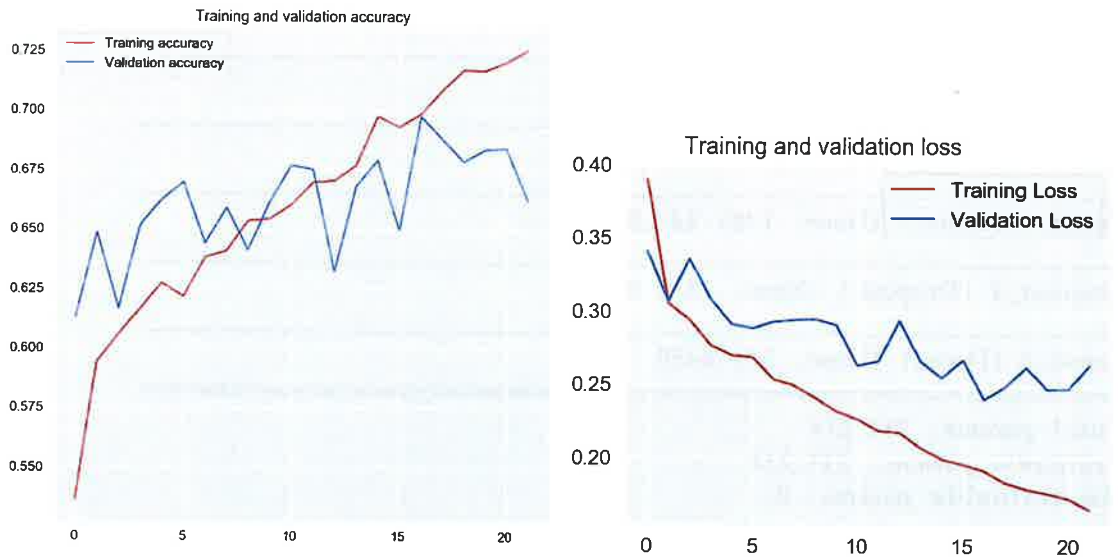
In the current approach, the Siamese network, adjusted to the presented classification problem, was used. To evaluate the obtained results, additional algorithms were implemented, choosing procedures that were successful in previous experiments, but similarly adjusting them to include the new requirement of minimizing the number of misclassifications between the red and green classes.

During the experiments, five-fold cross-validation was used (for the exact fold structure, see Figure 4). All experiments were performed using two NVIDIA TITAN RTX graphics card (with a total 48 GB of RAM), using the TensorFlow platform with the Python and Keras library, applying the Adam optimizer. Additionally, for the accuracy measurements, a custom function was used, defined as shown in Algorithm 3.

Algorithm 3: Accuracy function used for algorithm evaluation.

```
Require: (y_true, y_predicted)
  Compute classification accuracy with a fixed threshold on image distances
  Return K.mean((K.equal(y_true, K.cast(y_pred < 0.5, y_true.dtype)))
  model.compile(loss = contrastive_loss, optimizer = 'adam', metrics = [accuracy])
```

For each training sequence, one-hundred epochs were used. To optimize the entire solution in terms of the computational efficiency (which is an additional requirement related to the time needed before usable results are obtained; from the manufacturer's point of view, this time should be as short as possible), an early stopping mechanism was used. The patience parameter was used, set at five epochs, meaning that if during that time, there was no improvement to the solution accuracy, the training process was stopped, and the best obtained model was saved. Figure 7 shows an example of the training process in terms of the accuracy and loss functions.



(**a**) Training and validation accuracy (**b**) Training and validation loss

Figure 7. Example function changes for training and validation accuracy (**a**) and loss (**b**).

Since the data used during the experiments were stored in a time series manner (meaning that for each of the tools used, the images of drilled holes were stored exactly in the order they were made), it was additionally incorporated in the overall approach, to try and improve the algorithm's accuracy. Instead of a single image, sets of consecutive images of different lengths were used for the training process. With such an approach, the algorithm should be able to learn how hole edges change, while the drill is steadily dulling. The window parameter was incorporated into the solution, testing sequences of 5, 10, 15, and 20 images, with no window approach used as the baseline (experts from the furnishing company proposed such window sizes, since they can be incorporated into the production process and used in an actual factory). The first solution, which did not use any window, obtained an overall accuracy of 67% (while the green class recognition had the highest individual scores; see Table 2) and produced a significant number of errors for green-red and red-green misclassifications. Due to the manufacturer requirements, this approach was not satisfactory.

Table 2. Results for the Siamese network without using a window.

	Precision	Recall	f1-Score
Green	0.81	0.79	0.80
Yellow	0.53	0.53	0.53
Red	0.67	0.68	0.68
Accuracy			0.67

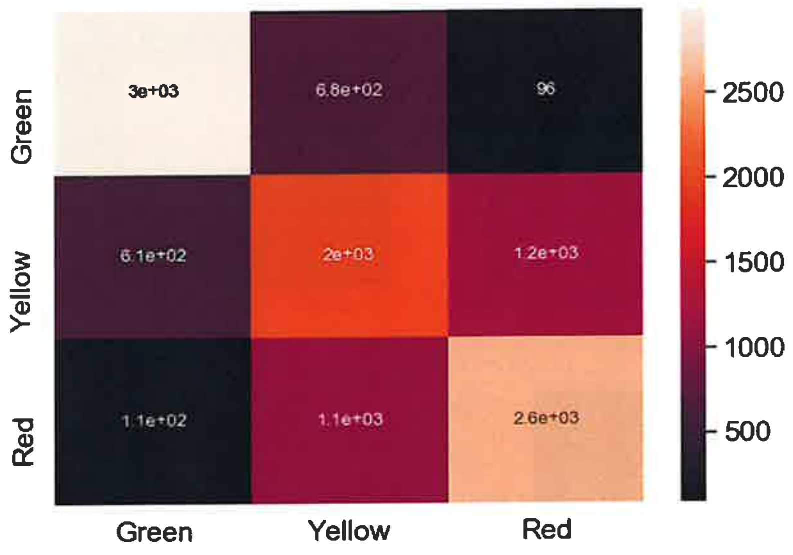


Figure 8. Confusion matrix for the Siamese model with no window used.

To establish a reasonable benchmark for the current algorithms' accuracy, different solutions were implemented and tested, with the manufacturer's requirements in mind. While choosing the algorithms, the main goal was to select as diverse a set as possible, both in terms of the base used (i.e., different pretrained networks and a custom network trained from scratch), as well as the classification methodology (single network or a set of randomly initialized classifiers using the bagging methodology). Since in previous experiments, some of the algorithms were tested and performed well for the overall accuracy parameter, the same algorithms were used for the current comparison. The final set contained the VGG19 pretrained network, 2 ensemble algorithms, the first using 5 and the second using 10 random VGG16 networks, the CNN model trained from scratch, and 5 random initializations of this model. All of those algorithms were trained using the window methodology, starting with no window and finishing at a window size of 20. The accuracy results obtained are presented in Table 3, where each row list the algorithm name.

Next, the Siamese network approach using different windows was tested, and it achieved the best accuracy results for a window size of 20 (82%). Although for smaller windows, it showed poorer results than some algorithms, it quickly outperformed them for larger windows. While the no window approach produced to many critical errors (see Figure. 8), Figure 9 clearly indicates that increasing the window size also resulted in a better classification rate for the red and green examples, with only 22 red samples classified as green and 15 green samples classified as red. With such a decrease in the misclassification rate, the presented solution reached the benchmark acceptable from the manufacturer's point of view.

Table 3. Classification results for the chosen algorithms, for classification with no window and with different window sizes (5, 10, 15, and 20) for consecutive samples.

Algorithm	No window	W = 5	W = 10	W = 15	W = 20
VGG19	67%	73%	76%	77%	78%
5xVGG16	67%	74%	77%	78%	79%
10xVGG16	67%	73%	76%	77%	78%
CNN-designed	70%	75%	78%	79%	80%
5xCNN-designed	67%	73%	77%	79%	80%
Siamese	67%	74%	78%	80%	82%

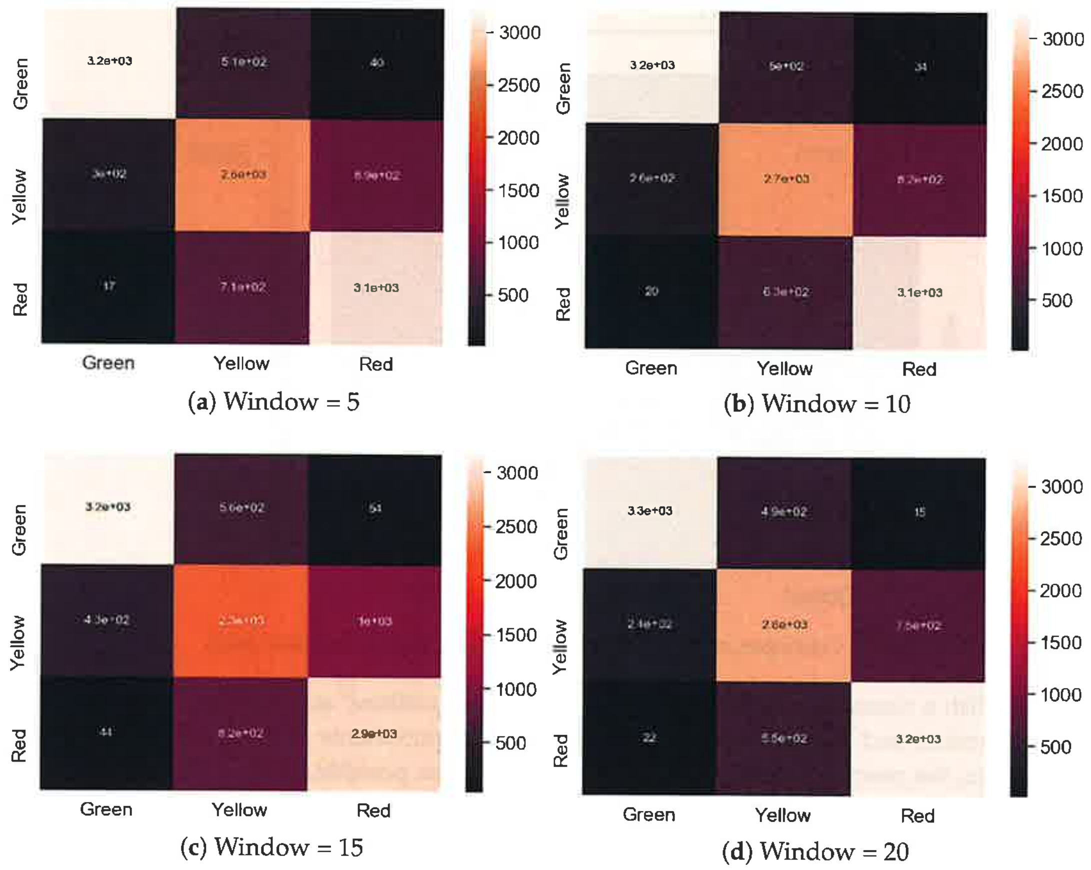


Figure 9. Confusion matrix for the Siamese model with different windows.

For future work, additional modifications of the presented solution will be considered, focusing on further decreasing the misclassification rate, especially for smaller windows. Furthermore, since storing the data samples in time series, such as in this case, is not always possible, finding some alternative to the windows used in the current approach might also be advisable.

6. Conclusions

In this work, the application of a Siamese network to the drill wear recognition problem is presented. Three classes are used for the evaluation: red, green, and yellow. The presented solution uses a custom CNN model and is trained on a considerable data set. Furthermore, the presented solution is evaluated in terms of accuracy, comparing it to different algorithms, implemented with the same manufacturer requirements in mind: reducing the number of misclassifications between the red and green classes. Additionally, since the input images are stored in the exact order in which they were made, the window parameter is introduced, to further increase the classification accuracy. The final solution is able to outperform all tested algorithms in terms of overall accuracy (82% for a window size of 20). Additionally, it is able to accurately distinguish between the red and green classes, with a total number of 37 misclassifications between them (22 red-green and 15 green-red errors). To the best of authors' knowledge, this is the first application of this methodology to the wood industry. The presented approach is highly adjustable, since in the case of changes in the samples (such as the material used or slight changes to the tools or methods used), transfer learning can be used to retrain the previous model for a new application, without the need to start from scratch.

To summarize, the presented solution achieved an overall accuracy and misclassification rate that fit into the initial acceptable ranges. With the simplified data collection methodology and low initial costs, it is readily applicable to the actual work environment, with very positive, initial feedback from the manufacturer. Furthermore, the Siamese network approach seems very promising, and while further research is still required, it is believed that additional improvement of both the accuracy and critical error rate is still possible.

Author Contributions: Conceptualization, J.K., I.A., A.J., B.Ś. and M.B.; methodology: J.K., B.Ś., I.A., M.B. and A.J.; resources, A.J.; writing—original draft preparation, I.A.; writing—review and editing, I.A.; visualization, B.Ś., M.B.; supervision, J.K.; project administration, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mikołajczyk, T.; Nowicki, K.; Bustillo, A.; Pimenov, D.Y. Predicting tool life in turning operations using neural networks and image processing. *Mech. Syst. Signal Process.* **2018**, *104*, 503–513. [CrossRef]
2. Merenda, M.; Laurendi, D.; Iero, D.; D'Addona, D.M.; Della Corte, F.G. Internet of Things Platform for Real-Time Intraoral Forces Monitoring. *Procedia CIRP* **2020**, *88*, 570–573. [CrossRef]
3. Mazo, C.; Alegre, E.; Trujillo, M. Using an ontology of the human cardiovascular system to improve the classification of histological images. *Sci. Rep.* **2020**, *10*, 1–14. [CrossRef] [PubMed]
4. Jemielniak, K. Commercial tool condition monitoring systems. *Int. J. Adv. Technol.* **1999**, *15*, 711–721. [CrossRef]
5. Górski, J.; Szymanowski, K.; Podziewski, P.; Śmietańska, K.; Czarniak, P.; Cyrankowski, M. Use of cutting force and vibro-acoustic signals in tool wear monitoring based on multiple regression technique for compreg milling. *Bioresources* **2019**, *14*, 3379–3388.
6. Kurek, J.; Kruk, M.; Osowski, S.; Hoser, P.; Wieczorek, G.; Jegorowa, A.; Górski, J.; Wilkowski, J.; Śmietańska, K.; Kossakowska, J. Developing automatic recognition system ofdrill wear in standard laminated chipboard drilling process. *Bull. Pol. Acad. Sci. Tech. Sci.* **2016**, *64*, 633–640.
7. Jemielniak, K.; Urbański, T.; Kossakowska, J.; Bombiński, S. Tool condition monitoring based on numerous signal features. *Int. J. Adv. Manuf. Technol.* **2012**, *59*, 73–81. [CrossRef]
8. Panda, S.; Singh, A.; Chakraborty, D.; Pal, S. Drill wear monitoring using back propagation neural network. *J. Mater. Process. Technol.* **2006**, *172*, 283–290. [CrossRef]
9. Kuo, R. Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. *Eng. Appl. Artif. Intell.* **2000**, *13*, 249–261. [CrossRef]
10. Jegorowa, A.; Górski, J.; Kurek, J.; Kruk, M. Use of nearest neighbors (k-NN) algorithm in tool condition identification in the case of drilling in melamine faced particleboard. *Maderas. Cienc. Tecnol.* **2020**, *22*, 189–196. [CrossRef]
11. Deng, L.; Yu, D. Deep learning: Methods and applications. *Found. Trends R Signal Process.* **2014**, *7*, 197–387. [CrossRef]

12. Bengio, Y. Learning deep architectures for AI. *Found. Trends R Signal Process.* **2009**, *2*, 1–127.
13. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
14. Kurek, J.; Wieczorek, G.; Kruk, M.; Swiderski, B.; Jegorowa, A.; Osowski, S. Transfer learning in recognition of drill wear using convolutional neural network. In Proceedings of the 18th International Conference on Computational Problems of Electrical Engineering (CPEE) 2017, Kutna Hora, Czech Republic, 11–13 September 2017; pp. 1–4.
15. Kurek, J.; Swiderski, B.; Jegorowa, A.; Kruk, M.; Osowski, S. Deep learning in assessment of drill condition on the basis of images of drilled holes. In Proceedings of the Eighth International Conference on Graphic and Image Processing (ICGIP 2016), Tokyo, Japan, 29–31 October 2016; International Society for Optics and Photonics: Bellingham WA 98227-0010 USA 2017; Volume 10225, pp. 1–8.
16. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Swiderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J. Data Augmentation Techniques for Transfer Learning Improvement in Drill Wear Classification Using Convolutional Neural Network. *Mach. Graph. Vis.* **2019**, *28*, 1–8.
17. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Swiderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J. Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network. *Mach. Graph. Vis.* **2019**, *28*, 1–8.
18. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; pp. 1–800.
19. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
20. Krizhevsky, A.; Sutskever, I.; Hinton, G. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
21. Shelhamer, E. BVLC AlexNet Model. Available online: https://github.com/BVLC/caffe/tree/master/models/bvlc_alexnet (accessed on 12.08.2020).
22. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* **2014**, 3, 1701–1708.
23. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
24. Hadsell, R.; Chopra, S.; LeCun, Y. Dimensionality reduction by learning an invariant mapping. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Tokyo, Japan, 29–31 October 2016; Volume 2, pp. 1735–1742.

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978. https://doi.org/10.3390/s20236978
mój indywidualny udział w jej powstaniu polegał na:

1. Koordynacja działań związanych z tworzeniem koncepcji.
2. Metodologii.
3. Nadzorze.
4. Administracji.
5. Walidacji.

[illegible]
Podpis

Warszawa, 27-01-2025

Izabella Antoniuk
izabella_antoniuk@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978. https://doi.org/10.3390/s20236978
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Przygotowaniu pierwszego szkicu.
4. Edycji i korekcji artykułu.

Izabella Antoniuk
Podpis

Warszawa, 27-01-2025

Bartosz Świderski
bartosz_swiderski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978. https://doi.org/10.3390/s20236978
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Wizualizacji.

Bartosz Świderski

Podpis

Warszawa, 27-01-2025

Albina Jegorowa
albina_jegorowa@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978. https://doi.org/10.3390/s20236978
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Akwizycji danych.
4. Zarządzaniu danymi.

A. Jegorowa

Podpis

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. Sensors 2020, 20, 6978. https://doi.org/10.3390/s20236978
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.

[illegible]
Podpis

10.2. Publikacja 2

PUBLIKACJA 2

Decision Confidence Assessment in Multi-Class Classification

Michał Bukowski, Jarosław Kurek, Izabella Antoniuk, Albina Jegorowa

Sensors 2021, 21, 3834

https://doi.org/10.3390/s21113834

Punktacja: 100, IF: 3,847

Communication

Decision Confidence Assessment in Multi-Class Classification

Michał Bukowski [1], Jarosław Kurek [1,*], Izabella Antoniuk [1] and Albina Jegorowa [2]

[1] Institute of Information Technology, Warsaw University of Life Sciences, Nowoursynowska 159, 02-776 Warsaw, Poland; michal.bukowski@buksoft.pl (M.B.); izabella_antoniuk@sggw.edu.pl (I.A.)
[2] Institute of Wood Sciences and Furniture, Warsaw University of Life Sciences, Nowoursynowska 159, 02-776 Warsaw, Poland; albina_jegorowa@sggw.edu.pl
* Correspondence: jaroslaw_kurek@sggw.edu.pl; Tel.: +48-505-482-708

Abstract: This paper presents a novel approach to the assessment of decision confidence when multi-class recognition is concerned. When many classification problems are considered, while eliminating human interaction with the system might be one goal, it is not the only possible option—lessening the workload of human experts can also bring huge improvement to the production process. The presented approach focuses on providing a tool that will significantly decrease the amount of work that the human expert needs to conduct while evaluating different samples. Instead of hard classification, which assigns a single label to each class, the described solution focuses on evaluating each case in terms of decision confidence—checking how sure the classifier is in the case of the currently processed example, and deciding if the final classification should be performed, or if the sample should instead be manually evaluated by a human expert. The method can be easily adjusted to any number of classes. It can also focus either on the classification accuracy or coverage of the used dataset, depending on user preferences. Different confidence functions are evaluated in that aspect. The results obtained during experiments meet the initial criteria, providing an acceptable quality for the final solution.

Keywords: confidence classification; confidence functions; multi-class classification; tool condition monitoring; laminated chipboard

check for updates

Citation: Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. *Sensors* **2021**, *21*, 3834. https://doi.org/10.3390/s21113834

Academic Editors: Panagiotis E. Pintelas, Sotiris Kotsiantis and Ioannis E. Livieris

Received: 31 March 2021
Accepted: 19 May 2021
Published: 1 June 2021

1. Introduction

Ensuring an efficient and smooth flow of production processes can be challenging, time-consuming, and, at times, also problematic. For example, in the wood industry, from the many tasks that need to be monitored, some of them will require specialized knowledge and precision, while others will use up a significant amount of time, and there are quite a lot of activities that combine all of those features. One of such tasks concerns evaluating the state of drills in the manufacturing process, which is a subset of problems widely known as tool condition monitoring. Usually, when manually performed, this task requires stopping the production process in order to evaluate individual tools. At the same time, a human expert is required to check used elements, without any indication to its actual state. Due to that, unnecessary downtime may occur, when it could have been avoided if the entire process had been, at least partially, automated.

When it comes to tool evaluation, many different approaches have been considered to either speed up the process, or avoid human intervention in general. For example, the main focus may include evaluating the state of elements without interrupting the actual manufacturing process, as presented in [1]. Most basic and commonly used approaches, such as the one presented in [2], measure different signals, such as vibration, noise, acoustic emission, cutting torque, feed force, and others, in order to evaluate the tool state. Similar approaches were used in [3], where data were extracted both from signal and frequency domains, along with wavelet coefficients, all in order to evaluate the obtained elements automatically, checking how relevant each item was to the selected problem. Further, in [4], the authors used different signals, over a wide range of cutting conditions, using a back-propagation neural network to predict the flank wear of a drill bit. Another approach which relies heavily on different sensors is presented in [5]. In this case, various sensors were used to collect data, which were later integrated using a fuzzy logic model.

While sensor-based approaches are quite widely used, they are not always the best solutions. First of all, the equipment required to even start taking different measurements tends to be quite expensive and would also require lengthy calibration, which, if conducted incorrectly, can affect the resulting accuracy. Furthermore, a setup which contains multiple sensors might be difficult to integrate into an actual work environment without affecting the production process. With all those problems, as well as some additional requirements that might appear in different industries, regarding the desired accuracy or some additional properties of the final solution, such as limiting the number of critical errors (which corresponds to any mistakes between border classes of tool wear), a simpler input might be required.

In some previous works, images were used as a base for drill state evaluation, using various machine learning algorithms. Such solution was considered in [6,7]. In those cases, the specialized sensors were dropped entirely, and instead of signals, images of drilled holes were used for evaluation purposes, requiring only a simple camera to obtain them. The presented solutions are based mainly on convolutional neural networks (CNNs, which have the additional advantage of not requiring any specialized, diagnostic features; those networks are also considered top solutions in the case of image recognition, as mentioned in [8]). The first of the two approaches uses the data augmentation technique to achieve dataset expansion without the need for additional samples, combined with a transferred learning methodology. Accuracy of 93% was achieved here, without the need for a complicated sensor setup. In the case of the second solution, a classifier ensemble was used to further increase the overall classification rate, exceeding a 95% accuracy rate. There are also some recent approaches that incorporate similar methodologies. In [9], various CNN networks were tested and evaluated to prepare an improved approach that focused more on limiting critical errors that the classifier makes. In another solution, presented in [10], a Siamese network was applied to the same problem, which is a new, CNN-based methodology. In both approaches, the window parameter, which included consecutive images of holes drilled in sequence, was used to further increase the achieved results. Finally, in [11], a more time-efficient approach was presented, this time using image color distribution, with an assumption that, after converting the image to gray-scale, there will be more pixels with mid-range values within images representing holes drilled by more used tools. All those solutions achieve high accuracy results and a relatively low amount of critical errors.

What can be noted is that while most of the presented solutions take into account the manufacturer requirements, they also have some drawbacks. First of all, in the case of more difficult examples, the solutions tend to make different errors in the final classification (some more severe than others). Second of all, the manufacturer cannot easily switch between different metrics used to evaluate the final solution. Those drawbacks led to the current approach, in which images are still used as input data, but instead of using a hard classification model, which assigns classes to each presented example, a more elastic approach is incorporated. Instead of classifying each sample as belonging to one of the classes (green for a tool which is new, red for a tool that should be discarded, and yellow for one that requires further evaluation), a confidence metric is incorporated to inform the user how exact the current classification is. Samples can then either be further classified or discarded and assigned to the human expert for that purpose. Furthermore, the solution can be adjusted to focus either on accuracy or the dataset coverage—since different industries might have varying requirements in regard to those aspects, such approach provides the user with more control over how the presented solution works. It will also allow for easier adaptation to chosen problems.

The novelty of this work is that it provides a robust way to quantify the uncertainty of any multi-class classification into a confidence parameter that allows us to discriminate some observations with low confidence in order to increase performance metrics for the rest of the observations. This approach allows easily combining human expert knowledge and algorithm ways of classification and can be added on top of the multi-class classifier.

2. Methodology

In previous work (see [11]), it was noted that while using a set of images converted to gray-scale, the amount of pixels placed in the mid ranges can be used to evaluate the state of the drill that was used to prepare the hole shown on each image. The research presented in this article continues on this assumption, but instead of hard classification, where each image is assigned a single class, the samples are evaluated in terms of decision confidence.

The presented confidence evaluation process consists of a few steps, involving initial data processing, model preparation, and the confidence function itself.

2.1. Dataset

The dataset used in the current experiments contained a total of 8526 images showing holes drilled by steadily declining tools. For the initial class evaluations, the resulting sample set was manually labeled. In the case of the presented dataset, external corner wear—W (mm) was used as a decisive factor for assigning a class to each image. This parameter was measured using a workshop microscope, TM-505 Multitoyo, Kawasaki, Japan. According to experts in this field, the parameter ranges were established as follows:

- $W < 0.2$ mm—drill classified as green;
- $0.2 \text{ mm} < W < 0.35$ mm—drill classified as yellow;
- $0.35 \text{ mm} < W$—drill classified as red.

From those images, 3780 were classified as the green class, 2800 samples were classified as the yellow class, and 1946 represented the red class. Images were chosen as input data, since, while showing the declining state of the drill (edges tend to be more jagged in the case of more used tools than in the case of new ones), they do not require a significant amount of time to obtain, and the acquisition process itself can be adjusted to the specific needs of each manufacturer. All samples used in the current research were obtained in cooperation with the Institute of Wood Sciences and Furniture at Warsaw University of Life Sciences. The summary of the dataset is presented in Table 1 below.

Table 1. Dataset structure.

Class	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Total
Green	840	840	700	840	560	3780
Yellow	420	700	560	560	560	2800
Red	406	280	420	280	560	1946
Total	1666	1820	1680	1680	1680	8526

Holes were drilled with a standard CNC vertical machining center, Busellato Jet 100 (Busellato, Thiene, Italy). To ensure that the entire setup is as close to the potential manufacturer requirements as possible, the drilled material was a material typically used in the furniture industry, laminated chipboard U511SM Swiss Krono Group (Swiss Krono Sp. z o.o., Żary, Poland). The drill used for this application was a 12 mm Faba WP01 (Faba SA, Baboszewo, Poland) with a tungsten carbide tip. Initial test piece dimensions were 2500 mm × 300 mm × 18 mm. To acquire the actual images, they were later divided into smaller ones, which were separately photographed. Example fragments representing each of the recognized classes are presented in Figure 1. Final images, representing one hole each, were obtained using a custom script, which extracted the desired area and saved it in separate images with three RGB color channels. The images were stored in the exact order in which they were made, facilitating easier evaluation of the obtained results, but the time series structure of the obtained data was not incorporated into the current solution. Example images showing the input images used by the presented procedure are shown in Figure 2.

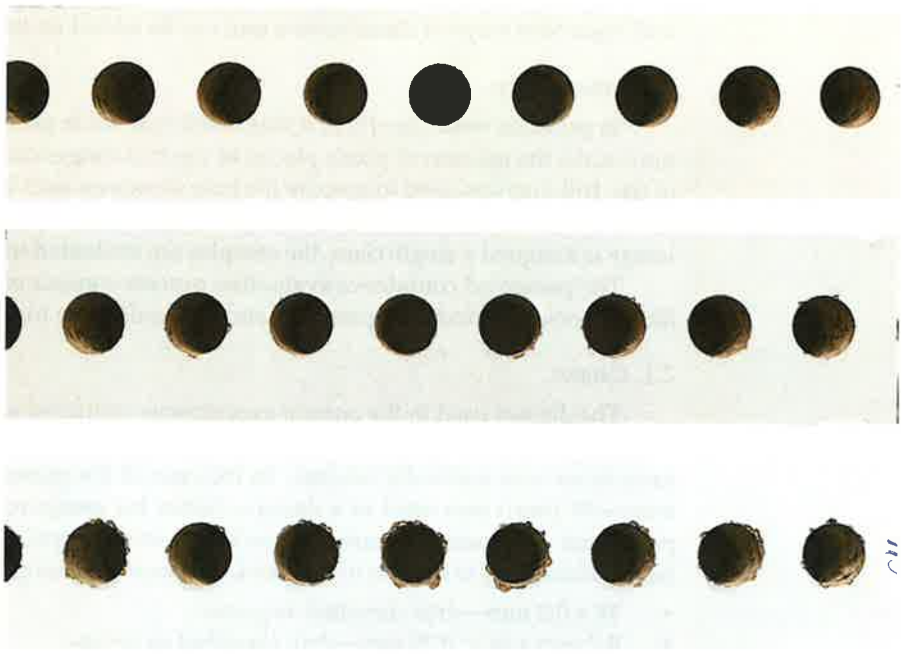


Figure 1. Images showing photographed samples after division into smaller elements, representing holes made by drills with different drill wear classifications: green (**top**), yellow (**middle**), and red (**bottom**).

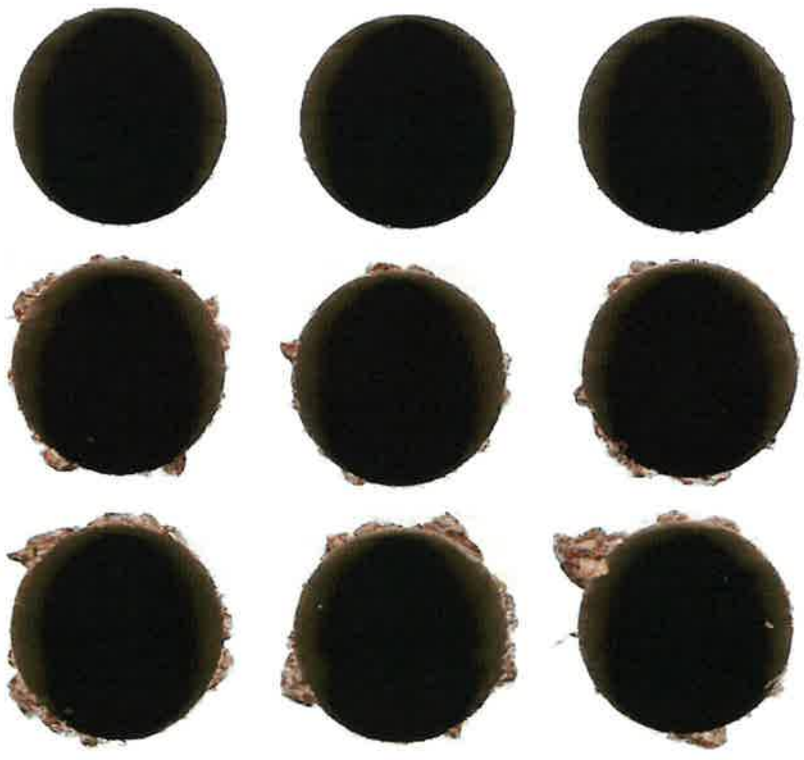


Figure 2. Images obtained after initial processing which are used as an input to the main procedure: green (**top**), yellow (**middle**), and red (**bottom**).

All the images in the current dataset were manually labeled as one of recognized classes by a human expert and later used by the prepared model.

2.2. *Model*

Images converted to gray-scale using ITU-R 601-2 luma transform specification were the input data for the following algorithm steps. During model preparation, the initial classification based on the overall grouping of pixels was conducted. The initial research presented in [11] showed that, although the images with holes of degrading quality show a steady increase in gray pixels (the pixels in each image were divided into three groups for that process—black for the hole, white for the laminated chipboard surface, and gray for the hole edge), there is no clear border between each class; hence, the images cannot be easily classified using only that count. At the same time, the general relation between the number of gray pixels and the quality of the drilled hole still remained, with images of holes belonging to the green class having significantly less gray pixels than those that belonged to the yellow or red classes.

During this initial step, the image preparation was conducted. Original images were represented in RGB and varied in size (the custom script prepared for this phase focused on cutting out the fragment of the image containing the hole, with the edges, and with as little detail from the surrounding sample as possible, but including any jagged parts of the hole edge). Due to that variation, in the first step, images were resized to 256 $\times$ 256 pixels to make sure that they have a uniform size. Additionally, since the information regarding the state of the hole edge does not require color values, the images were converted to gray-scale.

The next step involved counting occurrences of each pixel value and normalizing them to fit in the 0–1 range. This was accomplished by a simple operation of dividing each pixel count by the total number of pixels in the image. From those counts, an array containing 256 pixel values was prepared, which was later used as an input to the next part of the model. This element used Light Gradient Boosting Machine (also LGBM or Light GBM, described in [12]). LGBM uses tree-based learning, which grows trees vertically, with the maximal delta loss for leaves, and can handle larger datasets while using less memory. This approach also focuses on accuracy and has an efficiency parameter used as one of the main quality indicators. During experiments, 15 rounds of Bayesian optimization were used to choose and optimize hyperparameters with a multi-log loss metric. Data obtained from this element were later used in the following steps of the presented procedure.

In order to obtain probabilities with a window size of 1 (meaning that only a single image is taken into account, and not a sequence of images), the 5-fold cross-validation method was used. The presented method can also be easily expanded to include larger windows. Meanwhile, the baseline accuracy for the chosen set of parameters was 0.67 (these results were obtained in previous work [11], where no feature selection approach with a window equal to 1 achieved exactly the same result). Given the probabilities' distribution, achieved by 5-fold cross-validation, we calculate the confidences for each of the 4 different confidence functions defined in Section 3. For each of those 4 results, we calculate different metrics of how well they achieved their goal of measuring confidence, which are also defined in Section 3. We compare the results of those metrics in Section 4.

2.3. *Problem Formulation*

The main metrics considered with the current classification model were accuracy and the number of severe (red–green) errors. In order to increase both of them, several approaches can be adapted, including using a set of subsequent images instead of single ones, as conducted in previous authors' work [9] using window parameters. After that, the dominant value of all classifications can be used. This method, although it increases the classification rate in general, is only suitable for some problems, especially since it complicates the deployment of the classifier into the manufacturing process (the industry needs to be able to produce subsequent images during production).

In the current approach, instead of hard classification, where an image is assigned to one of the recognized classes, an additional class or state was added, making it possible to return an "I don't know" or an "undefined" pseudo-class, where some observations will not be classified at all. Later on, those samples labeled as "undefined" can be further examined by human experts. While this approach is not fully automatic, it can eliminate the majority of observations with a clear classification, leaving only harder and more interesting examples for manual evaluation, possibly resulting in better performance of the entire solution. The folds are the same as in Table 1; therefore, this process is based on 5-fold cross-validation. The overall structure of the presented solution is shown in Listing 1, where classification and calculating confidence are presented, and in Listing 2, where different confidence metrics are calculated.

Listing 1. Classifying observations, and calculating confidence.

```

#Initialize model

model = LgbmDrillClassifier(size=256x256,greyscale=True, window_size=1,
                            hyperparameters_rounds = 15)

#Calculate predictions using 5-fold cross validation

folds = [f1,f2,f3,f4,f5]
predicted_results = []
true_results = []

for (training_folds, test_fold) in enumerate_folds(folds):
    model.fit(training_folds)
    predictions = model.predict_trial(test_fold)
    predicted_results.append(predictions)
    true_results.extend(test_fold.true_y)

#Calculate confidence

confidences = {}
confidence_function_names = ['Shannon', 'l1=l_inf', 'l2', 'Gini']

for confidence_name in confidence_function_names:
    confidences[confidence_name] = calculate_confidence(confidence_name,
                                                         predicted_results,
                                                         true_results)
```

Listing 2. Calculating confidence metrics.

```

#Calculate metrics that can be used to compare confidence functions

# Calculate accuracy threshold

accuracy_threshold_results = {}

for confidence_name in confidence_function_names:
    for accuracy_threshold in [0.7, 0.75, 0.80, 0.85, 0.9]:
        result = calculate_accuracy_threshold(true_results,
                                              predicted_results,
                                              confidences[confidence_name],
                                              accuracy_threshold)
        accuracy_threshold_results[confidence_name] = result

# Calculate coverage threshold

coverage_threshold_results = {}

for confidence_name in confidence_function_names:
    for coverage_threshold in [0.95, 0.90, 0.80, 0.70, 0.60]:
        result = calculate_coverage_threshold(true_results,
                                           predicted_results,
                                           confidences[confidence_name],
                                           coverage_threshold)
        coverage_threshold_results[confidence_name] = result

# Calculate weighted accuracy gain

weighted_accuracy_gain_results = {}
for confidence_name in confidence_function_names:
    result = calculate_weighted_accuracy_gain(true_results,
                                           predicted_results,
                                           confidences[name])
    weighted_accuracy_gain_results[confidence_name] = result

# Calculate area under curve

auc_results = {}
for confidence_name in confidence_function_names:
    result = calculate_auc(true_results,
                          predicted_results,
                          confidences[name])
    auc_results[confidence_name] = result
```

3. Confidence Function

The approach presented in this paper is based on a confidence function, which will describe how sure used the model is of the presented classification. The results which have low confidence can be discarded, hence leaving only those with higher values in that aspect. By using such filtered samples, it is believed that better results can be obtained when compared with using the entire dataset with an unsure classification. An ideal situation will, for example, drop 2% of the least confident results, boosting the actual accuracy by around 10%.

3.1. *Confidence Function Constraints*

In order to use some methods as confidence functions, a set of constraints needs to be defined first. Assume that a result's probability vector of a multi-classification problem with a number of classes of $n \geq 3$ is given with probabilities obtained using the softmax function (which is a common practice in neural networks and other models [13]).

To achieve comparable results for different confidence functions, the function should be able to transform an n element vector of probabilities into a single value in the range [0.0, 1.0].

The confidence function $Conf$ should satisfy three constraints:

$$Conf(v_{eq}) = 0 \tag{1}$$

$$Conf(v_{unit}) = 1 \tag{2}$$

$$0 < Conf(v) < 1 \quad \forall v \in V_n \setminus \{v_{eq}, v_{unit}\} \tag{3}$$

where v_{eq} is the probability vector containing equal probabilities, v_{unit} is the probability vector with all but one element equal to 0, and V_n is the set of all probability vectors with length n.

Given some confidence threshold t, all observations that have a confidence score lower than t are categorized with the "undefined" pseudo-class. The coverage c is the fraction of all observations for some threshold t that are still normally classified, and we will denote the accuracy of that classification as a. Increasing the threshold should increase accuracy, but it will decrease coverage.

3.2. *Confidence Function Candidates*

While the constraints and some general assumptions for the presented model were defined, it still requires some candidates for the confidence function to be pointed out. For the initial approach, a total of three methods were considered: Shannon, Gini, and Norm confidence, which are outlined below.

3.2.1. Shannon-Based Confidence

Shannon entropy is a good candidate for confidence, as it is used as an inequality measure [14]. It can be defined as presented in Equation (4).

$$Conf_{Shannon}(v_n) = 1.0 - H_n(v_n) \tag{4}$$

where v_n is an n dimensional probability vector, and H_n is the entropy with a logarithm base equal to n. It satisfies all constraints because the maximal possible entropy achieved with v_{eq} is 1, and for $H_n(v_{unit})$, it is 0.

3.2.2. Gini-Based Confidence

Another measure of inequality [15] is the Gini coefficient. In order to satisfy the chosen constraints, they should be normalized by the Gini coefficient of the unitary vector v_{unit}. Gini confidence can then be defined as shown in Equation (5).

$$Conf_{Gini}(v_n) = Gini(v_n)/Gini(v_{unit}) \tag{5}$$

As $Gini(v_{eq})$ is 0, and maximal inequality is achieved by v_{unit}, this confidence measure also satisfies all constraints.

3.2.3. Norm-Based Confidence

To use a slightly different approach, the inequality of a given prediction can also be measured as the distance to the closest unitary vector. In order to satisfy the constraints, the distance needs to be adjusted, considering the maximal possible distance, which is the distance from v_{unit} to v_{eq}. To choose the distance metric, standard l_1, l_2, or l_{inf} norms can be used, as shown in Equation (6).

$$Conf_{norm}(v_n) = max_{0 \leq i \leq n}(1 - \frac{norm(v_n - [0, \ldots 1_i, \ldots, 0]_n)}{norm(v_{unit} - v_{eq})}) \tag{6}$$

The maximum norm l_{inf} is the baseline comparison of all confidence functions (see Equation (7)). It is the simplest one of the presented functions, and it just measures the distance between the maximum probability and 1.0, standardizing it to be between [0.0, 1.0] so it can fulfill the constraint set.

It is worth noting that with that scaling, l_{inf}-based confidence gives the same results as l_1-based confidence; therefore, a baseline value can be obtained using either of those functions.

$$Conf_{l_1} = Conf_{l_{inf}} \tag{7}$$

3.3. *Comparing Confidence Functions*

To select the best confidence function, which will be one that maximizes the accuracy a and coverage c for all thresholds t, a direct comparison of the chosen confidence functions is required for given model outputs. One additional factor that also needs to be included is the actual gain the confidence function gives for the current approach. This is achieved by comparing the confidence threshold accuracy with the default approach which does not use any confidence at all (default accuracy for the presented model).

3.3.1. Accuracy Threshold

Since the presented approach aims at being as versatile as possible, it is worth noting that depending on the specific application, the accuracy constraints might differ, requiring the solution to achieve specific values in that aspect. For that approach, the best confidence function would be one that, for the chosen accuracy threshold a, will achieve a confidence threshold t that it ensures the best coverage c.

In the presented case, the default accuracy of the used classifier is 0.67, and the goal accuracy is 0.80. The threshold t for Shannon-based confidence is 0.33, which corresponds to 0.80 accuracy a and 0.55 coverage c. For l_2-based confidence, the threshold t will be 0.41 with 0.46 coverage c. Therefore, in that problem, Shannon-based confidence is a better confidence function than l_2-based confidence.

3.3.2. Coverage Threshold

Another approach to this problem would consider, instead, that the confidence function should maximize the accuracy a for observations above the confidence threshold t corresponding to a given coverage threshold c. In general, this would correspond to eliminating the most problematic $(1 - c)$ fraction of cases from the set, and maximizing the accuracy on the rest of the observations.

For example, in the presented model, let us assume a 0.9 coverage threshold is our goal. With Gini-based confidence, the corresponding confidence threshold t would be 0.42 with accuracy a equal to 0.69. With the same requirements, for l_1-based confidence, the confidence threshold t would be 0.02 with accuracy a of 0.67.

3.3.3. Weighted Accuracy Gain

Weighted accuracy gain measures the weighted sum of the difference between different threshold accuracies and default classifier accuracies, as shown in Equation (8).

$$Wag = \frac{\sum_{t=0.0}^{1.0}(a_t - a_0) \times c_t}{n} \tag{8}$$

where a_t is accuracy with a confidence threshold t, a_0 is accuracy with a confidence threshold of 0, which corresponds to the baseline classifier accuracy, c_t is coverage with a confidence threshold t, and n is the number of thresholds considered.

3.3.4. Confidence Area under Curve

In the case when no accuracy threshold is given, several confidence thresholds can be checked instead, and the function maximizing both accuracy and coverage should be chosen. This problem is similar in formulation to the receiver operating characteristic area under the curve—roc_auc [16]—and we will be using the auc shortcut in the next sections.

The area under the confidence curve (as shown in Figure 3) can be calculated using the trapezoid rule. The used points were constructed in pairs (a, c) that correspond, respectively, to accuracy and coverage for each of the confidence thresholds t, from 0 to 1.0. It was assumed that an accuracy with confidence of 1.0 is also one of the thresholds with the lowest coverage. The baseline for this is also the accuracy result of the initial model, which does not include any type of confidence.

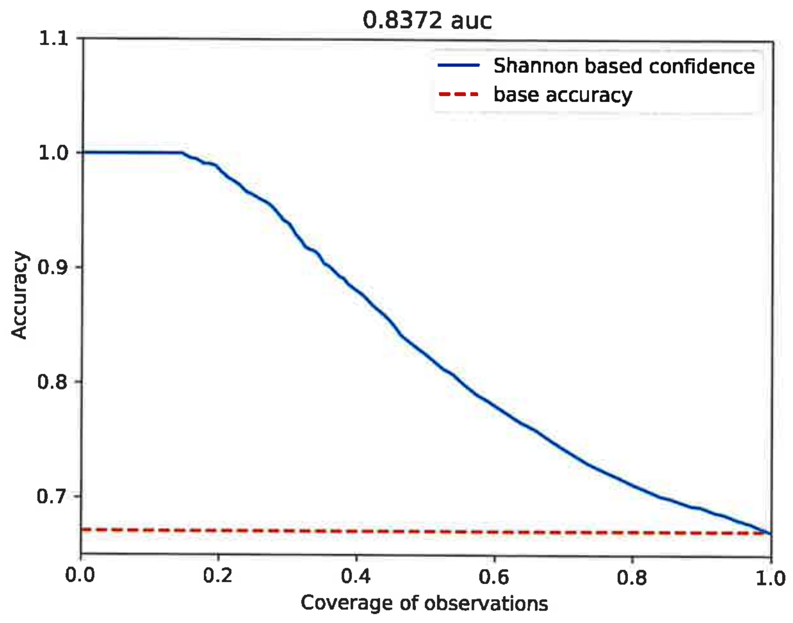


Figure 3. Plot presenting confidence curve for Shannon-based confidence, where it achieves a 0.8372 auc score.

4. Results and Discussion

The process of calculating results for different confidence functions is shown in Listing 2 and is based on the dataset presented in Table 1 using the 5-fold cross-validation presented in Listing 1.

For each confidence function, different accuracy and confidence threshold functions were used. There is an expected trade-off between accuracy gain and coverage drop. For example, the Shannon-based confidence accuracy change is presented in Figure 4a, represented by the blue line. As it can be seen, it comes with a drop in the coverage, as the same colored line representing this function in Figure 4b is the lowest one. All norm-based confidences behave similarly in that aspect. In the case of the general accuracy–coverage, Shannon-based confidence and Gini-based confidence seem to represent opposite trade-off values; therefore, their usage should be adjusted accordingly.

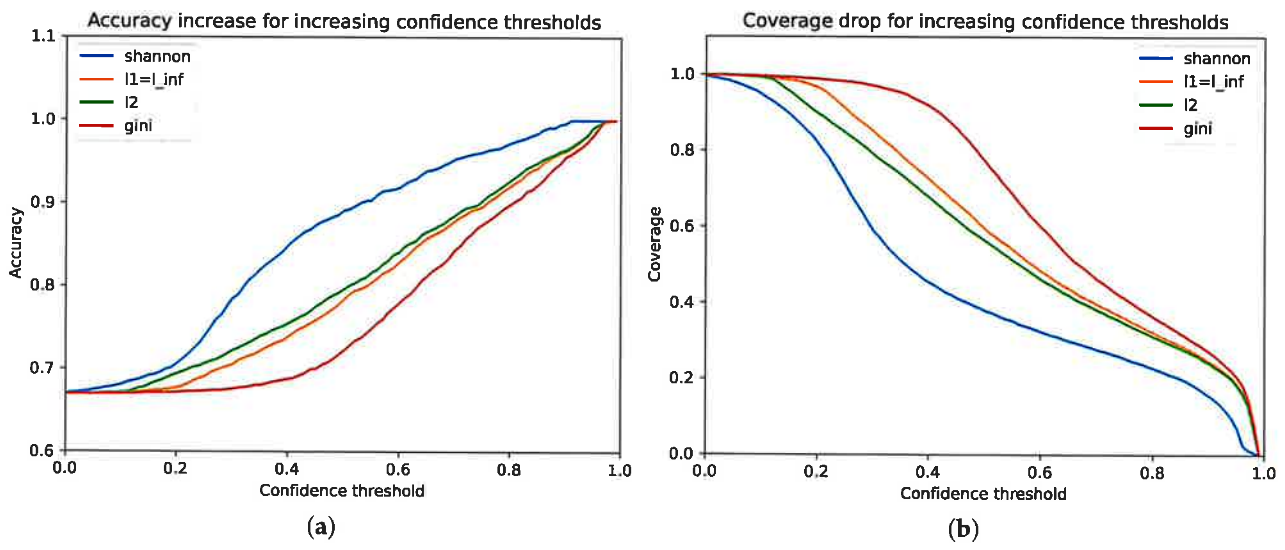


Figure 4. Accuracy (**a**) and coverage (**b**) curves for evaluated functions.

4.1. Comparison of Accuracy Threshold

The main goal of the presented research is to increase the accuracy and keep high coverage c. The first method of comparison focuses on the coverage of observations that is kept given some accuracy threshold. The best results for different accuracy thresholds a were obtained by different functions, with Gini-based confidence being the only exception to that (see Table 2). The used baseline l_{inf} was the best in two of five cases. While high accuracy can be obtained, the methods that actually achieve it also need to sacrifice a significant portion of coverage c. The highest value of coverage for the 90% threshold was achieved for Shannon-based confidence but covered only 36% of the used dataset. In comparison, the accuracy rate of 70% was able to cover over 87% of the data for the $l_1 = l_{inf}$ function. It is also worth noting that different functions performed best for different accuracy thresholds; therefore, some additional improvement can be achieved by using the best performing function for the current setup.

Table 2. Comparison of dataset coverage for different accuracy thresholds a_t for each evaluated confidence function.

Confidence Function	$a_t = 0.70$	$a_t = 0.75$	$a_t = 0.80$	$a_t = 0.85$	$a_t = 0.90$
Shannon	0.8403	0.6586	0.5389	0.4464	0.3611
$l_1 = l_{inf}$	0.8770	0.6929	0.5463	0.4429	0.3500
l_2	0.8692	0.6832	0.5493	0.4504	0.3456
Gini	0.8586	0.6661	0.5385	0.4424	0.3554

4.2. Comparison of Coverage Threshold

The problem presented in this paper can also be approached from the coverage threshold point of view. If the goal of classification, instead of achieving a specific accuracy, is more focused on including as many samples as possible (with only a specified fraction of examples being evaluated by a human expert), the coverage threshold approach should be used. The achieved results for different coverage parameters are presented in Table 3. In this case, the baseline l_{inf} metric is the best two out of five times. The best results are obtained for the l_2 norm, which wins three out of five times; however, differences for each coverage threshold c are often negligible.

Table 3. Comparison of accuracy obtained by the evaluated confidence functions for different coverage thresholds c_t.

Confidence Function	$c_t = 0.95$	$c_t = 0.90$	$c_t = 0.80$	$c_t = 0.70$	$c_t = 0.60$
Shannon	0.6799	0.6895	0.7063	0.7400	0.7756
$l_1 = l_{inf}$	0.6820	0.6945	0.7175	0.7457	0.7815
l_2	0.6807	0.6940	0.7182	0.7486	0.7823
Gini	0.6810	0.6895	0.7123	0.7426	0.7799

4.3. Comparison of Weighted Accuracy Gain

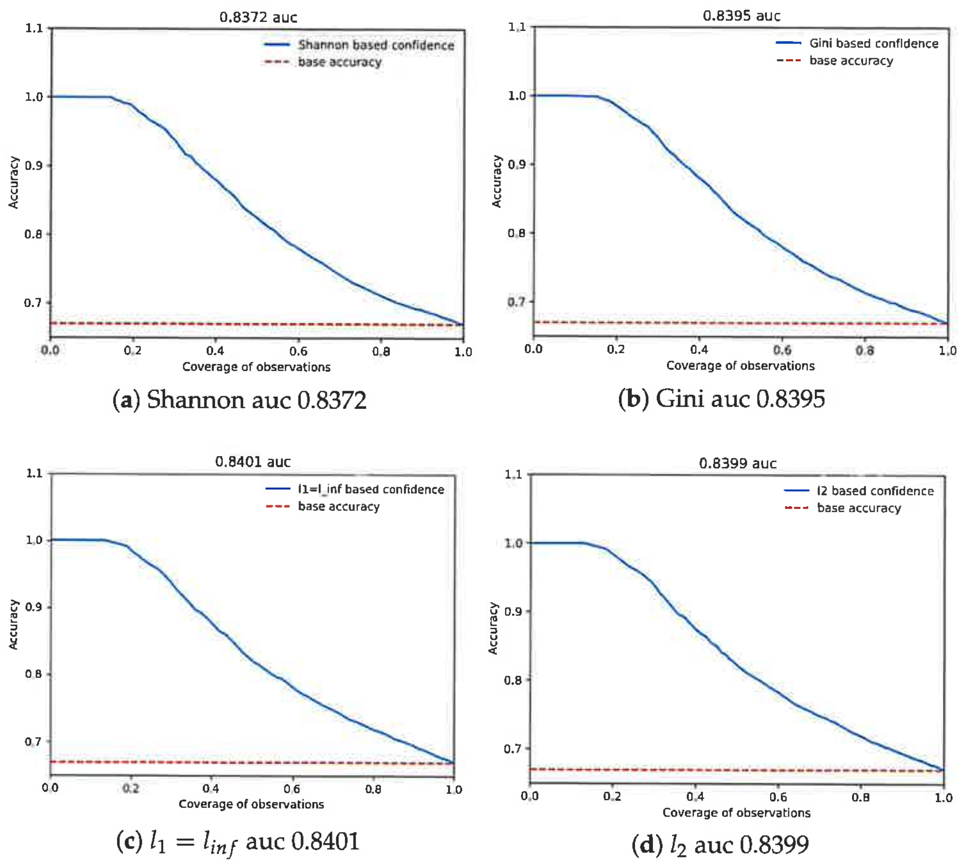
For this metric, the baseline l_{inf} performs significantly worse than the other metrics. The best result is achieved by Shannon-based confidence, and second place is taken by l_2. Gini-based confidence again is not the best solution. Results for the weighted accuracy gain comparison are presented in Table 4.

Table 4. Comparison of weighted accuracy gain with different confidence functions.

Confidence Function	Weighted Accuracy Gain
Shannon	0.0550
$l_1 = l_{inf}$	0.0491
l_2	0.0525
Gini	0.0387

4.4. Comparison of Area under Confidence Curve

In the case of the area under the confidence curve comparison, the obtained results are very indecisive, presenting similar qualities for all used functions. As it can be seen in Figure 5, all curves look very similar, and differences between all norms are negligible. The baseline confidence norm l_{inf} has the highest auc result of 0.401 (see Table 5), but differences between the functions are not significant. Further investigation is needed to check if this is the case for all models.



(**a**) Shannon auc 0.8372 (**b**) Gini auc 0.8395

(**c**) $l_1 = l_{inf}$ auc 0.8401 (**d**) l_2 auc 0.8399

Figure 5. Confidence threshold curves comparison.

Table 5. Comparison of threshold area under the curve with different confidence functions.

Confidence Function	Area under Curve
Shannon	0.8372
$l_1 = l_{inf}$	0.8401
l_2	0.8399
Gini	0.8395

4.5. Error Rate Evaluation

When it comes to the relation between accuracy and the used coverage threshold, one additional parameter needs to be evaluated. When it comes to overall requirements in different industries, usually mistakes between border classes are the most costly ones (green and red in the case of drill wear classification, which was presented in this paper). Such errors should be avoided in general, and the coverage parameter presented in this paper was introduced in order to discard as many unsure examples (for later classification by a human expert) as possible, in order to increase the classification accuracy and reduce the number of severe errors. In order to evaluate the impact of coverage on the number of critical errors that the classifier makes, the Shannon-based confidence from Figure 3 will be used.

Figure 6 shows confusion matrixes for three coverage thresholds, starting with the classification representing the full dataset, and finishing with coverage of 30%. As it can be seen, while the coverage decreases, the number of critical errors (red–green or green–red) between border classes diminishes at the same time. For the initial, baseline setup, with full dataset coverage (which can also be denoted as an approach that does not use confidence at all), the overall number of critical errors equaled 159, with 80 cases of the red class classified as green and 59 green samples classified as red. This number decreases to a total of 45 critical errors when the dataset coverage decreases to 60% (29 red–green and 16 green–red misclassifications). The Shannon-based confidence function mostly removes observations with the yellow class. With this coverage threshold, we keep 50% of red observations, 39% of yellow observations, and 78% of green observations. The number of critical errors is even lower for the 30% coverage threshold, with only four critical errors (three red–green and one green–red). This threshold removes the yellow observations almost completely.

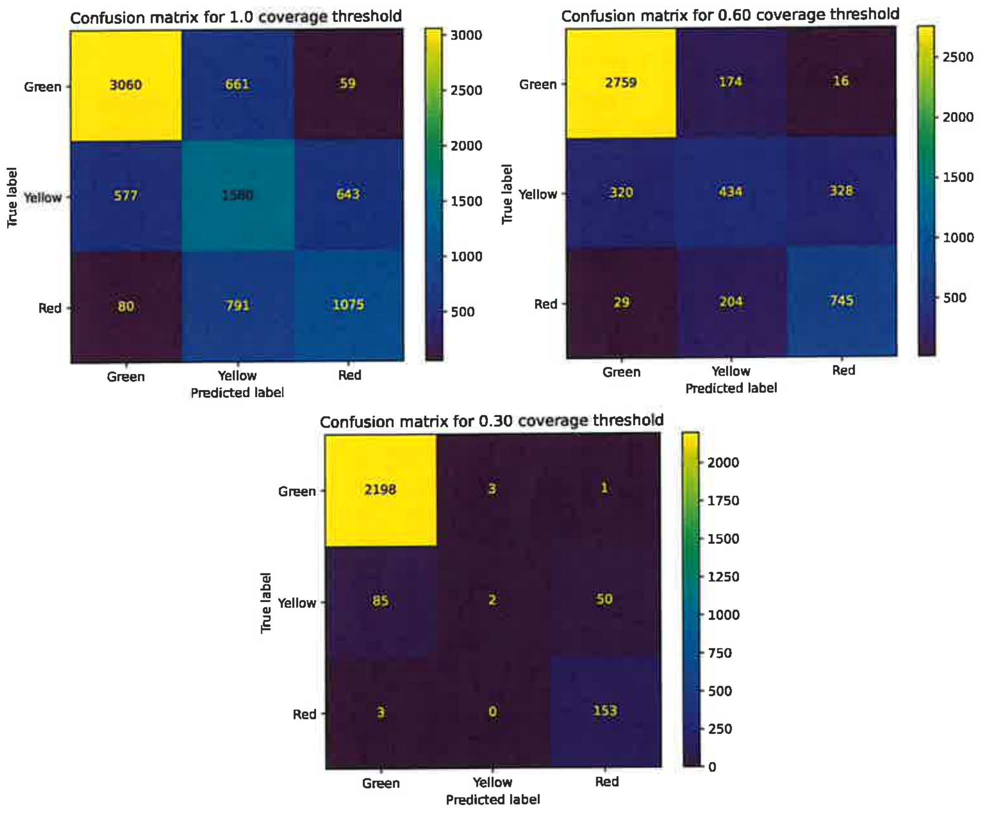


Figure 6. Confusion matrix obtained for different coverage thresholds: full dataset coverage and a baseline value for the critical error rate (**top left**), 60% coverage (**top right**), and 30% coverage (**bottom**).

4.6. Discussion

The problem which confidence calculation is trying to solve can be commonly seen in our world. The machine learning aid in diagnosing cancer can be a great aid if the expected accuracy of the used model can be controlled. The coverage threshold can be adjusted for human resources in different companies, thus allowing easy scaling of human–computer hybrid systems. It also aligns with the trend of explainable artificial intelligence [17], where the confidence of each prediction is a very good parameter outlining the model transparency.

For the four different confidence functions and the four different metrics, there is no clear winner. The used baseline l_{inf}-based confidence performed well in three out of four metrics and thus is a good, robust confidence function. However, if the focus is put on a particular metric, Shannon-based confidence and l_2-based confidence can be better solutions, depending on the chosen parameters. The Gini coefficient, which is often used as an inequality metric, performed poor as a confidence function, which is a surprising result.

There are other approaches that are similar to the confidence-based approach presented in this work. Bayesian networks, as presented in [18], can also be used to discriminate some observations into the "undefined" class. The Bayesian approach requires changing the underlying parameters into a parameter distribution, and sampling results from a posterior distribution. This, however, requires deep modification in the underlying model and is considerably slower due to the need for sampling from several different distributions.

Another approach that can also incorporate the "undefined" class is fuzzy classification [19]. Observations with low membership values with each class can be labeled as "undefined". There is also some work that defines confidence in fuzzy classification terms [20]. The caveat in this approach is that it requires entirely different problem formulations and datasets and thus cannot be easily combined with prior models.

5. Conclusions

In this article, a novel and adaptable classification algorithm was described. While the presented solution was mainly applied to drill wear classification, it is not limited to this task. Instead of focusing on hard classification, transferring to assigning a single class to each example, the method focuses more on evaluating confidence for each considered sample. While there is no clear winner in the evaluated confidence function set, the performance of at least one function for each accuracy or coverage threshold was at least acceptable. For some cases, the differences between different functions were negligible, while their performance was still satisfactory.

Even in its current state, the presented solution is quite versatile and can be easily adapted to any number of recognized classes. Furthermore, due to the confidence metrics applied, the model can be better evaluated in that aspect, pointing out how sure the classifier is when assigning a certain class to each sample. As an additional feature, depending on the manufacturer requirements, the method can focus more on obtaining the required accuracy rate, or covering a chosen fraction of samples. Finally, by discarding some of the examples, labeling them as too complicated for automatic classification (for the cases when the metric will show a confidence below the assigned threshold), and evaluating them by manual experts, the accuracy rate of the entire solution should perform better, avoiding severe errors which tend to be a problem with fully automatic solutions. All of the above features expand the available applications of the prepared algorithm. Additionally, when combined with a possible focus on either accuracy or dataset coverage, the overall functionality makes the presented solution more suited for various classification tasks that may appear in the wood and other industries.

Author Contributions: Conceptualization, M.B., J.K., I.A. and A.J.; methodology: M.B., J.K., I.A. and A.J.; resources, A.J.; writing—original draft preparation, I.A.; writing—review and editing, I.A.; visualization, M.B.; supervision, J.K.; project administration, J.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to privacy restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Górski, J.; Szymanowski, K.; Podziewski, P.; Śmietańska, K.; Czarniak, P.; Cyrankowski, M. Use of cutting force and vibro-acoustic signals in tool wear monitoring based on multiple regression technique for compreg milling. *BioResources* **2019**, *14*, 3379–3388.
2. Kurek, J.; Kruk, M.; Osowski, S.; Hoser, P.; Wieczorek, G.; Jegorowa, A.; Górski, J.; Wilkowski, J.; Śmietańska, K.; Kossakowska, J. Developing automatic recognition system ofdrill wear in standard laminated chipboard drilling process. *Bull. Pol. Acad. Sci. Technol. Sci.* **2016**, *64*, 633–640.
3. Jemielniak, K.; Urbański, T.; Kossakowska, J.; Bombiński, S. Tool condition monitoring based on numerous signal features. *Int. J. Adv. Manuf. Technol.* **2012**, *59*, 73–81. [CrossRef]
4. Panda, S.; Singh, A.; Chakraborty, D.; Pal, S. Drill wear monitoring using back propagation neural network. *J. Mater. Process. Technol.* **2006**, *172*, 283–290. [CrossRef]
5. Kuo, R. Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. *Eng. Appl. Artif. Intell.* **2000**, *13*, 249–261. [CrossRef]
6. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Świderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J . Data Augmentation Techniques for Transfer Learning Improvement in Drill Wear Classification Using Convolutional Neural Network. *Mach. Graph. Vis.* **2019**, *28*, 3–12.
7. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Świderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J. Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network. *Mach. Graph. Vis.* **2019**, *28*, 13–23.
8. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; pp. 1–800.
9. Jegorowa, A.; Kurek, J.; Antoniuk, I.; Dołowa, W.; Bukowski, M.; Czarniak, P. Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. *Wood Sci. Technol.* **2021**, *55*, 1–23. [CrossRef]
10. Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. *Sensors* **2020**, *20*, 6978. [CrossRef] [PubMed]
11. Jegorowa, A.; Antoniuk, I.; Kurek, J.; Bukowski, M.; Dołowa, W.; Czarniak, P. Time-efficient Approach to Drill Condition Monitoring Based on Images of Holes Drilled in Melamine Faced Chipboard. *BioResources* **2020**, *15*, 9611–9624.
12. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3146–3154.
13. Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv* **2018**, arXiv:1811.03378.
14. Mishra, S.; Ayyub, B.M. Shannon Entropy for Quantifying Uncertainty and Risk in Economic Disparity. *Risk Anal.* **2019**, *39*, 2160–2181. [CrossRef] [PubMed]
15. Bendel, R.B.; Higgins, S.S.; Teberg, J.E.; Pyke, D.A. Comparison of skewness coefficient, coefficient of variation, and Gini coefficient as inequality measures within populations. *Oecologia* **1989**, *78*, 394–400. [CrossRef] [PubMed]
16. Bewick, V.; Cheek, L.; Ball, J. Statistics review 13: Receiver operating characteristic curves. *Crit. Care* **2004**, *8*, 508. [CrossRef] [PubMed]
17. Dosilovic, F.K.; Brcic, M.; Hlupic, N. Explainable artificial intelligence: A survey. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 0210–0215. [CrossRef]
18. Chopra, P. Making Your Neural Network Say "I Don't Know"—Bayesian NNs Using Pyro and PyTorch. 27 November 2018. Available online: https://towardsdatascience.com/making-your-neural-network-say-i-dont-know-bayesian-nns-using-pyro-and-pytorch-b1c24e6ab8cd (accessed on 9 February 2021).
19. Kuncheva, L.I. *Fuzzy Classifier Design*; Springer: Heidelberg, Germany, 2000.
20. Nakashima, T.; Ghosh, A. Classification Confidence of Fuzzy Rule-Based Classifiers. In Proceedings of the 25th European Conference on Modelling and Simulation (ECMS), Krakow, Poland, 7–10 June 2011; pp. 466–471.

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834. https://doi.org/10.3390/s21113834
mój indywidualny udział w jej powstaniu polegał na:

1. Koncepcji.
2. Metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.

[illegible]
Podpis

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834. https://doi.org/10.3390/s21113834
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Nadzorze.
4. Administracji.
5. Walidacji.

[illegible]

Podpis

Warszawa, 27-01-2025

Izabella Antoniuk
izabella_antoniuk@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834. https://doi.org/10.3390/s21113834
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Przygotowaniu pierwszego szkicu.
4. Edycji i korekcji artykułu.

Izabella Antoniuk
Podpis

Warszawa, 27-01-2025

Albina Jegorowa
albina_jegorowa@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. Sensors 2021, 21, 3834. https://doi.org/10.3390/s21113834
mój indywidualny udział w jej powstaniu polegał na:

1. Współpracy w zakresie koncepcji.
2. Współpracy w zakresie metodologii.
3. Akwizycji danych.
4. Zarządzaniu danymi.

A. Jegorowa
Podpis

10.3. Publikacja 3

PUBLIKACJA 3

Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification

Michał Bukowski, Izabella Antoniuk, Jarosław Kurek

BULLETIN OF THE POLISH ACADEMY OF SCIENCES TECHNICAL SCIENCES, Vol. 71(6), 2023, Article number: e147338

https://doi.org/10.24425/bpasts.2023.147338

Punktacja: 100, IF: 1,2

BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 71(6), 2023, Article number: e147338
DOI: 10.24425/bpasts.2023.147338

ARTIFICIAL AND COMPUTATIONAL INTELLIGENCE

Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification

Michał BUKOWSKI, Izabella ANTONIUK, and Jarosław KUREK*

Department of Artificial Intelligence, Institute of Information Technology, Warsaw University of Life Sciences, Nowoursynowska 159, Warsaw, 02-776, Poland

Abstract. In this paper, we present an improved efficient capsule network (CN) model for the classification of the Kuzushiji-MNIST and Kuzushiji-49 benchmark datasets. CNs are a promising approach in the field of deep learning, offering advantages such as robustness, better generalization, and a simpler network structure compared to traditional convolutional neural networks (CNNs). Proposed model, based on the Efficient CapsNet architecture, incorporates the self-attention routing mechanism, resulting in improved efficiency and reduced parameter count. The experiments conducted on the Kuzushiji-MNIST and Kuzushiji-49 datasets demonstrate that the model achieves competitive performance, ranking within the top ten solutions for both benchmarks. Despite using significantly fewer parameters compared to higher-rated competitors, presented model achieves comparable accuracy, with overall differences of only 0.91% and 1.97% for the Kuzushiji-MNIST and Kuzushiji-49 datasets, respectively. Furthermore, the training time required to achieve these results is substantially reduced, enabling training on non-specialized workstations. The proposed novelties of capsule architecture, including the integration of the self-attention mechanism and the efficient network structure, contribute to the improved efficiency and performance of presented model. These findings highlight the potential of CNs as a more efficient and effective approach for character classification tasks, with broader applications in various domains.

Key words: efficient capsule networks; Kuzushiji-MNIST; Kuzushiji-49; deep learning.

1. INTRODUCTION

In recent years, deep learning algorithms have emerged as a prevalent method for addressing a diverse range of problems. The context of convolutional neural networks (CNNs) and their numerous applications in image-processing tasks is an especially interesting area of research [1–3]. Such tasks can include object recognition within images as well as assessment of various image parameters, for which CNN-based approaches are typically applied. However, to achieve high levels of accuracy, substantial volumes of training data are required.

Recent research has predominantly concentrated on incremental improvements in performance, e.g. enhancing accuracy by 0.01% percentage points. Although the quality of the obtained results has exhibited a gradual increase, this progress is accompanied by a simultaneous escalation in network complexity and the quantity of data needed to attain the desired accuracy. Unfortunately alternative approaches to the problem are often overlooked.

One approach to addressing this issue involves applying a distinct model, as exemplified by the introduction of convolutional neural networks (CNNs). In order to provide significant improvement to general quality of used solutions, some novel approaches are necessary. In that aspect, capsule networks (CapsNets or CNs) represent a promising methodology, offering an innovative perspective on object classification [4,5].

The primary objective of the authors in developing capsule networks was to enhance the capabilities of CNNs by designing a more efficient solution. The architecture of CNs is characterized by a shallower structure and fewer parameters, which results in improved generalization, especially when encountering new viewpoints. Capsule layers within the network can capture complex relationships between object parts and effectively represent the it as a whole. The learning process for these relationships, known as routing, has been analyzed by several researchers who have attempted to either improve [6–8] or eliminate it [9]. Additionally, other studies have focused on examining the theoretical properties of routing [10, 11].

In their early implementation, capsule networks (CNs) demonstrated state-of-the-art performance on the MNIST dataset. Additionally, obtained results were superior for overlapping digits when compared to CNN-based solutions [11]. CNs at this point were not able to achieve comparable performance on other datasets, such as CIFAR-10 with a 10.6% error rate. It is important to note though, that these scores remained within the range of initial CNN implementations prior to subsequent architectural improvements. CNs offer a series of advantages, such as preservation of position and pose information, reduced training data requirements, and robustness to translations, rotations, and other affine transformations. Considering those factors, it is worthwhile to assess and explore the full potential of these networks.

The CapsNet, is an advanced deep learning architecture introduced in 2017 by Geoffrey Hinton and his research team. This innovative neural network is designed to overcome the limitations inherent in CNNs by applying the concept of cap-

*e-mail: jaroslaw_kurek@sggw.edu.pl

Manuscript submitted 2023-04-22, revised 2023-07-16, initially accepted for publication 2023-09-04, published in December 2023.

sules. Capsules are diminutive components engineered to encapsulate the properties of an object in a more robust and comprehensible manner compared to conventional convolutional methods. In this manuscript, we will explore the fundamental principles of capsule networks and elucidate the associated advantages. Main improvement is provided by architecture modification and used parameter sets. Capsule networks exhibit exceptional performance in diverse tasks with a reduced number of parameters compared to state-of-the-art solutions.

In the conducted experiments, the Efficient Capsule Networks methodology is applied, as delineated by Mazzia *et al.* [12]. Presented model integrates the self-attention technique as a routing mechanism [13]. The rationale for selecting this model is threefold. Firstly, the self-attention mechanism has demonstrated remarkable success in large-scale language models [14]. Secondly, the implementation necessitates fewer parameters than the original capsule network. Lastly, it exhibits superior benchmark performance on the MNIST and smallNORB datasets, offering a robust codebase for the present investigation (available at [15], as a source code to [12]). Although the Efficient approach has received comparatively less attention in the existing literature, it holds substantial potential for a wide array of future applications.

In order to assess the efficacy of the proposed network, the KMNIST dataset was applied as a benchmark [16]. The primary motivation behind this selection was to demonstrate the suitability of capsule networks for 2D datasets, which exhibit fewer viewpoint parameters compared to 3D spaces due to the absence of perspective information. Furthermore, the Kuzushiji-49, a constituent of the KMNIST dataset, encompasses 49 distinct classes, presenting a five-fold increase in complexity in comparison with the MNIST dataset [17]. This serves to illustrate the adaptability of the proposed method to tackle more intricate challenges.

In this paper, we present an improved efficient Capsule Network model for the classification of the Kuzushiji-MNIST benchmark dataset. The model is based on the Efficient CapsNet architecture, which integrates the self-attention mechanism as a routing mechanism. The self-attention mechanism has demonstrated remarkable success in large-scale language models and offers a more efficient alternative to the dynamic routing mechanism used in the original CapsNet.

The primary objective of this research is to evaluate the efficacy of the proposed network on the Kuzushiji-MNIST dataset. We compare the performance of our model to the top solutions in the benchmark and assess its accuracy, training time, and number of parameters. By demonstrating the suitability of capsule networks for this dataset, we aim to highlight the unique contributions of our study and the potential of CNs as a more efficient approach to object classification.

2. CAPSULE NETWORK ADVANTAGES, DISADVANTAGES AND APPLICATIONS

In 2017, Geoffrey Hinton and his colleagues introduced a novel class of neural networks known as capsule networks [6]. The primary components of these networks, termed capsules, aim to encapsulate the attributes of an object, including its orientation, dimensions, and spatial location. Compared to conventional convolutional methods CapsNets offer a more robust and comprehensible approach.

CapsNets are designed to capture the hierarchical relationships between different features in an image. They are also designed to be more robust to variations in the position, scale, and orientation of objects in the input data, improving performance in tasks where these factors are important. Another improvement is dynamic routing (routing by agreement) used to guide information between capsules in a more efficient and meaningful way. This can lead to better performance and improved generalization when object classification is considered. When combined with reduced number of pooling layers than the traditional CNNs, more spatial information can be preserved. Finally, CNs preserve spatial relationships between features, resulting in better handling of overlapping objects.

At the same time CNs are not without drawbacks. They require more complex routing algorithms to establish relationships between different layers. CNs are also relatively new area of research. Fewer pre-trained models, optimization techniques, and best practices are readily available, than in the case of better explored solutions. Due to their increased computational complexity, CapsNets can be challenging to scale for larger input sizes or deeper architectures. Furthermore, while CNs are designed to be more robust to changes in viewpoint, pose, and other affine transformations, it might not be the case for adversarial examples or other kinds of noise. The dynamic routing algorithm used in CapsNets can make it challenging to interpret the learned features and understand the network decision-making process. Also due to routing algorithm and the need for careful hyperparameter tuning, they can be difficult to train. CapsNets have shown promise in certain computer vision tasks, but their applicability and performance across various domains have not yet been thoroughly explored.

Application of CNs for image segmentation might require some adjustment, either by appropriate data preprocessing, or incorporating additional methods in the overall solution. At the same time it can be clearly seen that this approach shows great promise. CNs are able to achieve similar results to state-of-the-art solutions, with fewer parameters and better generalization. When it comes to CNs applications, there are few areas of research, where such solutions are used.

First set of algorithms focuses on different aspects of image segmentation. Approaches belonging to this section are often parts of other, more complex systems, but they can also be a solution in itself. In [18] authors use locally-constrained routing and transformation matrix sharing, in the image segmentation of computer tomography scans of pathological lungs, muscle and adipose (fat) tissue from magnetic resonance imaging scans (MRI) of human subjects' thighs. CN-based processing was able to outperform other methods on all datasets, with less than 5% of the parameters used by U-Net: state-of-the-art solution in biomedical image segmentation. Similarly, in [19] CNs were used for object segmentation in medical data for the pathological lungs from low dose of CT scans, reducing number of parameters by 95.4%, while still maintaining better segmentation accuracy. Different approaches focus on magnetic resonance images of the left ventricle [20], brain tumour automatic segmentation [21] or categorizing cervical lesion imagery [22]. In all cases, CN-based solutions achieve high accuracy, with significantly lower number of parameters.

Second, fairly common set of applications for the capsule networks are ones connected to text analysis elements. CNs are able to handle different contexts and can be well adapted to various problems in this area. Such tasks can include cyberbullying detection [23], text sentiment classification [24] or general text classification [25]. Different set of solutions focuses on improving certain aspects of this process. In [26] in order to reduce number of parameters used for creating word embedding, compositional weighted coding method is proposed. Authors of [27] consider question-answering systems, providing Deep Refinement pipeline. CN and attention mechanism are used, while the pipeline is applied to primarily classify the text into two categories: sincere and insincere. Proposed question classification method outperforms the previously used ones, with the F1 score equal to 0.978. Authors of [28] consider the increased performance of capsule-based solutions, classifying hierarchical multi-label text with a simple CN. In [29] authors explore the possibility of sharing knowledge between related tasks in order to increase the amount of training data. They use capsule-based learning architecture for multi-task purpose. Final claim denotes it as unified, simple and effective, with routing algorithm able to cluster the features for each task in the network.

Solutions focusing on image recognition problems are the most interesting from the point of view of research presented in this paper. There are quite a few approaches in this field, showing that CNs handle such problems relatively well. In [30] the problem of sign language recognition is considered. Traffic sign detection using capsule network is the main topic presented in [31]. Authors use dynamic routing and route by agreement algorithms to instantiate object parameters, such as pose and orientation. As shown in [32], CNs can even be applied to military grade object detection. Authors introduce architecture based on CapsNet, with the presented variant denoted as multilevel CapsNet framework and report that the obtained precision for the object recognition task was superior to many other algorithms. One especially important factor for CNs in such complex problems is routing algorithm used. In [8] authors propose a general-purpose "routing by agreement" method, and the proposed method was able to improve the overall performance of the CNs. Another interesting application, presented in [33], uses capsule networks in the Q-Learning based game algorithms, while in [34] CNs are used in a complex, realistic scenarios of the real world navigation.

Authors of [35] consider similar problem to the one presented in this paper – handwritten character recognition. They use data augmentation to generate realistically modified examples, reflecting actual variations that tend to happen in human writing. Initial number of used samples was equal to 200 per class. Final solution was able to surpass results for the EMNIST-letter dataset, and achieve the results present in EMNIST-balanced, EMNIST-digits, and MNIST datasets. In [12] – solution used in experiments presented in this paper – authors investigate the overall CN efficiency. Proposed algorithm was able to achieve state-of-the-art results on three different datasets, with only 160K parameters – 2% of parameters used by CapsNet.

Despite the prevalent applications of capsule methodologies to image data, a number of studies have concentrated on the video domain. In a manner analogous to the generalization of 2D image-based convolutions to 3D convolutions for processing video frame sequences [36], the traditional 2D convolutional capsule routing was extended to 3D convolutional routing as described in [37]. This 3D convolutional routing approach enables the routing of capsules that are not only spatially proximate but also temporally related, thereby facilitating the generation of higher-layer capsule outputs. In [37] authors introduced a novel video capsule network, denoted as VideoCapsuleNet, which facilitates end-to-end action detection. Study presented in [38] introduces a novel approach called CapsuleVOS, designed for video object segmentation tasks. This method requires an input video sequence with frames containing initial object segmentation. The primary objective of CapsuleVOS is to accurately propagate the object segmentation across the entire video sequence.

Overall, capsule networks tend to work well for image-based problems, offering high accuracy with relatively low number of parameters and better efficiency. CNs in general are an interesting and very promising solution, with vast possibilities. Those advantages were the main reason, why CN-based solution was chosen as a focus of research presented in this paper. For an extensive review of various CapsNet applications, the reader is referred to the comparative study by Vijayakumar *et al.* [39].

3. DATASET

Dataset selection was one of key problems considered for the chosen research area. The CNs have diverse possibilities, but in order to show them, the images used need to be appropriate to the network capabilities.

In that aspect, the Kmnist dataset was considered [16]. The full dataset contains total of three subsets, each with increasing level of complexity. The images are represented as 2D grayscale ones, with examples in each set retaining common size. The first subset, Kuzushiji-MNIST, is a straight-up replacement of original MNIST dataset [40]. This dataset replicates the number of examples for train and test datasets (respectively 60 000 and 10 000), the number of classes (10 total) and image dimensions (grey-scale, 28×28 pixels each). Second subset, Kuzushiji-49 keeps the format but contains 270 912 samples belonging to total of 49 classes. It was designed to engage the machine learning in the field of Japanese literature, and contains instances of Hiragana characters. Final subset, Kuzushiji-Kanji, contains total of 3832 Kanji characters, represented by 140,426 images, with size equal to 64×64 pixels.

While it is a good initial benchmark, first dataset was deemed not complex enough to show full CN capabilities. On the other hand, the third dataset is highly unbalanced – some classes are represented by only single image – and due to high risk of this imbalance influencing the model performance, it is also not the best fit. For the testing purposes, the Kuzushiji-MNIST and Kuzushiji-49 sets were chosen, as they introduces both higher level of complexity and balance required to properly evaluate the CN capabilities. Example images from the Kuzushiji-49 dataset are presented in Fig. 1.

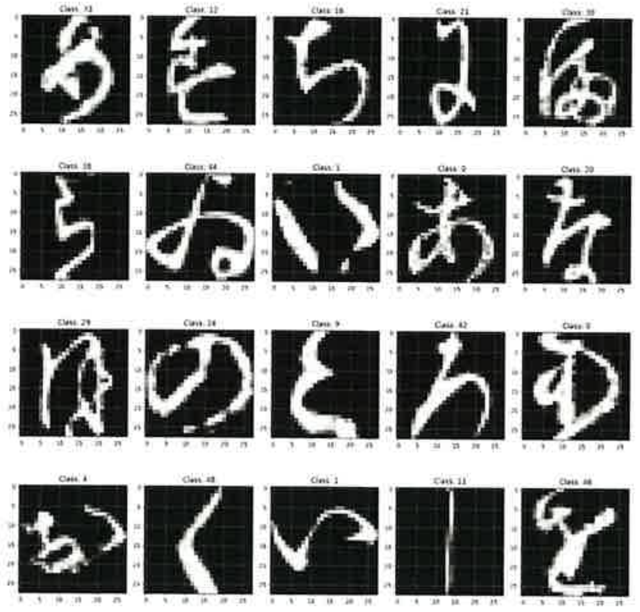


Fig. 1. Sample images representing different Hiragana characters from the Kuzushiji-49 dataset

4. MODELLING AND SETUP

The capsule networks are a new solution, with great promise due to their different approach to the object classification. Last iteration of capsule networks, called CapsNet uses the dynamic routing approach [6], with reconstruction network as regularizer and explainability mechanism.

The model network is a simple structure of single convolutional layer of 256 filters, layer of primary capsules which are connected to digits capsules. It encapsulates multiple scalars to form a vector, where length is the activation of the capsule (scaled to stay in 0–1 range). The vector dimensions are instantiating parameters of an object the capsule is capturing. To interpret those parameters, the generator regularization networks can be used, so each dimension can be perturbed and see what effect on reconstruction it will have. The general structure of CapsNet model is shown in Fig. 2

While the CapsNet model is interesting in the general approach, it is also a base model, which was already improved. In order to accurately assess the possibilities that CN provide, the Efficient CapsNet model was chosen as a base for research presented in this paper.

4.1. Efficient CapsNet

The Efficient CapsNet version of the CN model was chosen for benchmarking in this paper due to a few important reasons.

First of all, this solution was already tested on the MNIST dataset, achieving better results than the initial model, and therefore proving its superior capabilities. Additionally, Efficient CapsNet was also tested on different datasets, including one used for character recognition in the case of letters: EMNIST [35,41]. In both cases, the model achieved good results.

The Efficient CapsNet introduces some important changes in comparison to the original solution [12]. It exchanges the dynamic routing in the first approach, with the self-attention mechanism. Since it is a non-iterative method, this provides significant improvement to the model efficiency in terms of required operations. The study examines the efficiency of capsule networks. It is demonstrated, that an extreme architecture with only 161 000 parameters can still achieve state-of-the-art results on three distinct datasets, using just 6 800 000 (2%) of the original CapsNet parameters. The authors of model chosen as a base for research presented in this paper, also provide a very good python implementation of their solution, using tensorflow [15]. This fact is extremely important, since obtained results can be easily reproduced. The overall architecture of the Efficient CapsNet model is presented in Fig. 3.

To further improve the performance, we introduce a novel non-iterative, highly parallelizable routing algorithm. It can effectively handle a smaller number of capsules, replacing the dynamic routing mechanism. Comprehensive experiments with alternative capsule implementations have confirmed the efficacy of our approach and the capacity of capsule networks to efficiently incorporate visual representations more conducive to generalization. Our solution uses deeper architecture, with 4 convolutional layers before the 2 layers of capsules.

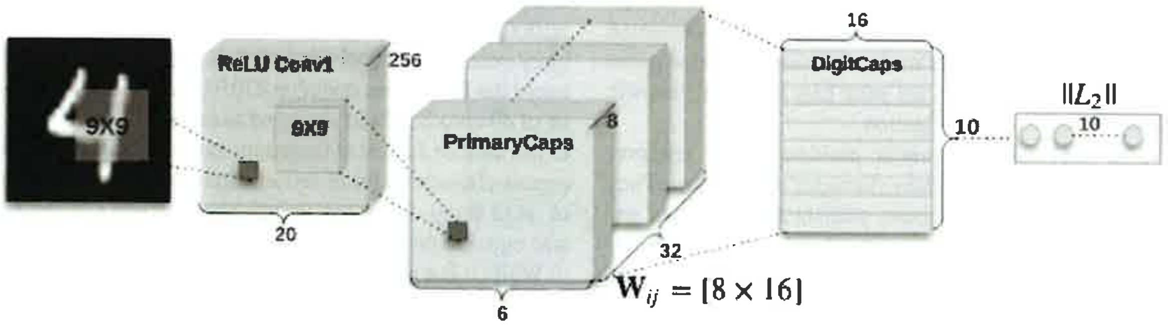


Fig. 2. Architecture of Original Capsule Network [6]

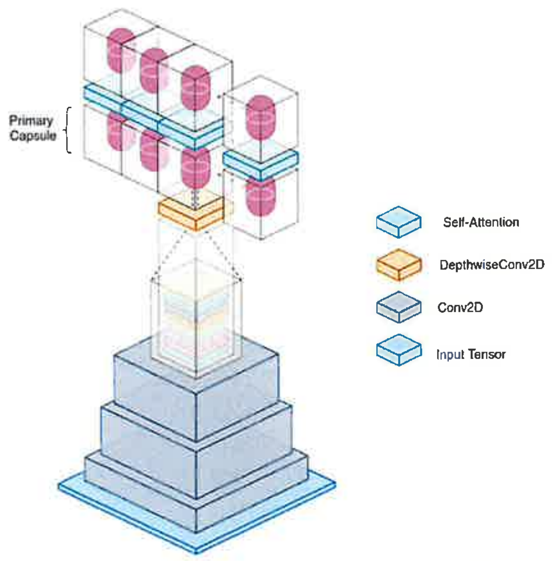


Fig. 3. Architecture of Efficient CapsNet Network [12]

4.2. Improvements and setup

All experiments were performed on a workstation with an Nvidia RTX3080 GPU with 10GB of memory and 32GB of DDR4 SDRAM. We use the TensorFlow 2.10.0 framework with CUDA 11 using Python 3.10.8.

For the dataset division, we used the well-established practices, present in the general CNN approaches. The sets were divided into three subsets: train, eval and test. The last one was provided by the authors of KMNIST. Finally, 20% of training set was used as validation for establishing the number of epochs and hyperparameters in order to select the best model.

The model was trained for 150 epochs using hyperparameters provided in [12]. The learning rate was modified to start at 0.0002 and minimal learning rate was set at 0.00003.

The overall architecture was also modified, increasing number of convolutional layers and primary capsules. In the presented approach total of 6 convolutional layers were used, each followed by batch normalization. After convolution layers, the primary capsule layer was placed. For the Kuzushiji-MNIST dataset, this layer consisted of 32 8-dimensional capsules. In case of Kuzushiji-49, the number of dimensions was increased, resulting in 32 10-dimentional capsules. Final layer contained number of capsules corresponding directly to number of recognized classes: 10 for the first set, and 48 for the Hiragana character set.

After the model setup, the following number of parameters were obtained for each dataset: 581 792 for Kuzushiji-MNIST, and 1 741 120 for Kuzushiji-49. Out of both parameters sets, the number of non-trainable ones equalled 1152 for the first and 1792 for the second one. Full layer structure for both models – including type of layer, output shape and number of used parameters - are presented in Table 1 and 2, respectively.

5. COMPARISON AND PERFORMANCE ANALYSIS

The novelty of our approach lies in the modifications made to the Efficient CapsNet model, which resulted in improved performance on the Kuzushiji-MNIST and Kuzushiji-49 datasets. By comparing the two models, we can gain valuable insights into the specific improvements made and their impact on performance.

The original CapsNet model, introduced by Sabour *et al.* [6], was designed to overcome the limitations of convolutional neu-

Table 1
Network architecture for the Efficient CapsNet trained on Kuzushiji-MNIST dataset

Layer (type)	Output Shape	Param
input_8 (InputLayer)	[(None, 28, 28, 1)]	0
conv2d_5 (Conv2D)	(None, 24, 24, 32)	832
batch_normalization_5 (BatchNormalization)	(None, 24, 24, 32)	128
conv2d_6 (Conv2D)	(None, 22, 22, 64)	18496
batch_normalization_6 (BatchNormalization)	(None, 22, 22, 64)	256
conv2d_7 (Conv2D)	(None, 20, 20, 96)	55392
batch_normalization_7 (BatchNormalization)	(None, 20, 20, 96)	384
conv2d_8 (Conv2D)	(None, 18, 18, 128)	110720
batch_normalization_8 (BatchNormalization)	(None, 18, 18, 128)	512
conv2d_9 (Conv2D)	(None, 8, 8, 256)	295168
batch_normalization_9 (BatchNormalization)	(None, 8, 8, 256)	1024
primary_caps_1 (PrimaryCaps)	(None, 32, 8)	16640
fc_caps_1 (FCCaps)	(None, 10, 32)	82240
length_capsnet_output (Length)	(None, 10)	0
Total params:	581 792	
Trainable params:	580 640	
Non-trainable params:	1 152	

Table 2
Network architecture for the Efficient CapsNet trained on Kuzushiji-49 dataset

Layer (type)	Output Shape	Param
input_9 (InputLayer)	[(None, 28, 28, 1)]	0
conv2d_6 (Conv2D)	(None, 24, 24, 32)	832
batch_normalization_6 (BatchNormalization)	(None, 24, 24, 32)	128
conv2d_7 (Conv2D)	(None, 22, 22, 64)	18496
batch_normalization_7 (BatchNormalization)	(None, 22, 22, 64)	256
conv2d_8 (Conv2D)	(None, 20, 20, 96)	55392
batch_normalization_8 (BatchNormalization)	(None, 20, 20, 96)	384
conv2d_9 (Conv2D)	(None, 18, 18, 128)	110720
batch_normalization_9 (BatchNormalization)	(None, 18, 18, 128)	512
conv2d_10 (Conv2D)	(None, 16, 16, 256)	295168
batch_normalization_10 (BatchNormalization)	(None, 16, 16, 256)	1024
conv2d_11 (Conv2D)	(None, 7, 7, 320)	737600
batch_normalization_11 (BatchNormalization)	(None, 7, 7, 320)	1280
primary_caps_1 (PrimaryCaps)	(None, 32, 10)	16000
fc_caps_1 (FCCaps)	(None, 49, 32)	503328
length_capsnet_output (Length)	(None, 49)	0
Total params:	1 741 120	
Trainable params:	1 739 328	
Non-trainable params:	1 792	

ral networks (CNNs) by using capsules instead of neurons. Capsules are groups of neurons that represent different properties of an object, such as its orientation, dimensions, and spatial location. The primary advantage of capsules is their ability to capture hierarchical relationships between different features in an image, making them more robust to variations in position, scale, and orientation.

The Improved Efficient CapsNet model builds upon the original CapsNet model by incorporating the self-attention mechanism as a routing mechanism, as proposed by Mazzia *et al.* [12]. This modification replaces the dynamic routing mechanism used in the original model, resulting in a more efficient and parallelizable routing algorithm. The self-attention mechanism has been successful in large-scale language models and has demonstrated the ability to capture complex relationships between different parts of an object.

In terms of performance, the Improved Efficient CapsNet model achieved comparable accuracy to the original CapsNet model on the Kuzushiji-MNIST and Kuzushiji-49 datasets. However, the Improved Efficient CapsNet model achieved this performance with significantly fewer parameters, making it a more efficient solution. The total number of parameters used in the Improved Efficient CapsNet model was 0.58M for the Kuzushiji-MNIST dataset and 1.7M for the Kuzushiji-49 dataset, compared to the original CapsNet model, which used 26.2M parameters. This reduction in parameters is particularly important for practical applications, as it allows for faster training times and reduces the computational resources required.

Furthermore, the Improved Efficient CapsNet model demonstrated faster training times compared to the original CapsNet model. The Improved Efficient CapsNet model was trained for 150 epochs, with a total training time of 50 minutes for the Kuzushiji-MNIST dataset and 4 hours and 30 minutes for the Kuzushiji-49 dataset. In contrast, the original CapsNet model required 1800 epochs and a training time of 290 hours for the Kuzushiji-49 dataset. This significant reduction in training time makes the Improved Efficient CapsNet model more practical and accessible for researchers and practitioners.

In conclusion, the Improved Efficient CapsNet model presented in this paper offers several improvements over the original CapsNet model. The incorporation of the self-attention mechanism as a routing mechanism results in a more efficient and parallelizable routing algorithm. This modification allows for faster training times and reduces the number of parameters required, while still achieving comparable accuracy on the Kuzushiji-MNIST and Kuzushiji-49 datasets. The Improved Efficient CapsNet model provides valuable insights into the specific improvements made and their impact on performance, making it a promising solution for character classification tasks.

6. RESULTS AND DISCUSSION

The main goal of research presented in this paper was to evaluate the Efficient CapsNet model in terms of applicability and overall performance. As shown in various research paper, the CNs can be used to solve different problems, ranging from text classification, to image segmentation and object recognition [19, 23, 26, 30–32]. What is even more important is that prepared models were able to achieve similar accuracy to state-of-the-art solutions in selected areas, using significantly fewer parameters.

Two subsets of Kmnist dataset were used to evaluate prepared model. Prepared solution is a modification of Efficient CapsNet approach [12]. According to specifications provided by the authors of the dataset, the metric accuracy on the Kuzushiji-49 was balanced. In order to evaluate the model performance, mean accuracy for all classes was calculated. Additionally, model trained on Kuzushiji-MNIST subset was used as a base evaluation for the modified solution used in experiments. Final results were compared to the best-performing models in given dataset, and are presented in Tables 3 and 4, as of December 2022.

Overall, both models managed to score in the top ten in their respective benchmarks. The network trained with Kuzushiji-MNIST dataset scored in 4th place, while the Kuzushiji-49 one was placed on 6th position. Both models achieved relatively high accuracy, reaching overall score equal to respectively 98.43% and 96.32%. While neither model was able to reach first place, both approaches performed reasonably well. The results are more than satisfactory, showing that capsule network architecture generalizes well for high number of classes. Presented solution can be trained in reasonable amount of time on consumer grade hardware. At the same time the achieved accuracy is on par with models using far larger number of parameters. For the Kuzushiji-MNIST dataset the overall accuracy difference reaches only 0.91%, while in the case of Kuzushiji-49 it equals 1.97%. While the difference is not negligible, it is small enough that the presented approach has a resonable chance to beat those scores after some improvements. In that aspect, one possible area of future work might include using ensemble solutions, or different overall network structure.

In Table 3, we present the eight leading models regarding their accuracy on the Kuzushiji-MNIST dataset as of December 2022. The shake-shake-26 2x96d (S-S-I) model, with Cutout 14, holds the first place, boasting an impressive accuracy of 99.34% but with a relatively high number of parameters (26.2M). Interestingly, despite not using a pre-trained network, it only takes roughly 6 hours and 46 minutes to train.

The "Improved Efficient Capsnet," the model at the core of study presented in this paper, achieved an accuracy of 98.43%, putting it at 4th place. Notably, it achieved this performance with significantly fewer parameters (0.58M), demonstrating its efficacy and computational efficiency. It completed training in just 50 minutes over 150 epochs, underscoring its relatively fast training time.

In comparison, the ResNet18 + VGG Ensemble model, with a slightly higher accuracy of 98.90%, required 26M parameters and had the benefit of using a pre-trained network, showing that our model can perform competitively without such advantages, while being significantly lighter.

Moving on to Table 4, which presents the top eight models regarding their balanced accuracy scores on the more complex Kuzushiji-49 dataset. Here, the "Improved Efficient Cap-

Table 3

Top 8 accuracy scores for Kuzushiji-MNIST as for December 2022

Place	Model	Accuracy	Number of parameters	Pretrained network	Number of epochs	Training time
1	shake-shake-26 2x96d (S-S-I), Cutout 14	99.34%	26.2M	No	200	6h46m
2	ResNet18 + VGG Ensemble	98.90%	26M	Yes	N/A	3m
3	PreActResNet-18 Manifold Mixup	98.83%	11M	No	200	N/A
4	**Improved Efficient Capsnet**	**98.43%**	**0.58M**	**No**	**150**	**50m**
5	PreActResNet-18 + Input Mixup	98.41%	11M	No	200	N/A
6	PreActResNet-18	97.82%	11M	No	200	N/A
7	Original Capsule Networks	97.66%	6.8M	No	150	1h35m
8	Keras Simple CNN Benchmark	94.63%	1.2M	No	12	N/A

Table 4

Top 8 balanced accuracy scores for Kuzushiji-49 as for December 2022

Place	Model	Accuracy	Number of parameters	Pretrained network	Number of epochs	Training time
1	Shake-Shake-26 2x96d (cutout 14))	98.29%	26.2M	No	1800	290h
2	PreActResNet-18 + Manifold Mixup	97.33%	11M	No	200	N/A
3	DenseNet-100 (k=12)%	97.32%	7M	No	1500	47h39m
4	PreActResNet-18 + Input Mixup	97.04%	11M	No	200	N/A
5	PreActResNet-18	96.64%	11M	No	200	N/A
6	**Improved Efficient Capsnet**	**96.32%**	**1.7M**	No	150	4h30m
7	Original Capsule Networks	91.37%	12.5M	No	150	14h
8	Keras Simple CNN Benchmark	89.36%	1.2M	No	12	N/A

snet" placed 6th with an accuracy of 96.32%. Although this is a slightly lower ranking than the previous table, it is worth noting that the model only used 1.7M parameters, confirming its ability to perform well even with lower computational resources. The training time was also relatively short, requiring only 4 hours and 30 minutes for 150 epochs.

In comparison, the leading Shake-Shake-26 2x96d (cutout 14) model achieved an accuracy of 98.29% but at the cost of a substantially larger number of parameters (26.2M) and a remarkably long training time of 290 hours.

In summary, the "Improved Efficient Capsnet" model demonstrates competitive performance on both the Kuzushiji-MNIST and Kuzushiji-49 datasets, achieving high accuracy rates with relatively few parameters and shorter training times, indicating its potential as a more efficient approach for Kuzushiji character recognition tasks.

The accuracy metric is commonly used due to its simplicity and direct interpretation. However, it does not provide a comprehensive picture of the model performance, particularly when the classes are imbalanced. For this reason, we consider precision, recall, and F1-score metrics in addition to accuracy for our performance assessment.

Precision is the ratio of true positive predictions to the total positive predictions, which indicates the exactness or quality of the model. Recall (also known as sensitivity) is the ratio of true positive predictions to the total actual positives, which illustrates the completeness or quantity the model can provide. The F1-score is the harmonic mean of precision and recall, providing a balanced measure between precision and recall.

In an interesting turn of events, we observed that the average and weighted average of precision, recall, and F1-score metrics across all classes are identical to the accuracy metric for our capsule network model applied on the Kuzushiji-MNIST dataset. This unusual equivalence is consistent for all models, even when tested on the Kuzushiji-49 dataset, with differences falling within a narrow margin of ±0.01 percentage points.

To illustrate, we present a detailed example of precision, recall, and F1-score metrics for each class from the Original Capsule Networks model on the Kuzushiji-MNIST dataset in Table 5.

The occurrence of such equalities is atypical, given the fundamental differences in these performance metrics. They each serve distinct purposes and are not expected to agree so closely unless the dataset is perfectly balanced and the model performs equally well on all classes. This unique finding indicates a remarkable robustness and balance in the classification capability of our improved capsule network model.

The analysis prompts a more in-depth investigation into the properties and configuration of the capsule network that yield such an unusual performance consistency across metrics. This will form the basis for subsequent research aimed at unravelling the inherent characteristics and peculiarities of this model, particularly in the context of Kuzushiji character recognition.

It is also important to point out that while analysing the code of better-scoring solutions, it was noted that some were using test set as a validation one for the training purposes. Such actions can lead to model overfitting, and while it achieves better

Table 5
Example comparison of accuracy metric with other metrics for the Original Capsule Networks model on the Kuzushiji-MNIST dataset

No. of class	Precision [%]	Recall [%]	F1-score [%]
0	96.46	98.20	97.32
1	97.98	97.10	97.54
2	98.25	95.40	96.80
3	97.16	99.10	98.12
4	96.65	95.20	95.92
5	98.18	97.20	97.69
6	97.82	98.70	98.26
7	99.10	98.70	98.90
8	98.11	98.40	98.25
9	96.95	98.60	97.77
Average [%]	97.66	97.66	97.66
Weighted average [%]	97.66	97.66	97.66
Accuracy [%]	97.66		

results on the original dataset, the overall solution versatility will suffer. Both models presented in this paper were trained using the established best practices for similar problems, splitting the dataset into train, eval and test datasets.

Presented approach is time- and resource-efficient, allowing incorporation of the k-fold cross-validation. In the case of the deep learning approach, due to long computational time, we utilized 5-fold cross-validation ($k = 5$). The validation was performed k-times with increasing training time. The computation was efficient enough to still remain within capabilities of personal workstation with reasonable overall training period. It is important to note that the top performing solution for the Kuzushiji-49 set (shake-shake-26 2x96d) required 1800 epochs and total of 290 hours of training. Experiments for that network were performed with eight Tesla V100 GPUs, with 26.2M of final parameters. The Efficient CapsNet approach presented in this paper used total of 150 epochs, while overall training time equalled 4.5 hours. The experiments were done on significantly less efficient machine using single GPU (full specification of the used workstation is presented in 4.2), while final parameters count equalled only 1.7M for the resulting model.

7. CONCLUSIONS

In this paper an approach to character classification for Kuzushiji-MNIST and Kuzushiji-49 datasets is presented. The solution uses model based on Efficient CapsNet architecture, taking advantage of strong points of capsule-network-based solutions, such as robustness, simpler structure and better generalization.

As shown by the obtained results, capsule networks are a promising solution, when chosen benchmark field is considered. Prepared model was able to score in the top 10 solutions in the two chosen trials. It was able to achieve very similar results to the top-performing ones, with significantly fewer network parameters used. The total accuracy difference between solution presented in this paper, and ones that scored in the first place for both benchmark was 0.91% for the Kuzushiji-MNIST dataset, and 1.97% for Kuzushiji-49. These results were achieved in much shorter training times, allowing the overall process to be performed on not-dedicated GPU workstations.

Overall, CN-based solutions show great promise, and while current approach to Efficient CapsNet modification was not able to achieve top results, it still offers significant advantages. CNs have better generalization and use fewer parameters. The total accuracy loss is more than compensated with shorter training time. Simpler network structure with fewer parameters used results in solution with much wider range of applications, while still leaving some room for future improvements.

REFERENCES

[1] J. Kurek, B. Swiderski, A. Jegorowa, M. Kruk, and S. Osowski, "Deep learning in assessment of drill condition on the basis of images of drilled holes," in *Eighth International Conference on Graphic and Image Processing (ICGIP 2016)*, Y. Wang, T. D. Pham, V. Vozenilek, D. Zhang, and Y. Xie, Eds., vol. 10225, International Society for Optics and Photonics. SPIE, 2017, p. 102251V, doi: 10.1117/12.2266254.

[2] A. Jegorowa, J. Kurek, I. Antoniuk, W. Dołowa, M. Bukowski, and P. Czarniak, "Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard," *Wood Sci. Technol.*, vol. 55, no. 1, pp. 271–293, Jan 2021, doi: 10.1007/s00226-020-01245-7.

[3] J. Kurek *et al.*, "Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network," *Mach. Graph. Vis.*, vol. 28, no. 1/4, p. 13–23, Dec. 2019, doi: 10.22630/MGV.2019.28.1.2.

[4] G. Hinton, A. Krizhevsky, and S. Wang, "Transforming auto-encoders," in *Artificial Neural Networks and Machine Learning – ICANN 2011*, vol. 6791, 06 2011, pp. 44–51, doi: 10.1007/978-3-642-21735-7_6.

[5] A. Jegorowa, J. Górski, J. Kurek, and M. Kruk, "Use of nearest neighbors (k-nn) algorithm in tool condition identification in the case of drilling in melamine faced particleboard," *Maderas-Cienc. Tecnol.*, vol. 22, no. 2, p. 189–196, 2020, doi: 10.4067/S0718-221X2020005000205.

[6] S. Sabour, N. Frosst, and G.E. Hinton, "Dynamic routing between capsules," 2017, doi: 10.48550/ARXIV.1710.09829. [Online]. Available: https://arxiv.org/abs/1710.09829

[7] F.D.S. Ribeiro, G. Leontidis, and S.D. Kollias, "Capsule routing via variational bayes," *CoRR*, vol. abs/1905.11455, 2019. [Online]. Available: http://arxiv.org/abs/1905.11455

[8] F.A. Heinsen, "An algorithm for routing capsules in all domains," *arXiv preprint arXiv:1911.00792*, 2019.

[9] A. Byerly, T. Kalganova, and I. Dear, "A branching and merging convolutional network with homogeneous filter capsules," *CoRR*, vol. abs/2001.09136, 2020. [Online]. Available: https://arxiv.org/abs/2001.09136

[10] S.R. Venkatraman, A. Anand, S. Balasubramanian, and R.R. Sarma, "Learning compositional structures for deep learning: Why routing-by-agreement is necessary," *CoRR*, vol. abs/2010.01488, 2020. [Online]. Available: https://arxiv.org/abs/2010.01488

[11] D. Wang and Q. Liu, "An optimization view on dynamic routing between capsules," 2018. [Online]. Available: https://openreview.net/forum?id=HJjtFYJDf

[12] V. Mazzia, F. Salvetti, and M. Chiaberge, "Efficient-CapsNet: capsule network with self-attention routing," *Sci. Rep.*, vol. 11, no. 1, jul 2021, doi: 10.1038/s41598-021-93977-0.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017, doi: 10.48550/ARXIV.1706.03762. [Online]. Available: https://arxiv.org/abs/1706.03762

[14] T.B. Brown, B. Mann, N. Ryder, and E.A. Subbiah, "Language models are few-shot learners," 2020, doi: 10.48550/ARXIV.2005.14165. [Online]. Available: https://arxiv.org/abs/2005.14165

[15] V. Mazzia, F. Salvetti, and M. Chiaberge, "Github repository for efficient capsnet." 2021. [Online]. Available: https://github.com/EscVM/Efficient-CapsNet

[16] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. (2018) Deep learning for classical japanese literature.

[17] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[18] R. LaLonde, Z. Xu, I. Irmakci, S. Jain, and U. Bagci, "Capsules for biomedical image segmentation," *Med. Image Anal.*, vol. 68, p. 101889, 2021.

[19] R. LaLonde and U. Bagci, "Capsules for object segmentation," *arXiv preprint arXiv:1804.04241*, 2018.

[20] Y. He, W. Qin, Y. Wu, M. Zhang, Y. Yang, X. Liu, H. Zheng, D. Liang, and Z. Hu, "Automatic left ventricle segmentation from cardiac magnetic resonance images using a capsule network," *J. X-Ray Sci. Technol.*, vol. 28, no. 3, pp. 541–553, 2020.

[21] M. Elmezain, A. Mahmoud, D.T. Mosa, and W. Said, "Brain tumor segmentation using deep capsule network and latent-dynamic conditional random fields," *J. Imaging*, vol. 8, no. 7, p. 190, 2022.

[22] X. Zhang and S.-G. Zhao, "Cervical image classification based on image segmentation preprocessing and a capsnet network model," *Int. J. Imaging Syst. Technol.*, vol. 29, no. 1, pp. 19–28, 2019, doi: 10.1002/ima.22291.

[23] A. Kumar and N. Sachdeva, "Multimodal cyberbullying detection using capsule network with dynamic routing and deep convolutional neural network," *Multimedia Syst.*, vol. 28, p. 2043–2052, 2022.

[24] B. Chen, Z. Xu, X. Wang, L. Xu, and W. Zhang, "Capsule network-based text sentiment classification," *IFAC-PapersOnLine*, vol. 53, no. 5, pp. 698–703, 2020.

[25] J. Kim, S. Jang, E. Park, and S. Choi, "Text classification using capsules," *Neurocomputing*, vol. 376, pp. 214–221, 2020.

[26] H. Ren and H. Lu, "Compositional coding capsule network with k-means routing for text classification," *Pattern Recognit. Lett.*, vol. 160, pp. 1–8, 2022.

[27] D.K. Jain, R. Jain, Y. Upadhyay, A. Kathuria, and X. Lan, "Deep refinement: Capsule network with attention mechanism-based system for text classification," *Neural Comput. Appl.*, vol. 32, pp. 1839–1856, 2020.

[28] J.S. Manoharan, "Capsule network algorithm for performance optimization of text classification," *J. Soft Comput. Paradigm (JSCP)*, vol. 3, no. 01, pp. 1–9, 2021.

[29] L. Xiao, H. Zhang, W. Chen, Y. Wang, and Y. Jin, "Mcapsnet: Capsule network for text with multi-task learning," in *Proceedings of the 2018 conference on empirical methods in natural language processing*, 2018, pp. 4565–4574.

[30] F. Beşer, M.A. Kizrak, B. Bolat, and T. Yildirim, "Recognition of sign language using capsule networks," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[31] A.D. Kumar, "Novel deep learning model for traffic sign detection using capsule networks," *arXiv preprint arXiv:1805.04424*, 2018.

[32] B. Janakiramaiah, G. Kalyani, A. Karuna, L.N. Prasad, and M. Krishna, "Military object detection in defense using multi-level capsule networks," *Soft Comput.*, vol. 27, p. 1045–1059, 2023, doi: 10.1007/s00500-021-05912-0.

[33] P.-A. Andersen, "Deep reinforcement learning using capsules in advanced game environments," *arXiv preprint arXiv: 1801.09597*, 2018.

[34] T. Molnar and E. Culurciello, "Capsule network performance with autonomous navigation," *arXiv preprint arXiv:2002.03181*, 2020.

[35] V. Jayasundara, S. Jayasekara, H. Jayasekara, J. Rajasegaran, S. Seneviratne, and R. Rodrigo, "TextCaps: Handwritten character recognition with very small datasets," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, jan 2019, doi: 10.1109/wacv.2019.00033.

[36] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, dec 2015, pp. 4489–4497, doi: 10.1109/ICCV.2015.510. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.510

[37] K. Duarte, Y. S. Rawat, and M. Shah, "Videocapsulenet: A simplified network for action detection," 2018.

[38] D. Ma and X. Wu, "Capsulerrt: Relationships-aware regression tracking via capsules," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 10 948–10 957.

[39] T. Vijayakumar, "Comparative study of capsule neural network in various applications," *J. Artif. Intell.*, vol. 1, no. 01, pp. 19–27, 2019.

[40] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[41] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "EMNIST: an extension of MNIST to handwritten letters," *CoRR*, vol. abs/1702.05373, 2017. [Online]. Available: http://arxiv.org/abs/1702.05373

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M., Antoniuk I., Kurek J.: Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification, Bulletin of the Polish Academy of Sciences, Technical Sciences, 2023, vol. 71, nr 6, s.1-10, Numer artykułu:e147338. DOI:10.24425/bpa-sts.2023.147338
mój indywidualny udział w jej powstaniu polegał na:

1. Koncepcji.
2. Metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.
7. Pozyskaniu i utrzymaniu danych.

[illegible]
Podpis

Warszawa, 27-01-2025

Izabella Antoniuk
izabella_antoniuk@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M., Antoniuk I., Kurek J.: Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification, Bulletin of the Polish Academy of Sciences, Technical Sciences, 2023, vol. 71, nr 6, s.1-10, Numer artykułu:e147338. DOI:10.24425/bpa-sts.2023.147338
mój indywidualny udział w jej powstaniu polegał na:

1. Przygotowaniu pierwszego szkicu.
2. Edycji i korekcji artykułu.

[illegible]
Podpis

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M., Antoniuk I., Kurek J.: Improved efficient capsule network for Kuzushiji-MNIST benchmark dataset classification, Bulletin of the Polish Academy of Sciences, Technical Sciences, 2023, vol. 71, nr 6, s.1-10, Numer artykułu:e147338. DOI:10.24425/bpa-sts.2023.147338
mój indywidualny udział w jej powstaniu polegał na:

1. Koordynacja działań związanych z tworzeniem koncepcji.
2. Metodologii.
3. Nadzorze.
4. Administracji.
5. Walidacji.

[illegible]

Podpis

10.4. Publikacja 4

PUBLIKACJA 4

Multiple Input CNN Architecture for Tool State Recognition in Milling Process Based on Time Series Signals

Michał Bukowski, Izabella Antoniuk, Karol Szymanowski, Artur Krupa, Jarosław Kurek

Operations Research and Decisions 2024: 34(3), 41-60.

https://doi.org/10.37190/ord240303

Punktacja: 70 (200 w dniu wysłania artykułu do czasopisma), IF: 0,7

Vol. 34, No. 3 (2024) | DOI: 10.37190/ord240303

OPEN ACCESS

Operations Research and Decisions

www.ord.pwr.edu.pl

OPERATIONS RESEARCH AND DECISIONS QUARTERLY

Multiple input CNN architecture for tool state recognition in the milling process based on time series signals

Michał Bukowski[1] Izabella Antoniuk[1] Karol Szymanowski[2] Artur Krupa[1] Jarosław Kurek[1*]

[1]*Department of Artificial Intelligence, Institute of Information Technology, Warsaw University of Life Sciences, Warsaw, Poland*

[2]*Department of Mechanical Processing of Wood, Institute of Wood Sciences and Furniture, Warsaw University of Life Sciences, Warsaw, Poland*

[*]*Corresponding author, email address: jaroslaw_kurek@sggw.edu.pl*

Abstract

The study presents a tailored application of a multiple-input convolutional neural network (CNN) for tool state recognition in the milling process. Our approach uniquely applies an 11-input CNN to classify tool wear in chipboard milling, utilizing scalogram images derived from time-series signals. The primary objective was to categorize tool wear into three classes: green, yellow, and red, signifying the progression of wear. The study involved 75 samples (25 samples per class), each comprising 11 signals transformed into scalograms via continuous wavelet transform. The dataset of 825 scalogram images enabled the development of a CNN-based diagnostic model, achieving a notable accuracy of 96.00%, which is an improvement over a previous methods (93.33%).

Keywords: *convolution neural network, tool condition monitoring, multiple input convolution neural network, deep learning*

1. Introduction

Furniture manufacturing is a complicated process, involving multiple steps and various problem along the way. Different stages require high levels of precision. Poor or ill-timed decisions about tool maintenance and/or exchange can result in a faulty product, not meeting the requirements and causing potential loss for the manufacturing company [6, 10, 14, 16]. One of the key elements influencing the quality of products, is tool condition monitoring during the milling process. This operation can be performed manually but in that instance it is a time consuming one, with high possibility of operator missing key indications of deteriorating tool state. Automating the process and providing additional insight is a high priority, while the overall process is widely and excessively discussed one [8, 29, 30]. Important aspect of this

Received 30 September 2023, accepted 3 July 2024, published online 17 October 2024
ISSN 2391-6060 (Online)/© 2024 Authors

The costs of publishing this issue have been co-financed by the program *Development of Academic Journals* of the Polish Ministry of Education and Science under agreement RCN/SP/0241/2021/1

procedure is the gradual deterioration of the cutting edge. In order to avoid unplanned and potentially problematic tool stoppage, an automatic and online-working solution is necessary. Such system usually needs to incorporate external identification of edge wear, such as signals recorded while the machine is operating [11, 27].

Numerous works focusing on wood-based materials are available [20, 21], with some approaches focused on determining most useful signals for identifying the tool condition during machining process [9, 13, 16, 23, 28, 31]. While the overall proceedings are well described, there still is a need for a more precise solutions, easier to incorporate in the production environment.

Machine learning is one area, where such solutions can be created. There are various methods and applications available, both for vision and signal based systems [1, 4, 5, 13, 25]. Different researchers consider problems related to the production process, its different aspects, and applications including very complex problem such as tree species recognition [7]. Such approaches show, that different algorithms can be well adapted to even most complicated tasks, assuming appropriate input data will be prepared.

There are various solutions focusing on different parts of the tool condition monitoring process. While the signals are common input in the wood industry applications, different solutions can include various machine learning approaches, often using different data. One common type of input are images. For example, CNNs are often paired with such data [6, 14, 15, 18, 19]. Such solutions often include various approaches to the problem of training artificial neural networks, like data augmentation or transfer learning using networks such as AlexNet [12, 22], prepared for ImageNet database [24, 26]. Although strictly image-based approaches are quite popular, there are some problems related to this kind of input. Usually in order to achieve satisfactory accuracy rate, large amounts of training data are necessary. Requirements for any given problem might differ, requiring tight cooperation with the manufacture. In that aspect changes in signals are easier to measure, assuming the proposed methodology is able to compute large amounts of data obtained from various sensors.

Using signals as a base input for a neural network poses series of problems. Firstly, the changes in signals are not always consistent throughout all data obtained from used sensor set. This can lead to problems in final solution, making it unable to accurately point out the key indicators of the deteriorating tool quality. Additionally, especially for signals that require very high precision while recording, the size of input data files can differ significantly between various sources. Research presented in this paper is largely based on the idea of using data thus available in a more optimal way, without losing the advantage of more precise measurements it provides.

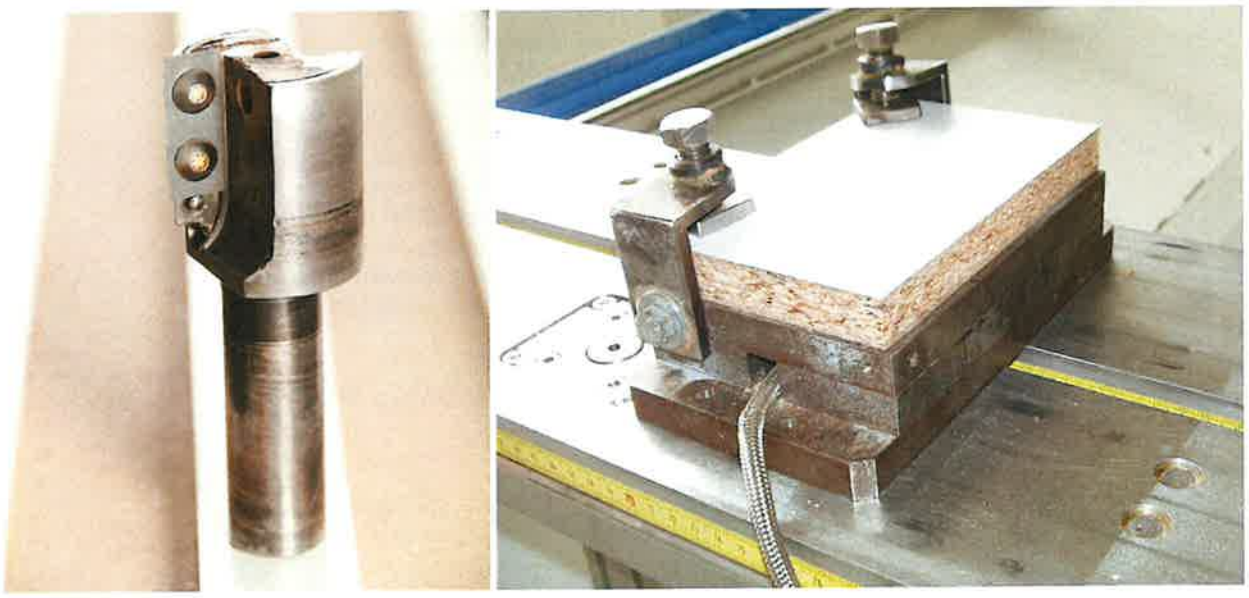
A relatively small section of approaches in this area considers transferring signals into images. For examplem, in [3] sound signals are converted to images using short-time Fourier transform. Authors first denoise the original signal and then convert it to images. After that process, the pre-trained CNN model is used to perform the deep feature extraction [18]. Last stage of method uses a support vector machine for the classification. Different solution [2], similarly to the approach presented in this paper, uses set of signals converted to scalograms. Authors use constant-Q transform with non-stationary Gabor transform, fusing features from vibration and acoustic signals with multiple input CNN in order to diagnose state of induction motor. They chose above methodology, with the assumption that computing continuous wavelet transform (CWT) is too time consuming, and in turn sacrificing the overall method accuracy.

In the presented research, a multiple input CNN architecture, using total of 11 separately recorded signals, is prepared and tested. Obtained data is down-sampled and converted to scalograms using CWT, in order to retain as much information from the original source as possible. To the best of authors knowledge, the proposed network architecture is first of its kind, both in general approach to signal preprocessing and incorporation as input, as well as the overall structure.

The rest of the work is organized as follows. In section 2, an overall discussion of used materials and data acquisition process is presented. Section 3 outlines the data preparation process, including the preprocessing stage and scalogram generation. The resulting dataset, CNN structure and performed experiments are described in section 4. Final paper conclusions are presented in section 6.

2. Materials

The main goal of the experiment was to build a diagnostic system capable to accurately measure the level of tool wear. The tests were done using a CNC machining centre (Jet 130; Busellato, Thiene, Italy), equipped with single edge cutter head with exchangeable carbide cutting edge, 40mm in diameter (Faba SA, Baboszewo, Poland), presented in Figure 1a). The mounted piece of chipboard is shown in Figure 1b).



a) b)

Figure 1. Edge cutter head (a) and example of chipboard piece mounted on the machining centre (b)

The state of the used tool was denoted as one of the tree states: green which means tool in a good state, yellow, defining tool in an intermediate state, and red, denoting tool that needs to be exchanged due to high wear level. The ranges for each state were measured using VB-max parameter defined as maximal flank wear. The ranges for each class were defined as follows:

- VB-max below 0.15 – green class;
- VB-max in the range: (0.151; 0.25) – yellow class;
- VB-max equal or above 0.3 – red class.

During the tests, a sample of a chipboard (DecoBoard P2, Pfleiderer) with dimensions of 300×150 mm was mounted on a measuring platform and a groove with a depth of 6 mm was milled. The spindle speed was set at 18,000 rpm with the feed rate equal to 0.15 mm per tooth.

2.1. Data acquisition

In order to ensure that acquired data is of the highest possible quality, a set of specialized sensors was selected for the measurement process. The entire set included following elements:

- X and Y forces (50kHz sampling rate) (Kistler 9601A sensor),
- acoustic emission (Kistler 8152B),
- noise (Brüel & Kjær 4189),
- vibrations (Kistler 5127B),
- finest HR 30 current.

During the tests, the milling process was stopped in order to determine the tool wear at different stages. The Mitutoyo TM-505 workshop microscope was used for those measurements. Standards used during the experiments are presented in Table 3 and the real physical properties are listed in Table 2. Full list of used equipment is shown in Table 1, while Figure 2 outlines the test stand setup.

Table 1. Apparatuses used to test the physical and mechanical parameters of wood-based materials

Parameter	Measuring apparatus
Density, kg/m^3	laboratory scale, calipers
Static bending strength, MPa	VebThuringerIndustriewerk SP 10
Elasticity modulus, MPa	VebThuringerIndustriewerk SP 10
Screw retention, N/mm	VebThuringerIndustriewerk SP 10
Tensile strength, MPa	VebThuringerIndustriewerk SP 10
Brinell hardness, HB	CV Instruments CV-3000LP8
Swelling 24 h, %	calipers
Water absorption 24h, %	calipers
Density profile	GreCon density Analyzer X-ray

Table 2. Selected mechanical and physical parameters of chipboard (Pfleiderer, DecoBoard P2 model)

Physical parameter	Value
Density, kg/m^3	670
Tensile strength, MPa	0.39
Swelling 24 h, %	61.8
Elasticity modulus, MPa	2950
Static bending strength, MPa	15.35

Obtained results were saved as a class label for the performed experiments. The registration was made at various degrees of wear:

- 4 times for the green state,
- 2 times for the yellow state,
- 3 times for the red state.

Two separate cards were used for acquisition, with different sampling speeds: National Instrument PCI-6111 and NIPCI-6034E. Tests were performed in accordance with CEN EN 310 (1994) and CEN EN 1534 (2020), using an Brinell CV 3000LDB tester (CV Instruments, Surrey, UK) as well as Instron 3382 testing machine (Norwood, MA, USA) respectively.

PC computer was used for the recording process, with National Instruments software, i.e. Lab ViewTM environment (National Instruments Corporation, ver. 2015 SP1, Austin, Texas, USA) using the NI PCI - 6034E and NI PCI – 6111 (Austin, Texas, USA) data acquisition cards. Two cards were used due to presence of signals with different frequency. To adequately record AE signal, card with high sampling frequency was required (2 MHz, measuring window of 0.3 s). Remaining signals were recorded at a frequency of 50 kHz, with 1.1s measuring window. The signals were connected to the cards separately for each frequency range, using BNC-2110 connection boxes.

Table 3. List of standards used in testing the properties of wood-based materials

Property	Norm
Static bending strength, MPa	PN-EN 310
Elasticity modulus, MPa	PN-EN 310
Screw retention, N/mm	PN-EN 320
Tensile strength, MPa	PN-EN 319
Swelling 24h, %	PN-EN 317
Water absorption 24h, %	PN-D-04234, PN-D-04213, PN-D-04213:1964
Sand content, %	ISO 3340 (PN-76/D-04245)

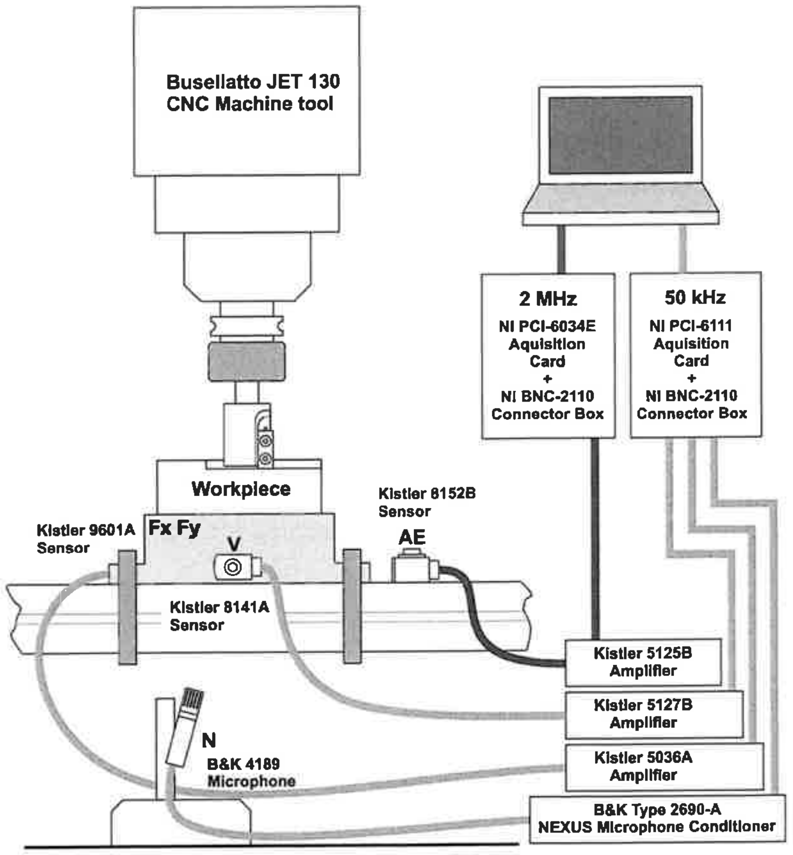


Figure 2. Outline of the measuring track used for the data collection during the machining process

In order to ensure minimal noise and changes in recorded signals, all sensors were kept in the same position in relation to the cutting zone and work-piece, through the entire measurement process. Outline of the measuring track is presented in Figure 2. The overall data structure of the obtained set is presented in Table 4.

Table 4. Data structure in the obtained dataset after recording all signals

Data set	Variable	Length of one trial	Sampling frequency [Hz]	Measure time [s]
DataHigh	acoustic emission	27,999,960	5,000,000	5.59
DataLow	force X	700,000	200,000	3.50
DataLow	force Y	700,000	200,000	3.50
DataLow	noise	700,000	200,000	3.50
DataLow	vibration	700,000	200,000	3.50
DataCurrent	device current	30,000	50,000	0.60
DataCurrent	device voltage	30,000	50,000	0.60
DataCurrent	head current	30,000	50,000	0.60
DataCurrent	head voltage	30,000	50,000	0.60
DataCurrent	servo current	30,000	50,000	0.60
DataCurrent	servo voltage	30,000	50,000	0.60

3. Data preparation

3.1. Downsampling based on reinterpolation

In the presented approach, the chosen input files were scalograms derived from the initial signals, represented as time series data. Due to large size of the more precisely measured signals, further usage can pose some problems. It was important to initially measure those elements with high accuracy, since using lower frequencies might have resulted in some errors and changes in the overall shape of signal curve. Therefore it was decided, that instead of lowering the initial quality of measured data, it will be reinterpolated before presenting it as an input for the multiple input CNN.

In order to retain the original signal shape, the entire length of it was divided into sections, and interpolated to feet in the given range. Since in presented experiments few signals of different lengths were recorded, they were all downsampled to the size of shortest signal (30,000). Figure 3 shows an example of originally registered signal, and its shape after the reinterpolation procedure. As can be seen, by using this method the general signal shape is retained, while the overall complexity is significantly reduced. Table 5 shows all original signal lengths, and final values after the downsampling procedure. Overview of the procedure used for this operation is presented in Algorithm 1.

```
Input: $n > 1$                                  /* desired length of the downsampled signal */
Input: signal                                                          /* original signal */
$oL \leftarrow length(signal)$                               /* length of the original signal */
$k \leftarrow linearspace(1, oL, n)$          /* generate new resolution for the downsampled signal */
$downsampled \leftarrow reinterpolation(1 : oL, signal, k)$         /* output - downsampled signal */
```

Algorithm 1. Finding the largest element

In the presented method linear space generates n points where spacing between the points equals $(oL-1)/(n-1)$. Reinterpolation function used is linear interpolation method defined as follows:

$$V_q = \text{reinterpolation}\ (X,\ V,\ X_q) \tag{1}$$

where: V_q – interpolated points used to find the underlying function $V = F(X)$ at the query points X_q, X – is a vector, V – is a vector with the same size as X, X_q is the same size as set of query points V_q.

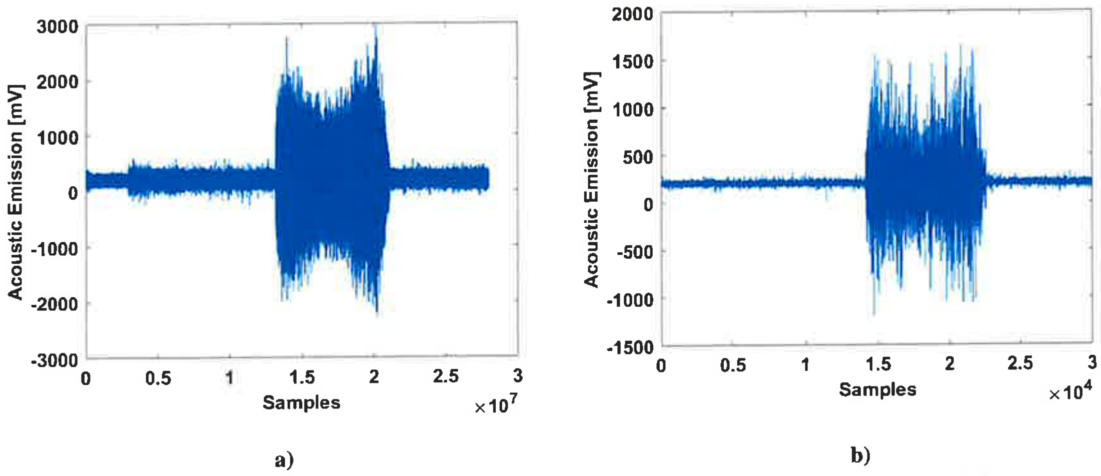


Figure 3. Original signal obtained during data acquisition (a), and a downsampled data used for scalogram generation (b); the general shape is retained

Table 5. The structure of the data variables in data sets

Data set	Signal	Length of one trial	Length of one trial after reinterpolation
DataHigh	acoustic emission	27,999,960	30,000
DataLow1	force X	700,000	30,000
DataLow2	force Y	700,000	30,000
DataLow3	noise	700,000	30,000
DataLow4	vibration	700,000	30,000
DataCurrent1	device current	30,000	30,000
DataCurrent2	device voltage	30,000	30,000
DataCurrent3	head current	30,000	30,000
DataCurrent4	head voltage	30,000	30,000
DataCurrent5	servo current	30,000	30,000
DataCurrent6	servo voltage	30,000	30,000

3.2. Scalogram generation

After the data reinterpolation process, all of the 75 data samples (each represented by 11 signals total), the signals are converted to scalograms. The images are then used as an input for the CNN network — initial layer has total of 11 inputs — one for each of the signals. Using scalograms images is a significant improvement, since it allows automatic feature extraction to take place. This removes the necessity for hand-crafting them.

A scalogram is a two-dimensional visual representation of a signal that provides information about its frequency and variation over time. It is obtained by applying the CWT to the signal in question. The scalogram serves as a powerful tool for the analysis of non-stationary signals, enabling researchers to identify and characterize time-varying frequency patterns in the data.

In a scalogram, the X-axis represents time, and the Y-axis represents the scale (which is inversely proportional to frequency). The colour or intensity at each point in the plot signifies the magnitude of the wavelet coefficients, providing a measure of the signal's energy distribution across different time scales and frequencies. By examining the scalogram, one can discern localized features such as transient events or frequency modulations, which may not be readily apparent in traditional time-domain or frequency-domain representations.

From the tool condition monitoring point of view it is an important factor. Changes in individual signals might not be noticeable enough or achieved class thresholds can differ between them, blurring the results clarity. Using scalograms not only provides a reasonable way for representing changes in time, but also can provide additional insight into them. For approaches such as CNN, it can lead to more accurate and stable results.

In order to generate scalograms as accurately as possible, CWT with filter bank is used. Default wavelet for this is set as analytic Morse (3, 60) wavelet. The Morse wavelet is a parametric family of continuous wavelets that are well-suited for the analysis of non-stationary signals due to their ability to adapt their time-frequency localization properties. The Morse wavelet is characterized by two parameters, the order (γ) and the symmetry (β). These parameters control the time-frequency localization, concentration of energy, and the number of oscillations in the wavelet. The time-bandwidth and symmetry parameters for the Morse wavelets can both be varied to tune it for the specific solution needs. Additionally, analytic Morlet (Gabor) wavelet (or bump wavelet) can be used.

One additional operation that is performed in order to optimize the solution is precomputing the filters. Using multiple input CNN, with set of 11 signals is a computationally costly operation, hence improving the process is necessary. CWT can use such previously prepared filters as input, reducing number of calculation that need to be performed on the fly. In turn, with filter bank used, the wavelets can be visualized in time and frequency domains. In order to gain additional insight, filter banks with specific frequency or period ranges, and measure with multiple 3dB bandwidths can also be created. To further improve obtained results, quality factor can be defined for the wavelets in the filter bank.

For the presented approach, the filters are normalized so that the peak magnitudes for all passbands are approximately equal to 2. The default filter bank is designed for a signal with 1024 samples and uses the analytic Morse (3, 60) wavelet. Additionally it uses the default scales: approximately 10 wavelet bandpass filters per octave (10 voices per octave). The highest-frequency passband is designed so that the magnitude falls to half of the peak value at the Nyquist frequency.

As implemented, the CWT uses L1 normalization. With L1 normalization, equal amplitude oscillatory components at different scales have equal magnitude in the CWT. It provides a more accurate representation of the signal. The amplitudes of the oscillatory components agree with the amplitudes of the corresponding wavelet coefficients. Making sure that the resulting signals are as close to the original as possible was one of the key points in the presented experiments. L1 normalization, along with the data preparation process made sure, that as much of the original information will be retained as possible.The general overview of the used scalogram generation method is presented in Algorithm 2.

```
Require: $f_s > 0$                                              ▷ sampling frequency of input signal
Require: signal                                                 ▷ original signal
  $n \leftarrow length(signal)$                                 ▷ length of the input signal
  $fb \leftarrow cwtfilterbank(n, fs)$                          ▷ generate a continuous wavelet transform (CWT) filter bank
  $[cfs, frq] \leftarrow wt(fb, signal)$                        ▷ generate continuous wavelet transform with filter bank
                                                                ▷ cfs - continuous wavelet transform (CWT) coefficients
                                                                ▷ frq - frequencies corresponding to the scales of cfs
  $t = (0 : n - 1)/f_s$                                         ▷ calculate time range based on sampling frequency
  $scalogram\_image \leftarrow pcolor(t, frq, abs(cfs))$        ▷ create scalogram in the image form.
```

Algorithm 2. Scalogram generation algorithm

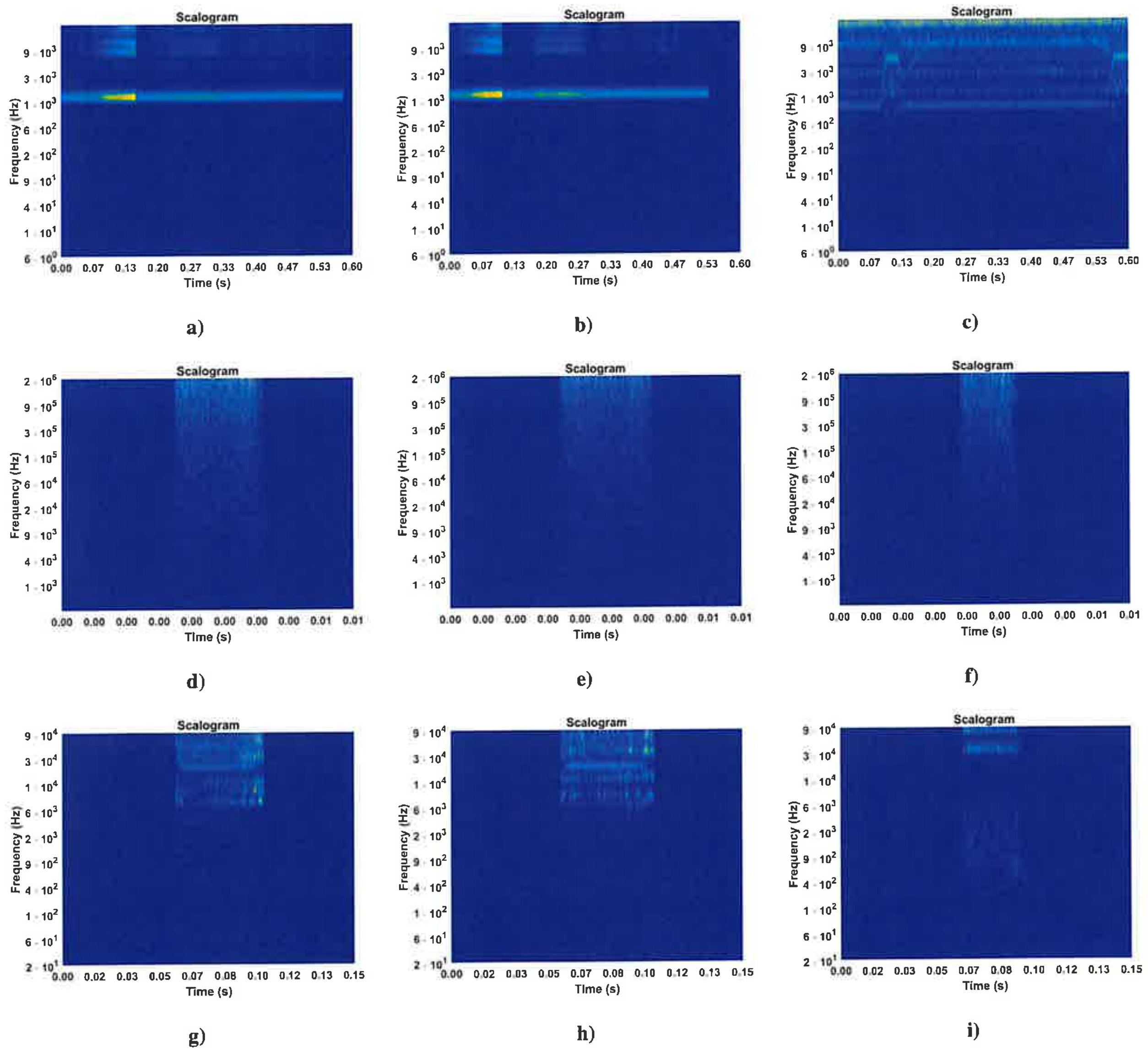


Figure 4. Exemplary scalograms generated for Curr1 (top row), High1 (middle row) and Low1 (bottom row) signals. For each scalogram, examples representing individual classes are shown in columns: a), d), g) – green, b), e), h) – yellow, c), f), i) – red

4. Methods

Previous approaches to tool wear classification show that even minimal adjustments can result in significant changes to the overall solution performance [14, 18, 19, 27]. Drawing from those experiences, the method proposed focuses on taking advantage of the strengths offered by different parts of the solution. For example, the use of signals ensures precise data, retaining information about the milling process and changing tool state. At the same time, this input in its unchanged form, introduces to much noise, blurring the borders between recognized classes. Using scalograms ensures, that as much of important information as possible will be retained in a form offering most advantages to the CNN. Finally, novel structure of the network in itself works well with the dataset prepared in this way.

4.1. Data sets for numerical experiments

The final dataset used for the network training consisted of 825 scalogram images, representing signals recorded during the milling process. The set was divided into training, validation and test sets, according to established practices in the artificial neural network training.

One important research goal presented in this paper was the incorporation of the overall manufacturer requirements in terms of solution performance. Key property in that case was reduction of critical errors between border classes:

- denoting example from green class as red, in which case a good tool will be discarded, resulting in unnecessary production downtime,
- classifying red instance as green, which can lead to poor product quality and financial loss associated with the need to discard such element.

Both cases are highly undesirable, and should be avoided.

4.2. Applying CNN for image classification

CNNs have emerged as a powerful tool for image classification tasks, with a profound impact in computer vision tasks. This class of deep learning models has demonstrated remarkable performance in classifying images by exploiting spatial, hierarchies and local connectivity patterns. CNNs employ a series of architectural components and techniques, to effectively learn and classify images. Such elements can include convolutional layers, activation functions, pooling layers, and fully connected layers.

The architecture of a CNN is typically composed of multiple layers arranged in a hierarchical manner. In this structure each layer performs a specific operation. The input image is processed sequentially through consecutive layers, ultimately resulting in a predicted class label. The main layers in a CNN are the convolutional layers, activation functions, pooling layers, and fully connected layers, which are interconnected to form a deep architecture. CNNs also employ batch normalization and dropout techniques to improve model stability and prevent overfitting.

A typical CNN architecture uses the following layers, arranged sequentially:

1. **Input layer.** The initial layer responsible for receiving raw image data in the form of a matrix with pixel values.

2. **Convolutional layers.** Layers performing the main computational operations by convolving the input image with learnable filters, detecting features such as edges, corners, and textures.
3. **Activation layers.** Following the convolutional layers, activation layers introduce non-linearity into the network by applying an activation function to the output of the convolution (such as the Rectified Linear Unit - ReLU).
4. **Pooling layers.** Layers performing the down-sampling operations to reduce the spatial dimensions of the feature maps, decreasing computational complexity and controlling overfitting.
5. **Fully connected layers.** Layers responsible for integrating high-level features extracted from the previous layers and making the final classification decision; they employ traditional feedforward neural network architecture and include an output layer with a Softmax activation function, generating class probabilities.

Additionally, key components of a CNN include:

- **Filters.** Also known as convolutional kernels; filters are learnable weight matrices that slide over the input image during the convolution operation; they are responsible for detecting specific patterns and features within the image.
- **Feature maps.** The output of the convolutional layers; feature maps represent the spatial arrangement of the detected features in the image.
- **Stride.** The step size by which filters slide over the input image during convolution, affecting the spatial dimensions of the resulting feature maps.
- **Padding.** The process of adding extra pixels around the input image before convolution, ensuring that the spatial dimensions of the feature maps are preserved.

Convolution used in CNN is a mathematical operation that combines the input image matrix and the filter matrix. It is performed by element-wise multiplication of the overlapping regions between the input and the filter, followed by summing up the results. This operation is repeated for each location in the input image, producing a feature map that highlights the presence of specific features.

During the training process, the CNN adjusts its filter weights to minimize the classification error. This is achieved through backpropagation - an algorithm that calculates the gradient of the loss function with respect to each weight by applying the chain rule. The gradients are then used to update the filter weights using optimization techniques. Used methods can include stochastic gradient descent (SGD) or more advanced methods like Adam.

To prevent overfitting and enhance generalization, CNNs employ regularization techniques such as L1 and L2 regularization, dropout, and batch normalization. Data augmentation is another approach to improving the model's performance. Such operations usually involve generation of new training samples by applying random transformations, such as rotation, scaling, or flipping, to the original images.

4.3. Multiple input CNN architecture

The multiple input CNN architecture proposed in this paper consist of 11 inputs. The layers are described as follows:

1. **ImageInputLayers** (11 inputs total). Architecture contains 11 ImageInputLayers, each accepting an input image of size 128×128×3, which means that each input image has a resolution of 128×128 pixels with 3 color channels (RGB).
2. **Convolution2DLayer** (11 layers). Each of the 11 ImageInputLayers is connected to its corresponding Convolution2DLayer with 64 filters of size 3x3x3. These layers perform the convolution operation on the input images, learning to extract relevant features from the input data.
3. **BatchNormalizationLayer** (11 layers). Each of the 11 Convolution2DLayer is connected to a BatchNormalizationLayer with 64 channels. These layers normalize the activations of the previous layer to stabilize the training process and improve convergence.
4. **ReLULayer** (11 layers). Each of the 11 BatchNormalizationLayers is connected to a ReLULayer. ReLU (Rectified Linear Unit) is an activation function that introduces non-linearity into the network by applying the function max(0, x) to the input, where x is the input value.
5. **MaxPooling2DLayer** (11 layers). Each of the 11 ReLULayers is connected to a MaxPooling2 DLayer with a stride of 1x1. Max-pooling reduces the spatial dimensions of the input by selecting the maximum value within a specified window size, which in this case is 1x1, meaning there is no reduction in spatial dimensions.
6. **DepthConcatenationLayer** (single layer). The 11 MaxPooling2DLayer outputs are concatenated along the depth dimension, forming a single tensor that is passed to the subsequent layers.
7. **Convolution2DLayer** (single layer). The DepthConcatenationLayer is connected to a Convolution2DLayer with 128 filters of size 3x3x704. This layer performs another round of feature extraction on the combined output from the previous layers.
8. **BatchNormalizationLayer** (single layer). This Convolution2DLayer is connected to a BatchNormalizationLayer with 128 channels, normalizing the activations before passing them to the next layer.

Table 6. Architecture of multiple inputs CNN model designed from scratch

No.	Layer	In	Out	Image size	Convolution	Stride padding	Batch norm
1	ImageInputLayer	0	1	128×128×3			
2	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
3	BatchNormalizationLayer	1	1				64
4	ReLULayer	1	1				
5	MaxPooling2DLayer	1	1			1×1/same	
6	ImageInputLayer	0	1	128×128×3			
7	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
8	BatchNormalizationLayer	1	1				64
9	ReLULayer	1	1				
10	MaxPooling2DLayer	1	1			1×1/same	
11	ImageInputLayer	0	1	128×128×3			
12	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
13	BatchNormalizationLayer	1	1				64
14	ReLULayer	1	1				
15	MaxPooling2DLayer	1	1			1×1/same	
16	ImageInputLayer	0	1	128×128×3			

No.	Layer	In	Out	Image size	Convolution	Stride padding	Batch norm
17	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
18	BatchNormalizationLayer	1	1				64
19	ReLULayer	1	1				
20	MaxPooling2DLayer	1	1			1×1/same	
21	ImageInputLayer	0	1	128×128×3			
22	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
23	BatchNormalizationLayer	1	1				64
24	ReLULayer	1	1				
25	MaxPooling2DLayer	1	1			1×1/same	
26	ImageInputLayer	0	1	128×128×3			
27	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
28	BatchNormalizationLayer	1	1				64
29	ReLULayer	1	1				
30	MaxPooling2DLayer	1	1			1×1/same	
31	ImageInputLayer	0	1	128×128×3			
32	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
33	BatchNormalizationLayer	1	1				64
34	ReLULayer	1	1				
35	MaxPooling2DLayer	1	1			1×1/same	
36	ImageInputLayer	0	1	128×128×3			
37	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
38	BatchNormalizationLayer	1	1				64
39	ReLULayer	1	1				
40	MaxPooling2DLayer	1	1			1×1/same	
41	ImageInputLayer	0	1	128×128×3			
42	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
43	BatchNormalizationLayer	1	1				64
44	ReLULayer	1	1				
45	MaxPooling2DLayer	1	1			1×1/same	
46	ImageInputLayer	0	1	128×128×3			
47	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
48	BatchNormalizationLayer	1	1				64
49	ReLULayer	1	1				
50	MaxPooling2DLayer	1	1			1×1/same	
51	ImageInputLayer	0	1	128×128×3			
52	Convolution2DLayer	1	1		64 3×3×3	1×1/same	
53	BatchNormalizationLayer	1	1				64
54	ReLULayer	1	1				
55	MaxPooling2DLayer	1	1			1×1/same	
56	DepthConcatenationLayer	11	1				
57	Convolution2DLayer	1	1		128 3×3×704	1×1/same	
58	BatchNormalizationLayer	1	1				128
59	ReLULayer	1	1				
60	FullyConnectedLayer	1	1	2097152××			
61	ReLULayer	1	1				
62	FullyConnectedLayer	1	1	128××			

No.	Layer	In	Out	Image size	Convolution	Stride padding	Batch norm
63	ReLULayer	1	1				
64	FullyConnectedLayer	1	1	64××			
65	ReLULayer	1	1				
66	FullyConnectedLayer	1	1	32××			
67	SoftmaxLayer	1	1				
68	ClassificationOutputLayer	1	0				

9. **ReLULayer** (single layer). This BatchNormalizationLayer is connected to a ReLULayer, introducing non-linearity into the network.
10. **FullyConnectedLayers** and ReLULayers (3 pairs). The architecture has three pairs of FullyConnectedLayers and ReLULayers with 128, 64, and 32 units, respectively. The FullyConnectedLayers enable the network to learn higher-level features and representations from the extracted features, while the ReLULayers introduce non-linearity.
11. **SoftmaxLayer**. The last FullyConnectedLayer is connected to a SoftmaxLayer with 3 classes. The SoftmaxLayer normalizes the input into a probability distribution over the 3 classes.
12. **ClassificationOutputLayer** (single layer). Finally, the architecture ends with a ClassificationOutputLayer, which computes the categorical cross-entropy loss for training and provides the final class predictions for the input images.

In summary, presented multiple input CNN architecture is a deep learning model with 11 input branches, each containing a series of Convolutional, Batch Normalization, ReLU, and MaxPooling layers. These branches are then combined using a DepthConcatenationLayer and followed by additional Convolutional, Batch Normalization, ReLU, FullyConnected, and Softmax layers to produce a final classification output. Overall network structure is visualized in Figure 5, while full outline of the used layers is presented in Table 6.

4.4. Improved approach for drill wear classification

As shown in previous research [17], connecting different classifiers into ensemble and using voting method to obtain final classification can improved overall results. Similar approach was used for final solution presented in this paper. While initial experiments obtained some acceptable results, there were still not satisfactory, and additional work to improve overall score was required. The evaluation of a machine learning model involves the partitioning of a dataset containing 75 samples into three distinct subsets: the training set, the validation set, and the test set. In this instance, the dataset is divided such that there are 60 samples for training, 10 samples for validation, and 5 samples for testing. Additionally, a k-fold cross-validation technique is employed, where $k = 15$.

The purpose of dividing the dataset into these three subsets is to ensure a rigorous and unbiased assessment of the model's performance. The training set is utilized for fitting the model, allowing it to learn patterns and relationships within the data. The validation set is used to tune the model's hyperparameters and to gauge its performance during the training process. This enables the identification of potential issues such as overfitting or underfitting, which can then be addressed before the final evaluation. Lastly, the test set serves as an independent subset of data that is employed to evaluate the model's performance,

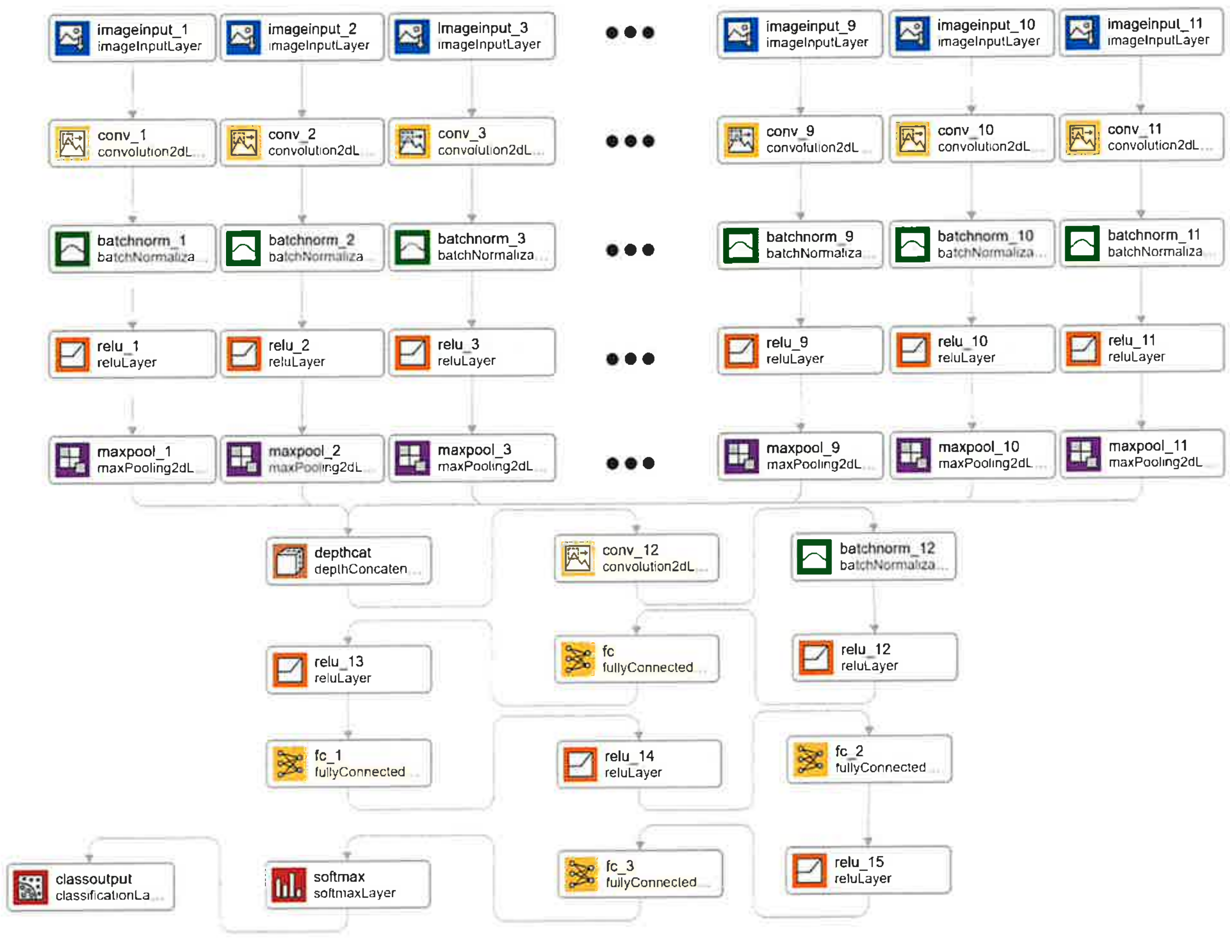

imageinput_1 imageInputLayer
imageinput_2 imageInputLayer
imageinput_3 imageInputLayer
imageinput_9 imageInputLayer
imageinput_10 imageInputLayer
imageinput_11 imageInputLayer
conv_1 convolution2dL...
conv_2 convolution2dL...
conv_3 convolution2dL...
conv_9 convolution2dL...
conv_10 convolution2dL...
conv_11 convolution2dL...
batchnorm_1 batchNormaliza...
batchnorm_2 batchNormaliza...
batchnorm_3 batchNormaliza...
batchnorm_9 batchNormaliza...
batchnorm_10 batchNormaliza...
batchnorm_11 batchNormaliza...
relu_1 reluLayer
relu_2 reluLayer
relu_3 reluLayer
relu_9 reluLayer
relu_10 reluLayer
relu_11 reluLayer
maxpool_1 maxPooling2dL...
maxpool_2 maxPooling2dL...
maxpool_3 maxPooling2dL...
maxpool_9 maxPooling2dL...
maxpool_10 maxPooling2dL...
maxpool_11 maxPooling2dL...
depthcat depthConcaten...
conv_12 convolution2dL...
batchnorm_12 batchNormaliza...
relu_13 reluLayer
fc fullyConnected...
relu_12 reluLayer
fc_1 fullyConnected...
relu_14 reluLayer
fc_2 fullyConnected...
classoutput classificationLa...
softmax softmaxLayer
fc_3 fullyConnected...
relu_15 reluLayer

Figure 5. Multiple input CNN architecture with 11 input image layers.

providing a reliable estimation of its generalization capabilities when applied to previously unseen data. The application of k-fold cross-validation is intended to further enhance the robustness of the evaluation process. This technique involves dividing the dataset into k equally sized partitions, where $k = 15$ in this case. The model is then trained and validated k times, with each partition serving as the validation set exactly once, while the remaining $k-1$ partitions are combined to form the training set. The average performance across all k iterations is calculated to yield a more reliable and stable performance metric. This process helps to minimize potential biases arising from the initial partitioning of the dataset and provides a more accurate assessment of the model's true performance.

To transition from the k-fold ($k = 15$) cross-validation to a confusion matrix, you must first complete the k-fold cross-validation process and then aggregate the predictions for each fold. The steps involved in that process are as follows:

1. Perform k-fold cross-validation:
   A. Divide the dataset into k equally sized partitions (folds).
   B. For each fold, train the model on the remaining $k-1$ folds combined as the training set, and validate the model on the current fold as the validation set.
   C. Store the predictions and true labels for each validation set.
2. Aggregate predictions:

   A. Combine the predictions and true labels from all k validation sets to form a single set of predictions and true labels.

3. Generate the confusion matrix:

   A. Compare the aggregated predictions to the true labels and create a matrix that represents the number of occurrences for each possible combination of predicted and true classes.

   B. The rows of the matrix represent the true classes, while the columns represent the predicted classes. Each cell in the matrix contains the count of instances where the model predicted a particular class (column) when the true class was another (row).

4. Interpret the confusion matrix:

   A. Diagonal elements represent correct predictions (true positives and true negatives), while off-diagonal elements represent incorrect predictions (false positives and false negatives).

   B. Analyse the confusion matrix to determine the model's performance metrics, such as accuracy, precision, recall, and F1-score.

5. Results and discussion

The results for the proposed approach were compared with previously prepared solutions, including one based on gradient boosting, extreme gradient boosting and random forest algorithms. The outline of the training progress of our CNN model over 200 iterations is presented in Figure 6. The top panel displays the model's accuracy as a percentage, while the bottom panel shows the loss. In the accuracy graph, the solid blue line indicates the mean accuracy across the training batches, with the shaded area representing one standard deviation from the mean. The dashed black line shows the validation accuracy. Notably, the validation accuracy closely follows the training accuracy, suggesting a good generalization of the model. The black dots indicate the epochs where the model achieved a new peak in validation accuracy. The final accuracy achieved by the model is denoted by the 'Final' marker at the end of the training process. The loss graph illustrates the decline in both training (orange line) and validation (black line) loss over time, which is a typical behavior of a converging model. The initial sharp decrease indicates rapid learning, which gradually stabilizes as the model optimizes. The black dots represent points of minimum validation loss, coinciding with the peaks in validation accuracy.

Obtained accuracy results for all methods, and final solution presented in this paper are presented in Table 7. Confusion matrix outlining the overall class predictions, with associated classification errors is shown in Figure 7. As can be seen, the presented approach performed well both in terms of overall accuracy – achieving highest score from all of the methods – and overall classification. Total of 3 instances were misclassified: 2 times example from red class was classified as yellow, and once yellow class was classified as red. It is important to note, that there are no instanced of red-green or green-red misclassifications (the most influential ones in terms of tool condition monitoring).

The overall performance of the multiple inputs CNN model is summarized in the Table 8 The model achieved an impressive overall accuracy of 96.00%. This high accuracy indicates that the model is highly effective in classifying the tool condition into the correct categories (green, yellow, red) based on the input scalogram images derived from the milling process signals. Furthermore, the precision (macro average)

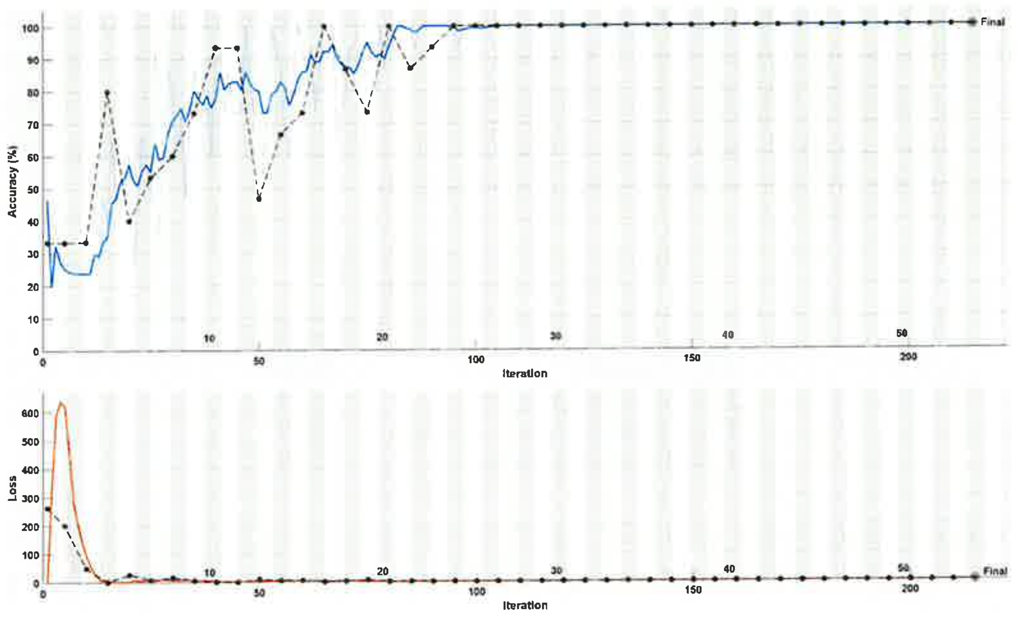


Figure 6. Outline of the training process for the proposed multiple input CNN architecture

Table 7. Final classification results with previously prepared approaches

Mode	Parameters	Accuracy [%]
Multiple inputs CNN	269.2M total learnables	96.00

Extreme gradient boosting	learning_rate = 0.1
	n_estimators = 100
		93.33

Random forest	n_estimators = 100
	min_samples_split = 2
	min_samples_leaf = 1
		86.66

Gradient boosting	min_samples_split = 2
	min_samples_leaf = 1
		86.66

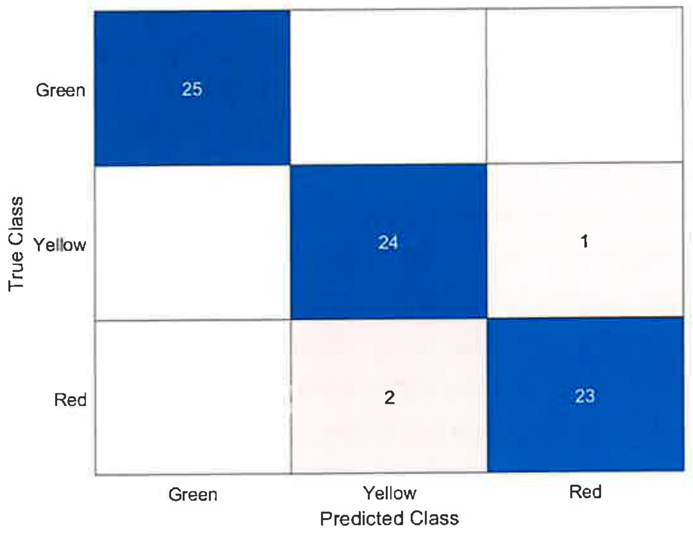


Figure 7. Confusion matrix outlining classification results for the proposed solution

of the model is 96.05%, suggesting that the model has a high level of reliability in its predictions. The recall (sensitivity, macro average) stands at 96.00%, which means that the model is capable of correctly identifying the majority of the relevant instances across all classes. The F1 score (macro average), which is a balance between precision and recall, is calculated to be 95.99%. This high F1 score underscores the model's balanced performance in both precision and sensitivity.

Table 8. Overall classification metrics for multiple inputs CNN model

Metric	Value [%]
Overall accuracy	96.00
Precision (macro average)	96.05
Recall (sensitivity, macro average)	96.00
F1 Score (macro average)	95.99

A more detailed insight into the model's performance is provided by the class-wise classification metrics, as presented in the Table 9. This analysis allows us to understand how the model performs for each specific class.

Table 9. Class-wise classification metrics for multiple inputs CNN model [%]

Class	Precision	Sensitivity	Specificity	F1 score
Green	100.00	100.00	100.00	100.00
Yellow	92.31	96.00	96.00	94.12
Red	95.83	92.00	98.00	93.88

Green class. For the green class, which represents tools in good condition, the model achieved 100% precision, sensitivity, and F1 score. This result indicates a perfect classification performance for this class, with no misclassifications.

Yellow class. The model showed a precision of 92.31% and a sensitivity of 96.00% for the yellow class, indicating tools in an intermediate state. The slightly lower precision suggests a few instances of over-predicting the yellow class, but a high sensitivity indicates a strong ability to correctly identify most of the yellow class instances. The F1 score for this class is 94.12%.

Red class. For the red class, indicating tools that need to be exchanged due to high wear, the model achieved a precision of 95.83% and a sensitivity of 92.00%. The high precision shows the model's ability to correctly identify the red class instances with minimal false positives, while the slightly lower sensitivity indicates some misses in identifying all the red class instances. The F1 score for the red class is 93.88%.

The results of the multiple inputs CNN model demonstrate its effectiveness in tool condition monitoring in a milling process. The high overall accuracy and balanced precision and recall across all classes indicate that the model is robust and reliable. Particularly noteworthy is the model's exceptional performance in classifying the green class without any errors, which is crucial for avoiding unnecessary tool changes and associated downtime. The slight variations in precision and sensitivity for the yellow and red classes suggest areas for further refinement. However, these results are still highly promising, indicating that the model can successfully differentiate between the varying degrees of tool wear, which is essential for effective tool maintenance and cost reduction.

In conclusion, the multiple inputs CNN model exhibits a strong potential for practical applications in tool condition monitoring. Future work may focus on further optimization of the model and expanding the dataset to enhance the model's generalization capabilities.

6. Conclusions

In this study, we introduced a multiple input CNN architecture, tailored for tool state recognition in milling processes. The development and implementation of this model have led to several significant findings and advancements in the field.

Foremost among these is the model's exceptional accuracy in classifying tool wear. It achieved an overall accuracy of 96.00%, a notable improvement over conventional methods. This high degree of accuracy not only demonstrates the model's robustness but also its reliability for practical applications in industrial settings.

The model's performance in classifying the state of tools was particularly remarkable. For tools in good condition (green class), it achieved perfect scores in precision, sensitivity, and F1 score. When identifying tools in an intermediate state (yellow class), the model showed a precision of 92.31% and a sensitivity of 96.00%. In the critical red class, indicative of tools requiring replacement, the model's precision and sensitivity were 95.83% and 92.00%, respectively. These results highlight the model's adeptness at distinguishing between various levels of tool wear with a high degree of accuracy.

Another key aspect of our study was the effective use of scalogram images derived from time-series signals. This approach allowed the CNN to extract detailed and nuanced features, facilitating the identification of complex patterns indicative of tool wear. Such a method is evidence of the significant potential of deep learning techniques in industrial and manufacturing applications.

The ability of the model to minimize errors in tool wear categorization is particularly beneficial for its application in real-world scenarios. This precision is vital in reducing unnecessary tool changes, optimizing production efficiency, and ensuring the quality of the final product.

However, our study also identified areas for potential improvement, especially in the yellow and red classes, where there were slight variations in precision and sensitivity. This finding provides a clear direction for future research and efforts to optimize the model.

In conclusion, the multiple input CNN model showcased in this study represents a significant step forward in the realm of tool condition monitoring. Its high accuracy and nuanced class-specific performance have the potential to revolutionize milling processes by enhancing tool maintenance, reducing operational costs, and ensuring product quality. Future research will aim to refine this model further, expand the dataset for more comprehensive training, and explore its application in various other industrial contexts.

References

[1] Bengio, Y. Learning deep architectures for AI. *Foundations and trends® in Machine Learning 2*, 1 (2009), 1–127.

[2] Choudhary, A., Mishra, R. K., Fatima, S., and Panigrahi, B. K. Multi-input CNN based vibro-acoustic fusion for accurate fault diagnosis of induction motor. *Engineering Applications of Artificial Intelligence 120* (2023), 105872.

[3] Demir, F., Turkoglu, M., Aslan, M., and Sengur, A. A new pyramidal concatenated CNN approach for environmental sound classification. *Applied Acoustics 170* (2020), 107520.

[4] Deng, L., and Yu, D. Deep learning: methods and applications. *Foundations and trends® in signal processing 7*, 3–4 (2014), 197–387.

[5] GOODFELLOW, I., BENGIO, Y., AND COURVILLE, A. *Deep learning*. MIT Press, 2016.

[6] HU, J., SONG, W., ZHANG, W., ZHAO, Y., AND YILMAZ, A. Deep learning for use in lumber classification tasks. *Wood Science and Technology 53*, 2 (2019), 505–517.

[7] IBRAHIM, I., KHAIRUDDIN, A. S. M., ABU TALIP, M. S., AROF, H., AND YUSOF, R. Tree species recognition system based on macroscopic image analysis. *Wood Science and Technology 51* (2017), 431–444.

[8] ISKRA, P., AND HERNÁNDEZ, R. E. Toward a process monitoring and control of a CNC wood router: Development of an adaptive control system for routing white birch. *Wood and Fiber Science 42*, 4 (2010), 523–535.

[9] JEGOROWA, A., GÓRSKI, J., KUREK, J., AND KRUK, M. Use of nearest neighbors (k-NN) algorithm in tool condition identification in the case of drilling in melamine faced particleboard. *Maderas: Ciencia y Tecnologia 22*, 2 (2020), 189 – 196.

[10] JEGOROWA, A., KUREK, J., ANTONIUK, I., DOŁOWA, W., BUKOWSKI, M., AND CZARNIAK, P. Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. *Wood Science and Technology 55*, 1 (2021), 271–293.

[11] JEMIELNIAK, K., URBAŃSKI, T., KOSSAKOWSKA, J., AND BOMBIŃSKI, S. Tool condition monitoring based on numerous signal features. *The International Journal of Advanced Manufacturing Technology 59*, 1-4 (2012), 73–81.

[12] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM 60*, 6 (2017), 84–90.

[13] KUO, R. J. Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. *Engineering Applications of Artificial Intelligence 13*, 3 (2000), 249–261.

[14] KUREK, J., ANTONIUK, I., GÓRSKI, J., JEGOROWA, A., ŚWIDERSKI, B., KRUK, M., WIECZOREK, G., PACH, J., ORŁOWSKI, A., AND ALEKSIEJUK-GAWRON, J. Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network. *Machine Graphics &Vision 28*, 1/4 (2019), 13–23.

[15] KUREK, J., ANTONIUK, I., GÓRSKI, J., JEGOROWA, A., ŚWIDERSKI, B., KRUK, M., WIECZOREK, G., PACH, J., ORŁOWSKI, A., AND ALEKSIEJUK-GAWRON, J. Data augmentation techniques for transfer learning improvement in drill wear classification using convolutional neural network. *Machine Graphics &Vision 28*, 1/4 (2019), 3–12.

[16] KUREK, J., KRUK, OSOWSKI, S., M., HOSER, P., WIECZOREK, G., JEGOROWA, GÓRSKI, J., A., WILKOWSKI, J., ŚMIETAŃSKA, K., AND KOSSAKOWSKA, J. Developing automatic recognition system of drill wear in standard laminated chipboard drilling process. *Bulletin of the Polish Academy of Sciences: Technical Sciences 64*, 3 (2016), 633–640.

[17] KUREK, J., KRUPA, A., ANTONIUK, I., AKHMET, A., ABDIOMAR, U., BUKOWSKI, M., AND SZYMANOWSKI, K. Improved drill state recognition during milling process using artificial intelligence. *Sensors 23*, 1 (2023), 448.

[18] KUREK, J., ŚWIDERSKI, B., JEGOROWA, A., KRUK, M., AND OSOWSKI, S. Deep learning in assessment of drill condition on the basis of images of drilled holes. In *Eighth International Conference on Graphic and Image Processing (ICGIP 2016) 29-31 October 2016, Tokyo, Japan (2017)* (Bellingham, 2017), T. Pham, V.Vozenilek and Z. Zeng, Eds., vol. 10225, SPIE, pp. 375–381.

[19] KUREK, J., WIECZOREK, G., ŚWIDERSKI, B., KRUK, M., JEGOROWA, A., AND OSOWSKI, S. Transfer learning in recognition of drill wear using convolutional neural network. In *2017 18th International Conference on Computational Problems of Electrical Engineering (CPEE) 11-13 September 2017, Kutna Hora, Czech Republic* (2017), IEEE, pp. 1–4.

[20] LEMASTER, R. L., LU, L., AND JACKSON, S. The use of process monitoring techniques on a CNC wood. Part 1. Sensor selection. *Forest Products Journal 50*, 7/8 (2000), 31–38.

[21] LEMASTER, R. L., LU, L., AND JACKSON, S. The use of process monitoring techniques on a CNC wood router. Part 2. Use of a vibration accelerometer to monitor tool wear and workpiece quality. *Forest Products Journal 50*, 9 (2000), 59-64.

[22] LIN, K. K.-Y. (2020) Github repository for AlexNet model (accessed on 5 August 2023).

[23] PANDA, S. S., SINGH, A. K., CHAKRABORTY, D., AND PAL, S. K. Drill wear monitoring using back propagation neural network. *Journal of Materials Processing Technology 172*, 2 (2006), 283–290.

[24] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M., BERG A. C. AND FEI-FEI L. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision 115*, 3 (2015), 211–252.

[25] SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural Networks 61* (2015), 85–117.

[26] STANFORD VISION LAB, STANFORD UNIVERSITY, PRINCTON UNIVERSITY (2020) ImageNet web page (accessed on 5 August 2023).

[27] ŚWIDERSKI, B., ANTONIUK, I., KUREK, J., BUKOWSKI, M., GÓRSKI, J., AND JEGOROWA, A. Tool condition monitoring for the chipboard drilling process using automatic, signal-based tool state evaluation. *BioResources 17*, 3 (2022), 5349–5371.

[28] ŚWIDERSKI, B., KUREK, J., OSOWSKI, S., KRUK, M., AND JEGOROWA, A. Diagnostic system of drill condition in laminated chipboard drilling process. In *21st International Conference on Circuits, Systems, Communications and Computers (CSCC 2017) Agia Pelagia Beach, Heraklion, Crete, Greece, July 14-17, 2017)*, N. Mastorakis, V. Mladenov and A. Bulucea, Eds., vol. 125 of *MATEC Web of Conferences*, EDP Sciences, 04002.

[29] SZWAJKA, K., AND TRZEPIECIŃSKI, T. Effect of tool material on tool wear and delamination during machining of particleboard. *Journal of Wood Science 62*, 4 (2016), 305–315.

[30] WEI, W., LI, Y., XUE, T., TAO, S., MEI, C., ZHOU, W., WANG, J., AND WANG, T. The research progress of machining mechanisms in milling wood-based materials. *BioResources 13*, 1 (2018), 2139–2149.

[31] WILKOWSKI, J., AND GÓRSKI, J. Vibro-acoustic signals as a source of information about tool wear during laminated chipboard milling. *Wood Research 56*, 1 (2011), 57–66.

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
M. Bukowski, I. Antoniuk, K. Szymanowski, A. Krupa, J. Kurek. Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024: 34(3), 41-60. DOI 10.37190/ord240303
mój indywidualny udział w jej powstaniu polegał na:

1. Koncepcji.
2. Metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.

[illegible]
Podpis

Warszawa, 27-01-2025

Izabella Antoniuk
izabella_antoniuk@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
M. Bukowski, I. Antoniuk, K. Szymanowski, A. Krupa, J. Kurek. Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024: 34(3), 41-60. DOI 10.37190/ord240303
mój indywidualny udział w jej powstaniu polegał na:

1. Przygotowaniu pierwszego szkicu.

[illegible]
Podpis

Warszawa, 27-01-2025

Karol Szymanowski
karol_szymanowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
M. Bukowski, I. Antoniuk, K. Szymanowski, A. Krupa, J. Kurek. Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024: 34(3), 41-60. DOI 10.37190/ord240303
mój indywidualny udział w jej powstaniu polegał na:

1. Akwizycji danych.
2. Zarządzaniu danymi.

[illegible]

Podpis

Warszawa, 27-01-2025

Artur Krupa
artur_krupa@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
M. Bukowski, I. Antoniuk, K. Szymanowski, A. Krupa, J. Kurek. Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024: 34(3), 41-60. DOI 10.37190/ord240303
mój indywidualny udział w jej powstaniu polegał na:

1. Edycji i korekcji artykułu.

Artur Krupa
Podpis

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
M. Bukowski, I. Antoniuk, K. Szymanowski, A. Krupa, J. Kurek. Multiple input CNN architecture for tool state recognition in the milling process based on time series signals. Operations Research and Decisions 2024: 34(3), 41-60. DOI 10.37190/ord240303
mój indywidualny udział w jej powstaniu polegał na:

1. Nadzorze.
2. Administracji.
3. Walidacji.

[illegible]
Podpis

10.5. Publikacja 5

PUBLIKACJA 5

A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard

Michał Bukowski, Albina Jegorowa, Jarosław Kurek

Przeglad Elektrotechniczny. 2024 May 1;2024(5).

https://doi.org/10.15199/48.2024.05.52

Punktacja: 70, IF: 0

1. Michał BUKOWSKI[1], 2. Albina JEGOROWA[2], 3. Jarosław KUREK[1]

Department of Artificial Intelligence, Institute of Information Technology, Warsaw University of Life Sciences (1),
Department of Mechanical Processing of Wood, Institute of Wood Sciences and Furniture, Warsaw University of Life Sciences (2)
ORCID: 1. 0000-0003-1567-879X; 2. 0000-0002-8935-845X, 3. 0000-0002-2789-4732

doi:10.15199/48.2024.05.52

A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard

Streszczenie. *Artykuł ten przedstawia szczegółową ocenę wydajności różnych architektur sztucznej inteligencji do klasyfikacji otworów wiertniczych w płytach wiórowych laminowanych. Badanie obejmuje własną sieć neuronową konwolucyjną (CNN), pięciokrotną sieć CNN, VGG19, pojedyncze i pięciokrotne VGG16, zespół sieci CNN, VGG19 i 5xVGG16, oraz transformery wizyjne (ViT). Wydajność każdego modelu mierzono i porównywano na podstawie dokładności klasyfikacji. Modele transformatorów wizyjnych, szczególnie model B_32 trenowany przez 8000 epok, wykazały wyższą skuteczność, osiągając dokładność 71.14%. Pomimo tego osiągnięcia, badanie podkreśla potrzebę równoważenia wydajności modelu z innymi aspektami, takimi jak zasoby obliczeniowe, złożoność modelu i czas szkolenia. Wyniki zwracają uwagę na znaczenie starannego doboru i dopracowania modelu, kierując się nie tylko wskaźnikami wydajności, ale także konkretnymi wymaganiami i ograniczeniami zadania i kontekstu. Studium stanowi solidną podstawę do dalszych badań nad innymi modelami opartymi na transformatorach oraz zachęca do głębszych badań nad dopracowaniem modeli w celu w pełni wykorzystania potencjału tych architektur SI w zadaniach klasyfikacji obrazów.* ***(Nowatorskie podejście z wykorzystaniem transformatorów wizyjnych (VIT) do klasyfikacji otworów wierconych w płytach wiórowych pokrytych melaminą)***

Abstract*. This paper presents a comprehensive performance evaluation of various AI architectures for a classification of holes drilled in melamine faced chipboard, including custom Convolutional Neural Network (CNN-designed), five-fold CNN-designed, VGG19, single and five-fold VGG16, an ensemble of CNN-designed, VGG19, and 5xVGG16, and Vision Transformers (ViT). Each model's performance was measured and compared based on their classification accuracy, with the Vision Transformer models, particularly the B_32 model trained for 8000 epochs, demonstrating superior performance with an accuracy of 71.14%. Despite this achievement, the study underscores the need to balance model performance with other considerations such as computational resources, model complexity, and training times. The results highlight the importance of careful model selection and fine-tuning, guided not only by performance metrics but also by the specific requirements and constraints of the task and context. The study provides a strong foundation for further exploration into other transformer-based models and encourages deeper investigations into model fine-tuning to harness the full potential of these AI architectures for image classification tasks.*

Słowa kluczowe: Vision Transformer, Convolutional Neural Network, monitorowanie stanu narzędzia, płyta wiórowa laminowana
Keywords: Vision Transformer, Convolutional Neural Network, tool state monitoring, melamine faced chipboard

Introduction

The process of manufacturing furniture involves intricate and precision-demanding steps. One such critical stage is drilling holes in melamine faced chipboard, where errors can lead to significant financial losses due to reduced product quality. Traditionally, the condition of the drill is manually monitored to identify the optimal moment for replacement, thus ensuring consistently high product quality. While manual monitoring offers some control, it is not sufficiently efficient. As such, there is a pressing need for a more automated, accurate, and efficient solution.

In the pursuit of this solution, tool condition monitoring (TCM) methodologies have been developed to evaluate and assess the state of various utensils, including drills. Such methods often require a multitude of diverse sensors to collect data, which is subsequently used to diagnose the drill's condition [7]. While these approaches can produce accurate results, they often necessitate extensive preprocessing, and mistakes at any stage can compromise the final result. Moreover, these solutions can be costly and complex to implement and maintain. Despite the advanced features generated from the vast array of registered signals, the accuracy of such solutions rarely surpasses 90% [3], [4].

The incorporation of machine learning algorithms into the wood industry is a growing trend. For example, algorithms have been developed to recognize wood species based on macroscopic texture images [2]. When using image-based samples, convolutional neural networks (CNN) are often applied [1], [5], [6], [8], [9], [17], [18]. However, their application often encounters obstacles such as the requirement of large datasets and the need for close cooperation with manufacturers to ensure task specificity, among other factors. Considering these limitations, this work presents a novel approach applying Vision Transformers for classifying holes drilled in melamine faced chipboard. This approach attempts to mitigate the complexities and enhance the adaptability of the system to specific manufacturing requirements. The primary enhancement lies in the elimination of complex equipment, reducing the requirement to a camera that captures images of the drilled holes. These images form the basis for assessing the drill's condition. Prior research, which tested various algorithms such as CNNs, transfer learning, and data augmentation methodologies, confirmed that this approach can accurately predict the state of the drill based solely on images, while improving overall prediction accuracy [5], [6], [8], [9]. Given the state of the art in the field of artificial intelligence, this paper makes use of Vision Transformers (ViT), a revolutionary architecture that is currently considered a benchmark in the field. Unlike traditional convolutional neural networks, which process image data in a local and hierarchical manner, Vision Transformers treat image data as a sequence of patches and leverage self-attention mechanisms to capture global dependencies. This technique has shown unprecedented success in various image classification tasks, outperforming established CNN architectures on multiple benchmarks. In this paper, we demonstrate the applicability and effectiveness of Vision Transformers in the context of classifying holes drilled in melamine faced chipboard, aiming to further advance tool condition monitoring practices.

Data Set

The dataset comprises images of holes drilled during the experiment. The images were collected in collaboration with the Institute of Wood Sciences and Furniture at the Warsaw University of Life Sciences. A standard CNC vertical machining centre, Busellato Jet 100, Thiene, Italy, was used for the drilling process. The material drilled was a standard laminated chipboard (U511SM – Swiss Krono 88 Group), typically used in the furniture industry, with dimensions of 2500x300x18. A 12mm Faba WP-01 drill with a tungsten carbide tip was utilized.

Five different drills were used during the drilling process. Each drill underwent cycles of operation, and the external corner wear parameter was monitored between cycles. This allowed for assigning appropriate classes (Green, Yellow, Red) to the obtained images based on the level of drill wear. Images produced by each drill were stored separately, preserving the order of creation to reflect the gradual deterioration of drill condition. This could serve as additional information during the learning process. Table 1 provides details about the data acquisition process and final corner wear measurements for each drill at the end of the last drilling cycle.

Table 1. Sample counts for each class before and after data augmentation. Values in each cell are presented in following order: Green, Yellow, Red.

No. of drill	Original	Augmented	Total
1	840/420/406	840/840/840	2,520
2	840/700/280	840/840/840	2,520
3	700/560/420	700/700/700	2,100
4	840/560/280	840/840/840	2,520
5	560/560/560	560/560/560	1,680

Mentioned three classes correspond to the condition of the drill used to make the holes:

- Green class: good condition, where the drill is new and not yet worn, can be further used.
- Yellow class: worn condition, where the drill is used and may require manual evaluation to determine if it is still good enough for production.
- Red class: requiring replacement, where the drill is used to a point of being unusable and should be replaced immediately.

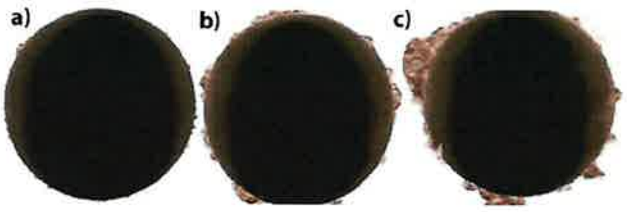


Figure.1. Evaluation of drill wear - Three classes of drill hole conditions in laminated chipboard: a) green class: hole made by a new, unworn drill, b) yellow class: hole made by a used drill, requiring manual evaluation for further use, c) red class: hole made by a drill that is too worn for use and needs immediate replacement.

Vision Transformer model

Convolutional Neural Networks (CNNs) have traditionally been the go-to models for computer vision tasks due to their ability to exploit local correlations in data, thanks to their convolutional nature. However, their efficiency can be undermined when the tasks require understanding the global context of an image or when the data lacks the aforementioned local correlation, due to which the CNN's hierarchical structure might introduce unnecessary inductive bias.

The Vision Transformer (ViT) model, introduced by Google Research [10]-[16], presents a departure from the typical CNNs by adopting a transformer architecture, originally de- signed for natural language processing tasks. Transformers rely on self-attention mechanisms, providing a better under- standing of global context as every part of the input contributes to the final representation of every other part.

Advantages of ViT:

ViTs possess several advantages over CNNs:

- Global Context Understanding: ViTs are more capable of understanding the global context in images as they are not restricted to local features.
- Parameter Efficiency: ViTs often require fewer parameters than CNNs for similar performance, which can lead to more efficient models.
- Transfer Learning: ViTs can leverage the benefits of transfer learning better, due to their capability to learn from both vision and language domains.

Disadvantages of ViT:

Despite their advantages, ViTs also have some limitations:

- Computational Requirements: ViTs are often more computationally intensive than CNNs, especially for larger inputs. This might be a limiting factor for real-time applications or for deployments on devices with limited computational power.
- Training Data: ViTs typically require more training data to outperform CNNs. This might be a constraint when only limited labeled data are available.

In consideration of these factors, the decision to utilize ViT for our study was informed by the global nature of our task, where understanding the entire context of each image is crucial. Furthermore, given that we have a sufficiently large dataset for training, the advantages of using ViT outweigh the potential drawbacks.

Numerical Experiments

We used transfer learning on the Vision Transformer (ViT) models, utilizing a range of pretrained models: R_Ti _16, S_32, B_32, R26_S_32, B_16, and S_16 [10]-[16].

The performance of each pretrained model was assessed in terms of mean accuracy, standard deviation of accuracy, and model size. The models were trained at different learning rates and over different numbers of epochs. The complete results of these experiments are presented in Table II.

Upon analysis of the experimental results, it can be observed that the model B 32, with a learning rate of 0.003 and trained over 8000 epochs, achieved the highest mean accuracy of 71.14% with a standard deviation of 0.35%. This model also had a size of 398 MiB.

When considering the balance between computational re- sources (model size and training time) and performance, the S 16 model also demonstrated impressive results. It achieved a mean accuracy of 70.54% with a standard deviation of 0.47%, trained with a learning rate of 0.01 over 4000 epochs, with a comparatively smaller model size of 115 MiB.

These experiments highlight the impact of different training parameters and model architectures on the performance of ViT models for the given task.

Table 2. Results of numerical experiments: Comparative Performance Metrics of Pretrained ViT Models.

Pretrained model name	Learning rate	Epochs	Mean Acc	Std Acc	Model Size
R_Ti_16	0.003	500	65.68%	1.14%	40 MiB
R_Ti_16	0.003	1000	64.90%	1.13%	40 MiB
R_Ti_16	0.003	2000	67.16%	0.35%	40 MiB
S_32	0.003	500	64.54%	1.21%	118 MiB
S_32	0.003	1000	68.06%	0.25%	118 MiB
B_32	0.003	500	65.60%	0.87%	398 MiB
B_32	0.003	1000	68.04%	0.98%	398 MiB
B_32	0.01	2000	68.55%	0.32%	398 MiB
B_32	0.003	4000	70.96%	0.22%	398 MiB
B_32	**0.003**	**8000**	**71.14%**	**0.35%**	**398 MiB**
R26_S_32	0.003	500	65.08%	0.97%	170 MiB
R26_S_32	0.003	1000	67.99%	0.71%	170 MiB
B_16	0.003	500	67.14%	1.52%	391 MiB
B_16	0.003	1000	68.69%	0.44%	391 MiB

B_16	0.01	1000	69.18%	0.53%	391 MiB
B_16	0.003	2000	70.00%	0.48%	391 MiB
S_16	0.003	500	66.71%	0.32%	115 MiB
S_16	0.003	1000	68.49%	0.27%	115 MiB
S_16	0.01	2000	69.83%	0.17%	115 MiB
S_16	0.03	2000	68.72%	0.95%	115 MiB
S_16	0.003	4000	69.54%	0.41%	116 MiB
S_16	0.01	4000	70.54%	0.47%	115 MiB

Numerical Experiments Using Different AI Architectures

The dataset comprises images of holes drilled during the experiment. The images were collected in collaboration with the Institute of Wood Sciences and Furniture at the Warsaw University of A wide array of machine learning models, including various artificial intelligence architectures, were applied in the numer- ical experiments conducted during this study. The primary fo- cus was on the assessment of their ability to perform accurate classifications and the results are compiled in Table 3.

The models under investigation included a self-designed Convolutional Neural Network (CNN-designed), five-fold implementation of the CNN-designed model (5xCNN-designed), VGG19, single and five-fold VGG16, an ensemble of CNN-designed, VGG19, and 5xVGG16 models, and the Vision Transformer (ViT) models.

The CNN-designed model and its five-fold counterpart were developed specifically for this task, adopting unique design principles derived from the nature of the data and the specifics of the classification problem.

The VGG models were implemented following their original design principles but were fine-tuned to the task at hand. Both the single and the five-fold VGG16 models were used, and an ensemble model was also implemented for a more diversified approach.

The Vision Transformer (ViT) models, a more recent development in the field of AI, were also used in the experiments. These models are based on the transformer architecture, which has shown exceptional performance in various machine learning tasks. Various configurations and training epochs of the ViT models were used in the numerical experiments.

The parameters and configurations for each model were meticulously selected and fine-tuned during preliminary testing and validation stages to optimize performance. The models were then trained on the same dataset to ensure a fair comparison of their performance.

Following the training phase, the models were evaluated on a test set, and their performance was assessed based on classification accuracy. The results of these experiments provide an in-depth understanding of how each model performs, allowing for a comprehensive comparison of different AI architectures.

Table 3. Classification results for chosen algorithms.

#	Model	Accuracy
1	CNN-designed	69.78%
2	5xCNN-designed	67.35%
3	VGG19	66.77%
4	5xVGG16	67.13%
5	10xVGG16	66.98%
6	Ensemble (1,3,4)	69.26%
7	**Vision Transformers**	**71.14%**

Discussion

The experimental results achieved during the performance evaluation phase offer several insightful takeaways, especially when the performance of Vision Transformer (ViT) models is juxtaposed with the performances of other AI modelling approaches. The performance comparison is presented in Table 3.

Considering the highest accuracy, ViT models outperform the rest of the models with an accuracy of 71.14%. Specifically, the B_32 model trained for 8000 epochs demonstrated the best performance among the pre-trained models used in this study. The ensemble approach combining CNN-designed, VGG19, and 5xVGG16 also resulted in high accuracy, but it was still lower than the top-performing ViT model.

The comparison also illustrates that the CNN-designed model and its five-fold implementation rendered decent performances with accuracies of 69.78% and 67.35% respectively. Nevertheless, their performances were not up to par with the B_ 32 ViT model.

For the VGG models, both single and five-fold VGG16 delivered similar results, and the performance was further improved when they were used in an ensemble with CNN-designed model, yet the results were less satisfactory than the ones obtained using ViT models.

This comparative evaluation thus accentuates the superior capability of ViT models for the task in focus. Despite using other high-performing AI models like CNN and VGG, the study confirmed the efficiency and effectiveness of pre-trained ViT models, especially with a large number of training epochs. In conclusion, while ViT models have proven to be a robust solution in this context, their use should be carefully assessed based on the available resources and the specific requirements of the task at hand.

Conclusion

This study presented an in-depth comparison of several AI architectures, including a custom Convolutional Neural Network (CNN-designed), five-fold CNN-designed, VGG19, single and five-fold VGG16, an ensemble of CNN-designed, VGG19, and 5xVGG16, and the Vision Transformers (ViT) for a classification task. This assortment of models allowed us to examine the advantages and disadvantages of each, using their performance on the same task as a point of comparison. Across the board, it was the ViT models, specifically the B_32 model trained for 8000 epochs, that outperformed all others, achieving an accuracy of 71.14%. The ViT models' success underscores the potential of transformer-based models in image classification tasks, a domain traditionally dominated by convolutional-based approaches. Despite this success, it is worth noting that the high performance of ViT models also came with increased model complexity and potentially longer training times, emphasizing the trade-offs often encountered in model selection.

The CNN-designed models and their five-fold counterparts also demonstrated respectable performance, highlighting the effectiveness of custom models tailored to a particular task. Meanwhile, the performance of the VGG models and ensemble model, although commendable, was still outclassed by the ViT models. The ensemble approach did demonstrate that combining models can yield higher performance, but not necessarily surpass the best individual models in this experiment.

To conclude, these experiments underline the importance of model selection and fine-tuning in achieving optimal performance. While the ViT models demonstrated superior performance in this study, it is critical to consider other factors, such as computational resources, model complexity, and training times. Indeed, the choice of model should be guided not only by performance metrics but also by the specific requirements and constraints of the task and context. Future work may include the exploration of other transformer-based models or further fine-tuning of the models studied here.

REFERENCES

[1] Hu, J., Song, W., Zhang, W., Zhao Y., Yilmaz A., (2019). Deep learning for use in lumber classification tasks Wood Sci Technol 53(2): 505-517.DOI: https://doi.org/10.1007/s00226-019-01086-z.

[2] Ibrahim, I., Khairuddin, A. S. M., Talip, M. S. A., Arof, H., Yusof, R.,
(2017). Tree species recognition system based on macroscopic image analysis. Wood science and technology, 51(2), 431-444.

[3] Jemielniak K., Urba´nski T., Kossakowska J., Bombi´nski S., (2012). Tool condition monitoring based on numerous signal features. Int J AdvManuf Technol 59: 73-81. DOI: https://doi.org/10.1007/s00170-011-3504-2.

[4] Kuo R., (2000). Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. Eng Appl Artif Intell 13: 249-261. DOI: https://doi.org/10.1016/S0952-1976(00)00008-7.

[5] Kurek J., Antoniuk I., Górski J., Jegorowa A., Świderski B., Kruk M., Wieczorek G., Pach J., Orłowski A., Aleksiejuk-Gawron J., (2019a). Data Augmentation Techniques for Transfer Learning Improvement in Drill Wear Classification Using Convolutional Neural Network. Machine Graphics and Vision 28: 3-12.

[6] Kurek J., Antoniuk I., G´orski J., Jegorowa A., Świderski B., Kruk M., Wieczorek G., Pach J., Orłowski A., Aleksiejuk-Gawron J., (2019b). Classifiers ensemble of transfer learning for improved drill wear classification using convolutional neural network. Machine Graphics and Vision 28:13-23.

[7] Kurek J., Kruk M., Osowski S., Hoser P., Wieczorek G., Jegorowa A., Górski J., Wilkowski J., Śmietańska K., Kossakowska J., (2016). Developing automatic recognition system of drill wear in standard laminated chipboard drilling process Bulleting of the Polish Academy of Science. Technical Sciences 64: 633-640. DOI: https://doi.org/10.1515/bpasts-2016-0071.

[8] Kurek J., Swiderski B., Jegorowa A., Kruk M., Osowski S., (2017a). Deep learning in assessment of drill condition on the basis of images of drilled holes In: International Conference on Graphic and Image Processing. ICGIP. DOI: https://doi.org/10.1117/12.2266254.

[9] Kurek J., Wieczorek G., Świderski B., Kruk M., Jegorowa A., Osowski S., (2017b). Transfer learning in recognition of drill wear using convolutional neural network. 1. In: International Conference on Computational Problems of Electrical Engineering. IEEE. DOI: https://doi.org/10.1109/CPEE.2017.8093087.

[10]Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. & Houlsby, N. An Image is Worth 16x16 Words:Transformers for Image Recognition at Scale. ICLR. (2021)

[11] Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M. & Dosovitskiy, A. MLP-Mixer: An all-MLP Architecture for Vision. ArXiv Preprint ArXiv:2105.01601. (2021)

[12] Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J. & Beyer, L. How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers. ArXiv Preprint ArXiv:2106.10270. (2021)

[13] Chen, X., Hsieh, C. & Gong, B. When Vision Transformers Outperform ResNets without Pretraining or Strong Data Augmentations. ArXiv Preprint ArXiv:2106.01548. (2021)

[14] Zhuang, J., Gong, B., Yuan, L., Cui, Y., Adam, H., Dvornek, N.,Tatikonda, S., Duncan, J. & Liu, T. Surrogate Gap Minimization Improves Sharpness-Aware Training. ICLR. (2022)

[15] Zhai, X., Wang, X., Mustafa, B., Steiner, A., Keysers, D., Kolesnikov, A. & Beyer, L. LiT: Zero-Shot Transfer with Locked-image Text Tuning. CVPR. (2022)

[16] Steiner, A., Kolesnikov, A., Zhai, X., Wightman, R., Uszkoreit, J. & Beyer, L. How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers. (2022)

[17] Jegorowa, A.; Górski, J.; Kurek, J.; Kruk, M. Use of nearest neighbors (K-NN) algorithm in tool condition identification in the case of drilling in melamine faced particleboard. Maderas Cienc. Tecnol. 2020, 22, 189–196. https://doi.org/10.4067/S0718-221X2020005000205.

[18] Jegorowa, A., Kurek, J., Antoniuk, I., Dołowa, W., Bukowski, M. & Czarniak, P. Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. Wood Science And Technology. 55, 271-293 (2021,1,1), https://doi.org/10.1007/s00226-020-01245-7

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M, Jegorowa A, Kurek J. A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard. Przeglad Elektrotechniczny. 2024 May 1;2024(5).DOI 10.15199/48.2024.05.52
mój indywidualny udział w jej powstaniu polegał na:

1. Koncepcji.
2. Metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.
7. Edycji i korekcji artykułu.

[illegible]
Podpis

Warszawa, 27-01-2025

Albina Jegorowa
albina_jegorowa@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M, Jegorowa A, Kurek J. A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard. Przeglad Elektrotechniczny. 2024 May 1;2024(5).DOI 10.15199/48.2024.05.52
mój indywidualny udział w jej powstaniu polegał na:

1. Akwizycji danych.
2. Zarządzaniu danymi.

[illegible]
Podpis

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski M, Jegorowa A, Kurek J. A Novel Approach using Vision Transformers (VIT) for Classification of Holes Drilled in Melamine Faced Chipboard. Przeglad Elektrotechniczny. 2024 May 1;2024(5).DOI 10.15199/48.2024.05.52
mój indywidualny udział w jej powstaniu polegał na:

1. Przygotowaniu pierwszego szkicu.
2. Nadzorze.
3. Administracji.
4. Walidacji.

[illegible]
Podpis

10.6. Publikacja 6

PUBLIKACJA 6

Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard

Michał Bukowski, Jarosław Kurek, Bartosz Świderski, Albina Jegorowa

Sensors 2024, 24, 1092.

https://doi.org/10.3390/s24041092

Punktacja: 100, IF: 3,9

Article

Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard

Michał Bukowski [1], Jarosław Kurek [1,*], Bartosz Świderski [1] and Albina Jegorowa [2]

1 Institute of Information Technology, Warsaw University of Life Sciences, 02-776 Warsaw, Poland; michal.bukowski@buksoft.pl (M.B.); bartosz_swiderski@sggw.edu.pl (B.Ś.)
2 Institute of Wood Sciences and Furniture, Warsaw University of Life Sciences, 02-787 Warsaw, Poland; albina_jegorowa@sggw.edu.pl
* Correspondence: jaroslaw_kurek@sggw.edu.pl

Abstract: The advancement of machine learning in industrial applications has necessitated the development of tailored solutions to address specific challenges, particularly in multi-class classification tasks. This study delves into the customization of loss functions within the eXtreme Gradient Boosting (XGBoost) algorithm, which is a critical step in enhancing the algorithm's performance for specific applications. Our research is motivated by the need for precision and efficiency in the industrial domain, where the implications of misclassification can be substantial. We focus on the drill-wear analysis of melamine-faced chipboard, a common material in furniture production, to demonstrate the impact of custom loss functions. The paper explores several variants of Weighted Softmax Loss Functions, including Edge Penalty and Adaptive Weighted Softmax Loss, to address the challenges of class imbalance and the heightened importance of accurately classifying edge classes. Our findings reveal that these custom loss functions significantly reduce critical errors in classification without compromising the overall accuracy of the model. This research not only contributes to the field of industrial machine learning by providing a nuanced approach to loss function customization but also underscores the importance of context-specific adaptations in machine learning algorithms. The results showcase the potential of tailored loss functions in balancing precision and efficiency, ensuring reliable and effective machine learning solutions in industrial settings.

Keywords: drill-wear analysis in woodworking; XGBoost custom loss functions; melamine-faced chipboard machining; critical error reduction in classification

check for updates

Citation: Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A. Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. *Sensors* **2024**, *24*, 1092. https://doi.org/10.3390/s24041092

Academic Editor: Yuan Yao

Received: 19 December 2023
Revised: 29 January 2024
Accepted: 6 February 2024
Published: 7 February 2024

1. Introduction

Machine learning (ML) has become a cornerstone in various industrial applications, revolutionizing how data are utilized to make critical decisions. Among the plethora of ML algorithms, eXtreme Gradient Boosting (XGBoost) has emerged as a powerful tool, particularly for classification tasks. However, the conventional implementation of XGBoost often falls short in complex multi-class classification scenarios, where the stakes of misclassification are high. This paper addresses the pivotal role of customizing loss functions within the XGBoost framework to cater to the nuanced demands of specific industrial applications, particularly focusing on the drill-wear analysis in melamine-faced chipboard production—a critical process in the furniture manufacturing industry [1–5].

Laminated chipboard panels are commonly applied materials in the furniture industry, known for their cost-effectiveness. However, they also present unique challenges during the production process, such as wear and tear on tools and unpredictability in material consistency due to variations in glue density and the presence of air pockets [6–9]. Consequently, accurately determining the precise timing for tool replacement during drilling operations is a critical issue that calls for automated solutions [10–12].

Tool condition monitoring in wood-based industries, including furniture manufacturing, has been extensively explored, with various methodologies already in place. These include using a range of sensors to measure parameters such as acoustic emission, noise, vibrations, cutting torque, and feed force [13–17]. Furthermore, the application of ML algorithms in this domain has been diverse, ranging from monitoring overall machining processes to identifying different wood species [18–20].

Recent trends have shifted towards leveraging simplified inputs, such as images, with transfer and deep learning methodologies that have been proven effective [21,22]. These approaches are further enhanced by data augmentation and classifier ensemble techniques [23,24]. The choice of classifiers also significantly impacts solution quality [25–28].

At the heart of ML-based solutions is the crucial step of feature extraction. In the context of drill wear recognition, numerous parameters derived from images of drilled holes can be considered. There is a growing need for systematic methodologies to select the most compelling feature extraction methods, highlighting the potential benefits of different approaches [10,11,13,17,21–24,26–32].

In recent years, the eXtreme Gradient Boosting (XGBoost) [33–37] algorithm has emerged as a powerful tool for tackling complex predictive tasks. Its ability to handle non-linear relationships and interactions between features makes it particularly suitable for industrial applications where precision and reliability are paramount. However, traditional applications of XGBoost may not fully address the nuances of critical error mitigation in the context of drill-wear analysis.

This study introduces novel custom loss functions within the XGBoost framework, aiming to enhance the model's performance in identifying and mitigating critical errors in the drill-wear analysis of melamine-faced chipboard. The rationale behind customizing loss functions is to fine-tune the algorithm's sensitivity to errors that have significant implications for production quality and efficiency. By tailoring the loss functions, we aim to achieve a more nuanced and contextually aware model that can provide more accurate predictions and insights, thereby reducing downtime and improving overall manufacturing processes.

Furthermore, the adaptation of XGBoost with these custom loss functions presents a novel approach in the field of industrial machine learning applications. It exemplifies the potential for algorithmic customization to meet specific industrial challenges, paving the way for more targeted and effective machine learning solutions in various sectors. The following sections detail the methodology adopted for this study, the development and implementation of the custom loss functions, and an analysis of the results obtained from applying this novel approach to drill-wear analysis in melamine-faced chipboard processing.

Despite the advancements in applying ML algorithms in the wood industry, a significant research gap remains in addressing the specific challenges of drill-wear analysis in the production of melamine-faced chipboards. The current study aims to fill this gap by exploring the use of custom loss functions in the XGBoost algorithm for enhanced critical error mitigation. The motivation behind this research stems from the industry's need for precise tool condition monitoring, which directly impacts production quality and efficiency.

A critical aspect that differentiates this study from conventional approaches is the specific goal set by industry experts. In this context, the primary objective is not merely to achieve the highest accuracy metric but to minimize the number of critical errors. This means that the model should ideally avoid misclassifying the extreme classes, specifically minimizing errors between the 'Green' and 'Red' classes, even if it leads to a higher overall accuracy error. The rationale is that in industrial applications, particularly in the context of drill-wear analysis, the consequences of misclassifying a tool in need of immediate replacement (Red) as in good condition (Green), or vice versa, can be far more detrimental than a lower accuracy in less critical classifications. This approach represents a significant shift from traditional ML objectives and underscores the importance of tailoring ML solutions to meet specific industrial needs. The pursuit of this goal forms the core of our research, aiming to develop a model that aligns with the practical demands of the furniture manufacturing industry, ensuring reliability and effectiveness in a real-world production environment.

Similarly to the exploration of various custom loss functions for different learning requirements in this article, future work could also delve into the analysis of network sensitivity, as demonstrated in several key studies. Naik and Kiran (2021) proposed a novel sensitivity-based method for feature selection within deep neural networks, showing its efficacy in identifying important features across datasets. Asheghi et al. (2020) enhanced neural network models for sediment load prediction by employing diverse sensitivity analysis methods, leading to improved model predictability. Yeung et al. (2010) discussed the significance of sensitivity analysis in artificial neural networks for engineering system design in their comprehensive book. Abbaszadeh Shahri et al. (2022) introduced an innovative approach for uncertainty quantification in groundwater table modeling, leveraging automated predictive deep learning for more accurate predictions. Ghaderi and Abbaszadeh Shahri (2022) developed a hybrid intelligence model to delineate soil layers using clay sensitivity, showcasing the potential of hybrid models in geo-engineering applications. These studies collectively highlight the crucial role of sensitivity analysis in enhancing model performance and reliability across various domains [38–42].

2. Materials and Methods

2.1. Data Collection

For the experimental analysis, image data were acquired in a setting that closely resembled a real-world production environment. The drilling process utilized a sophisticated automated CNC workstation, specifically the Busselato Jet 100 model, manufactured in Piovenne Rochette, Italy. Drilling was executed using a standard FABA drill with a diameter of 12 mm (model WP-01 (FABA, Treviso, Italy), see Figure 1). Operational parameters, including a rotation speed of 4500 RPM and a feed rate of 1.35 m/min, were selected based on the recommendations provided by the drill manufacturer. The material used for drilling was standard laminated chipboard, specifically the KRONOPOL U 511 model (Kronopol, Żary, Poland), featuring a thickness of 18 mm, a standard specification in the furniture industry [43–45]. The experiment involved the preparation of 610 distinct chipboard profiles from the laminated chipboard panels [46] (illustrated in Figure 2). Each profile measured 300 mm $\times$ 35 mm $\times$ 18 mm and included 14 evenly spaced holes located centrally. This arrangement was chosen to minimize the impact of material stress on the quality of the holes, thus ensuring a consistent and uniform drilling area across the panel. The total number of holes drilled in all the panels amounted to 8540. Regular inspections were conducted using a Mitutoyo microscope (model TM-500, based in Kawasaki, Japan) to assess the condition of the drill throughout the process. According to the observed wear, the drill was categorized into one of three distinct classes. The first, Good (Green), is indicative of a new drill in optimal condition for use. The second, Worn (Yellow), signifies a drill in an exemplary state; it is still functional but potentially needs replacing soon. The third, Requiring Replacement (Red), represents a drill that is no longer usable and needs immediate replacement.

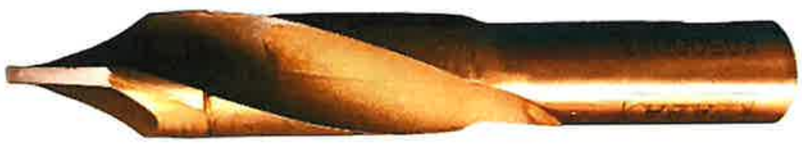


Figure 1. The FABA WP-01 drill used during experiments.

In the evaluation of drill conditions during inspections, the state of the drill was determined based on a wear parameter, denoted as W. This parameter represents the disparity in the width of the cutting edge when comparing a new drill (measurement taken near the outer corner) to the width of the same section in the tool under review.

The unit of measurement for W is millimeters [mm]. Based on the guidelines provided by the manufacturer, the condition of the drill is classified into distinct categories. A drill is considered in the 'Good' condition if W is less than 0.2 [mm]. It falls under the 'Worn' category if W lies between 0.2 [mm] and 0.35 [mm]. When W exceeds 0.35 [mm], the drill is deemed to require replacement.

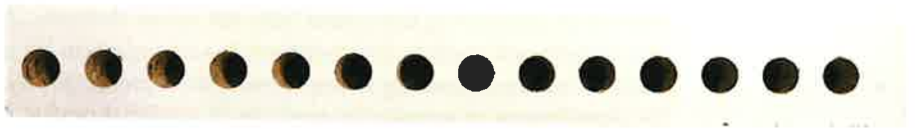


Figure 2. Example initial profile scans with drilled holes.

Subsequent to these assessments, the obtained profiles underwent scanning as part of the further analysis processes. The scanning was conducted at a resolution of 1200 dots per inch [dpi]. For examples of sample profiles that exhibit holes drilled by tools of varying wear conditions, refer to Figure 2.

A total of 610 images of chipboard profiles were captured and subsequently segregated into distinct files corresponding to each hole. This organization facilitated not only the individual examination of each case but also the automation of subsequent image processing stages. The composition of the dataset applied is detailed in Table 1.

Table 1. Breakdown of image counts for each drill class in the dataset.

Drill Number	Green Class	Yellow Class	Red Class	Total
Drill 1	840	420	406	1666
Drill 2	840	700	280	1820
Drill 3	700	560	420	1680
Drill 4	840	560	280	1680
Drill 5	560	560	560	1680
Total	3780	2800	1946	8526

2.2. Feature Extraction Techniques

Several methods for generating features, resulting in five primary feature groups, have been selected. The figure below (Figure 3) summarizes the total count of features derived from each method.

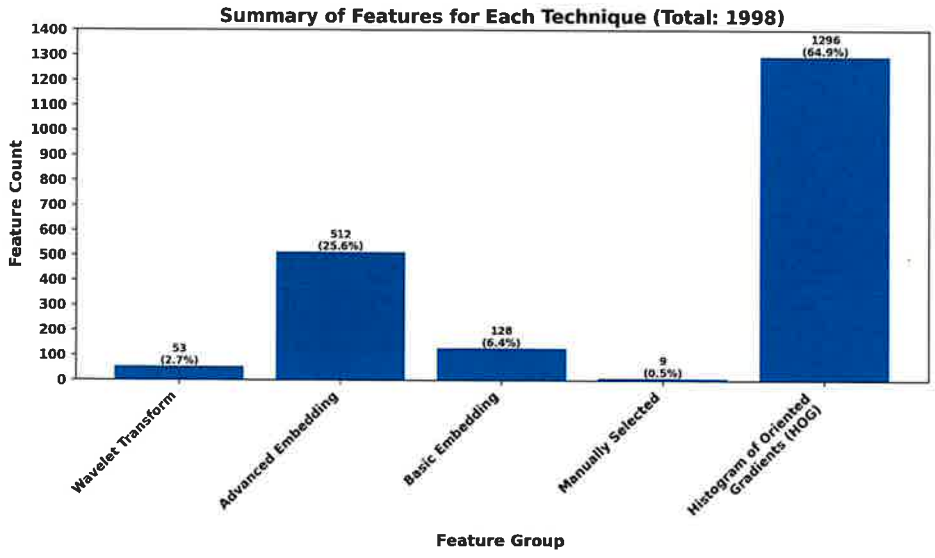


Figure 3. Summary of features for each technique.

The bar chart presented provides a comprehensive summary of features utilized across different feature extraction techniques. A total of 1998 features have been cataloged, with their distribution across various methods explicitly detailed.

At the lower end of the spectrum, Wavelet Transform accounts for a modest portion, representing 53 features, which is approximately 2.7% of the total. High-level Embedding (based on Pretrained Convolutional Neural Network-CNN-512 variables) shows a more significant contribution with 512 features, forming 25.6% of the overall count. Low-level embedding (based on Pretrained Convolutional Neural Network-CNN-128 variables) makes up 6.4%, while Manual Selection is the least represented with only nine features, equating to a mere 0.5%.

The most substantial proportion of features is attributed to the Histogram of Oriented Gradients (HOG) technique, which encompasses a staggering 1296 features. This figure constitutes 64.9% of the total feature count, underscoring the predominant reliance on this method in the feature extraction process.

The chart effectively illustrates the reliance on different feature extraction techniques in the analysis, with HOG standing out as the most utilized method by a significant margin.

2.2.1. 2-D Morlet Wavelets in Wavelet Scattering Image Decomposition

The wavelet scattering approach, often referred to as the scattering transform or wavelet transform, applies convolutions of input signals to gauge similarities among objects. This approach's efficiency in recognizing local correlations arises from the merging of two akin values.

Tracing the origins of wavelet scattering brings us to the Fourier transform's development, an essential technique in signal processing. However, Fourier representation struggles with instability at higher frequencies during signal deformation. This is attributed to the sine wave's limitations in localizing frequency-specific information [47].

Addressing this issue, the wavelet transform decomposes [48] a signal into a set of varying wavelets, effectively pinpointing the signal's high-frequency elements. The translation property inherent in the wavelet operator ensures covariance in the representation—a shift in the signal correspondingly shifts its wavelet coefficients, posing a challenge for comparing signals with translations. Achieving translation invariance is crucial for applications such as classification.

Fundamentally, the wavelet transform operates on the dot product using a kernel. This kernel is a wavelet function tailored to meet time-frequency analysis needs. The wavelet's frequency domain spectrum is adjustable through the scale parameter a, while the time domain spectrum is adjusted via the offset parameter b.

The applied decomposition utilizes a Morlet wavelet, incorporating both real and complex components, and is graphically represented in the time domain [49,50]:

$$\Psi_\sigma(t) = c_\sigma \pi^{-\frac{1}{4}} e^{-\frac{1}{2}t^2}(e^{i\sigma t} - \kappa_\sigma) \tag{1}$$

where κ_σ serves as the admissibility criterion:

$$\kappa_\sigma = e^{-\frac{1}{2}\sigma^2} \tag{2}$$

and c_σ is the normalization constant:

$$c_\sigma = \left(1 + e^{-\sigma^2} - 2e^{-\frac{3}{4}\sigma^2}\right)^{-\frac{1}{2}} \tag{3}$$

The fast Fourier transform for this system is expressed as:

$$\hat{\Psi}_\sigma(\omega) = c_\sigma \pi^{-\frac{1}{4}} \left(e^{-\frac{1}{2}(\sigma-\omega)^2} - \kappa_\sigma e^{-\frac{1}{2}\omega^2}\right) \tag{4}$$

This process involves a Gaussian-windowed sine wave conducting the convolution across various frequency locations using indexed wavelets (Ψ_v), with the wavelet transform (Ψ_x) producing a range of dispersion coefficients.

This methodology, when applied to the chosen dataset, yielded a total of 53 distinct features.

The implementation of wavelet feature extraction from images, as described in the context of 2-D Morlet wavelets and wavelet scattering image decomposition, is detailed in Algorithm 1, where the specific steps of the procedure are outlined.

Algorithm 1 Wavelet Feature Extraction from Images

```
 1: procedure WAVELETFEATUREEXTRACTION
 2:     Define root folders and classes for image datasets        ▷ for 5 drills and 3 classes
 3:     Create imageDatastore instances for each dataset and classes
 4:     for each dataset combination (1 through 5) do
 5:         Combine training sets from other four datasets
 6:         Assign corresponding labels to the combined training set
 7:         Set aside one dataset as the testing set
 8:         Extract wavelet features for both training and testing sets
 9:         Assign labels to the training and testing sets
10:     end for
11:     Save all wavelet features and labels to a file
12: end procedure
13:                                                                ▷
14: procedure GETWAVELETSCATTERING2(ImageDatastore)
15:     Read all images from the datastore
16:     Define wavelet scattering for ImageSize=[224 224]
17:     Initialize empty array for all features
18:     for each image in the datastore do
19:         Extract wavelet features using defined wavelet scattering
20:         Compute the mean of the features
21:         Append the features to the features array
22:     end for
23:     return features array
24: end procedure
```

2.2.2. Pretrained Network: ResNet-18 for Extracting Lower-Level and High-Level Features

ResNet-18, part of the Residual Network family, is a convolutional neural network (CNN) that has been pretrained on more than a million images from the ImageNet database [51,52]. Despite its overall depth, the '18' in ResNet-18 signifies the count of its primary layers: the convolutional and fully-connected layers [53]. The network is 18 layers deep and can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images.

The network layers include a series of convolutional layers, each followed by batch normalization and a rectified linear unit (ReLU) activation function. The defining characteristic of ResNet is the introduction of skip connections (or shortcut connections) that bypass one or more layers. These connections are represented as identity mappings, adding the output from an earlier layer to a later layer. This helps to mitigate the vanishing gradient problem that affects deep networks, allowing for the training of much deeper networks.

In our work, we utilize ResNet-18 as a feature extractor. The extracted features from the intermediate layers of the network are used as input to machine learning models for various classification tasks. The pretrained network is beneficial in scenarios where the dataset is not sufficiently large to train a deep network from scratch. By leveraging transfer learning, we can apply the powerful feature representations learned by ResNet-18 on ImageNet to new tasks with limited data.

The architecture of ResNet-18, as depicted in Figure 4, consists of an initial convolutional layer followed by a series of residual blocks that make up the core of the network. These residual blocks contain the skip connections that are the hallmark of the ResNet architecture. The network concludes with a global average pooling layer and a fully connected layer that outputs the class probabilities.

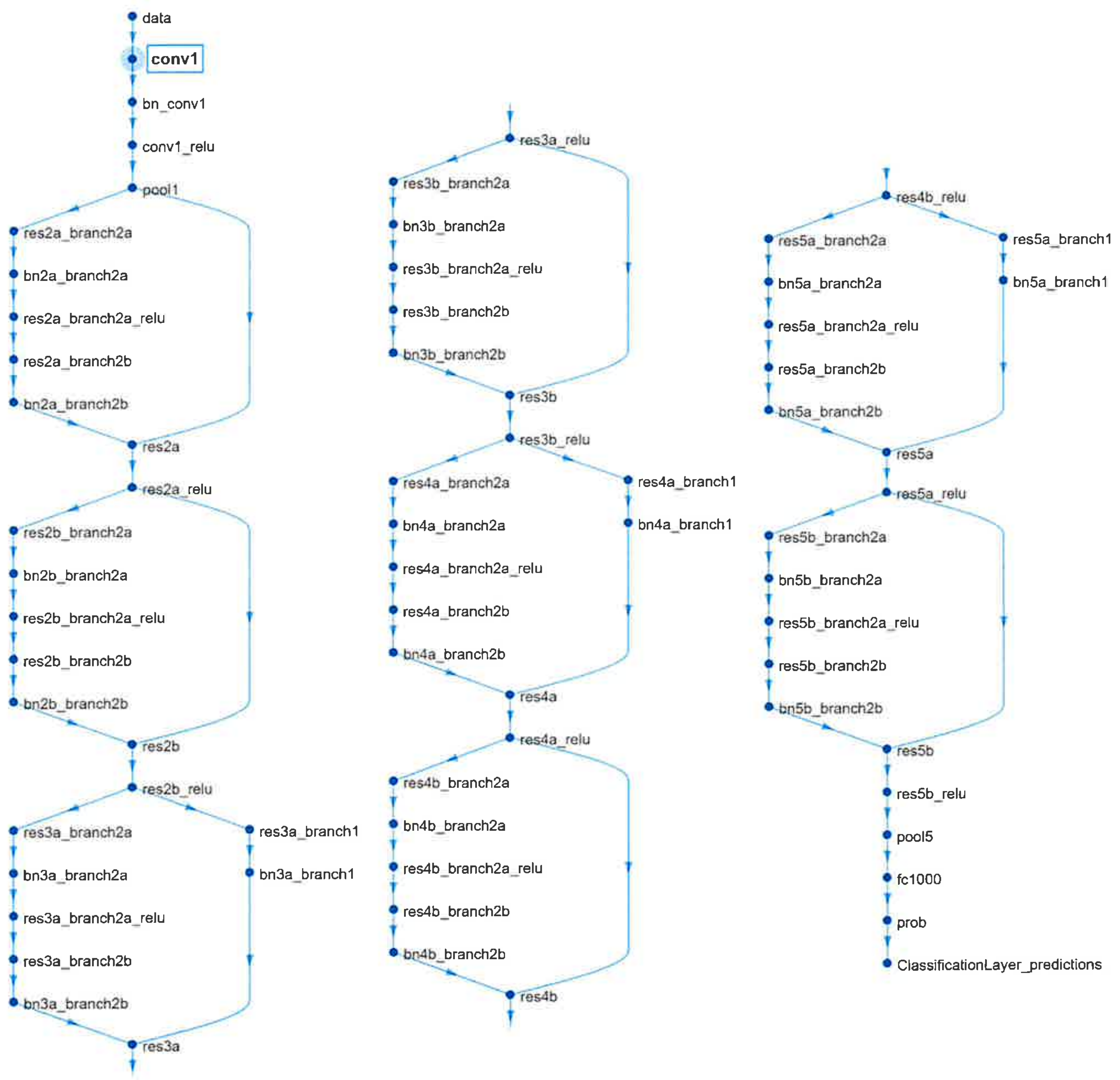


Figure 4. The architecture of the pretrained ResNet-18 network.

For a comprehensive understanding of ResNet-18 architecture, including the specific functions of each layer, Table 2 has been added. This table provides a detailed layer-by-layer description of the ResNet-18 network, highlighting the critical roles of layers 35 and 68 in feature extraction.

Table 2. Detailed layer-by-layer description of the ResNet-18 network architecture. Layers 35 and 68, highlighted in the table, are used for extracting lower-level and high-level features, respectively.

No.	Layer's Name	Layer's Type	Description	Total Learnables
1	data	Image Input	224 × 224 × 3 images with 'zscore' normalization	0
2	conv1	2-D Convolution	64 7 × 7 × 3 convolutions with stride [2 2] and padding [3 3 3 3]	9472
3	bn_conv1	Batch Normalization	Batch normalization with 64 channels	128
4	conv1_relu	ReLU	ReLU	0
5	pool1	2-D Max Pooling	3 × 3 max pooling with stride [2 2] and padding [1 1 1 1]	0
6	res2a_branch2a	2-D Convolution	64 3 × 3 × 64 convolutions with stride [1 1] and padding [1 1 1 1]	36,928
7	bn2a_branch2a	Batch Normalization	Batch normalization with 64 channels	128
8	res2a_branch2a_relu	ReLU	ReLU	0
9	res2a_branch2b	2-D Convolution	64 3 × 3 × 64 convolutions with stride [1 1] and padding [1 1 1 1]	36,928
10	bn2a_branch2b	Batch Normalization	Batch normalization with 64 channels	128
11	res2a	Addition	Element-wise addition of 2 inputs	0
12	res2a_relu	ReLU	ReLU	0
13	res2b_branch2a	2-D Convolution	64 3 × 3 × 64 convolutions with stride [1 1] and padding [1 1 1 1]	36,928
14	bn2b_branch2a	Batch Normalization	Batch normalization with 64 channels	128
15	res2b_branch2a_relu	ReLU	ReLU	0
16	res2b_branch2b	2-D Convolution	64 3 × 3 × 64 convolutions with stride [1 1] and padding [1 1 1 1]	36,928
17	bn2b_branch2b	Batch Normalization	Batch normalization with 64 channels	128
18	res2b	Addition	Element-wise addition of 2 inputs	0
19	res2b_relu	ReLU	ReLU	0
20	res3a_branch2a	2-D Convolution	128 3 × 3 × 64 convolutions with stride [2 2] and padding [1 1 1 1]	73,856
21	bn3a_branch2a	Batch Normalization	Batch normalization with 128 channels	256
22	res3a_branch2a_relu	ReLU	ReLU	0
23	res3a_branch2b	2-D Convolution	128 3 × 3 × 128 convolutions with stride [1 1] and padding [1 1 1 1]	147,584
24	bn3a_branch2b	Batch Normalization	Batch normalization with 128 channels	256
25	res3a_branch1	2-D Convolution	128 1 × 1 × 64 convolutions with stride [2 2] and padding [0 0 0 0]	8320
26	bn3a_branch1	Batch Normalization	Batch normalization with 128 channels	256
27	res3a	Addition	Element-wise addition of 2 inputs	0
28	res3a_relu	ReLU	ReLU	0
29	res3b_branch2a	2-D Convolution	128 3 × 3 × 128 convolutions with stride [1 1] and padding [1 1 1 1]	147,584
30	bn3b_branch2a	Batch Normalization	Batch normalization with 128 channels	256
31	res3b_branch2a_relu	ReLU	ReLU	0
32	res3b_branch2b	2-D Convolution	128 3 × 3 × 128 convolutions with stride [1 1] and padding [1 1 1 1]	147,584

Table 2. *Cont.*

No.	Layer's Name	Layer's Type	Description	Total Learnables
33	bn3b_branch2b	Batch Normalization	Batch normalization with 128 channels	256
34	res3b	Addition	Element-wise addition of 2 inputs	0
35	res3b_relu	ReLU	ReLU	0
36	res4a_branch2a	2-D Convolution	256 3 × 3 × 128 convolutions with stride [2 2] and padding [1 1 1 1]	295,168
37	bn4a_branch2a	Batch Normalization	Batch normalization with 256 channels	512
38	res4a_branch2a_relu	ReLU	ReLU	0
39	res4a_branch2b	2-D Convolution	256 3 × 3 × 256 convolutions with stride [1 1] and padding [1 1 1 1]	590,080
40	bn4a_branch2b	Batch Normalization	Batch normalization with 256 channels	512
41	res4a_branch1	2-D Convolution	256 1 × 1 × 128 convolutions with stride [2 2] and padding [0 0 0 0]	33,024
42	bn4a_branch1	Batch Normalization	Batch normalization with 256 channels	512
43	res4a	Addition	Element-wise addition of 2 inputs	0
44	res4a_relu	ReLU	ReLU	0
45	res4b_branch2a	2-D Convolution	256 3 × 3 × 256 convolutions with stride [1 1] and padding [1 1 1 1]	590,080
46	bn4b_branch2a	Batch Normalization	Batch normalization with 256 channels	512
47	res4b_branch2a_relu	ReLU	ReLU	0
48	res4b_branch2b	2-D Convolution	256 3 × 3 × 256 convolutions with stride [1 1] and padding [1 1 1 1]	590,080
49	bn4b_branch2b	Batch Normalization	Batch normalization with 256 channels	512
50	res4b	Addition	Element-wise addition of 2 inputs	0
51	res4b_relu	ReLU	ReLU	0
52	res5a_branch2a	2-D Convolution	512 3 × 3 × 256 convolutions with stride [2 2] and padding [1 1 1 1]	1,180,160
53	bn5a_branch2a	Batch Normalization	Batch normalization with 512 channels	1024
54	res5a_branch2a_relu	ReLU	ReLU	0
55	res5a_branch2b	2-D Convolution	512 3 × 3 × 512 convolutions with stride [1 1] and padding [1 1 1 1]	2,359,808
56	bn5a_branch2b	Batch Normalization	Batch normalization with 512 channels	1024
57	res5a_branch1	2-D Convolution	512 1 × 1 × 256 convolutions with stride [2 2] and padding [0 0 0 0]	131,584
58	bn5a_branch1	Batch Normalization	Batch normalization with 512 channels	1024
59	res5a	Addition	Element-wise addition of 2 inputs	0
60	res5a_relu	ReLU	ReLU	0
61	res5b_branch2a	2-D Convolution	512 3 × 3 × 512 convolutions with stride [1 1] and padding [1 1 1 1]	2,359,808
62	bn5b_branch2a	Batch Normalization	Batch normalization with 512 channels	1024
63	res5b_branch2a_relu	ReLU	ReLU	0
64	res5b_branch2b	2-D Convolution	512 3 × 3 × 512 convolutions with stride [1 1] and padding [1 1 1 1]	2,359,808
65	bn5b_branch2b	Batch Normalization	Batch normalization with 512 channels	1024

Table 2. *Cont.*

No.	Layer's Name	Layer's Type	Description	Total Learnables
66	res5b	Addition	Element-wise addition of 2 inputs	0
67	res5b_relu	ReLU	ReLU	0
68	pool5	2-D Global Average Pooling	2-D global average pooling	0
69	fc1000	Fully Connected	1000 fully connected layer	513,000
70	prob	Softmax	softmax	0
71	ClassificationLayer _predictions	Classification Output	crossentropyex with 'tench' and 999 other classes	0

2.2.3. High-Level Features Feature Extraction Using Pretrained Convolutional Networks

In our study, high-level feature extraction was conducted using features from the network's 68th layer called 'pool5'. This layer was selected for its comprehensive accumulation of image-derived features essential for classification. Specifically, 512 distinct features were extracted for each image. The network's pre-training on the expansive ImageNet [51] database eliminates the need for additional training for effective feature extraction. For a complete layer-by-layer breakdown of the ResNet-18 architecture, and specifically the significance of the 68th layer in feature extraction, refer to Table 2.

The process of high-level feature extraction utilizing a pretrained convolutional neural network (CNN), such as ResNet-18, is summarized in Algorithm 2. This algorithm outlines the key steps for preprocessing the images and extracting features from a specific layer of the CNN.

Algorithm 2 CNN Feature Extraction Using ResNet-18's 35th/68th Layer

```
1: procedure CNNFeatureExtraction
2:     Initialize root folders for each dataset                 ▷ for 5 drills
3:     Define categories: Green, Yellow, Red                    ▷ for three subfolders
4:     Create imageDatastore instances for each dataset
5:     Combine datasets for training and testing
6:     Load pretrained ResNet-18 model
7:     Define feature extraction layer (e.g., 68th layer or 35th layer)
8:     for each combined dataset (1 through 5) do
9:         Read image
10:        Preprocess image according to CNN input requirements
11:        Extract features using the specified CNN layer
12:        Store extracted features
13:    end for
14:    Output the set of extracted features
15: end procedure
```

2.2.4. Low-Level Extracting of Features Using ResNet-18's 35th Layer

This study also investigates the efficacy of initial-stage features in addressing the problem at hand. To this end, a feature set was derived from the 35th layer of the ResNet-18 architecture. This process mirrored the methodology used for extracting features from the network's 68th layer. Typically, the initial layers of a deep learning network such as ResNet-18 are known for capturing more basic, rudimentary features. These layers generally maintain a higher spatial resolution and result in a greater total count of activations. Notably, the 35th layer is pivotal as it is the final layer contributing 128 novel features, ensuring an optimal spatial representation at a resolution of 28×28. A detailed description of the ResNet-18 network architecture, including the specific role of the 35th layer, can be found in Table 2.

The process of low-level extracting of features using the 35th layer of the ResNet-18 architecture is detailed in Algorithm 2. This approach leverages the initial-stage features from the CNN for our specific classification task

2.2.5. Manually Defined Feature Set

The extraction of features included consideration of physical properties derived from analyzing photographs of the defects. Specifically in furniture production, different types of flaws impact the quality of the end product differently. For example, a hole with a single, extensive chip is more detrimental than one with multiple, minor chips near the perimeter. The latter can be concealed during the assembly of the furniture, whereas the former often leads to the rejection of the product. To encapsulate these distinctions, a set of manually defined features was developed, encompassing nine key attributes:

1. Diameter of the smallest circle encompassing the hole;
2. Diameter of the largest circle that can fit inside the hole;
3. Variation in hole diameters;
4. Total area covered by holes;
5. Area of the convex hull;
6. Total perimeter length;
7. Length of the longest axis of the ellipse fitting the image;
8. Length of the shortest axis of the ellipse fitting the image;
9. Solidity (ratio of area to convex hull area).

The specific steps for generating these features are detailed in Algorithm 3.

Algorithm 3 Generation of Individual Image Features Using Custom Method

```
procedure GENERATEMANUALFEATURES (file)
    Read image $I$ from $file$
    Adjust $I$ and convert to binary images $bw$
    Label connected components in $bw$ to $L$
    Extract properties of regions in $L$ into $s$
    Find the largest region in $s$ as $circle$
    Extract edge coordinates $x, y$ from $circle$
    Calculate inscribed circle properties: $R, cx, cy$
    Calculate convex hull points $x, y$ of $circle$
    Calculate circumscribed circle properties: $center, radius$
    Compute features: $R, radius, radius - R$ , $s(numer).Area$, etc.
    return computed features as ManualFeatures
end procedure
```

2.2.6. Histogram of Oriented Gradients (HOG) for Feature Extraction

Histogram of Oriented Gradients (HOG) is a type of feature descriptor, similar to SIFT (scale-invariant feature transform) and SURF (speeded-up robust feature), widely used in the realm of computer vision for extracting features [54–58]. HOG is particularly adept at capturing an object's form or structural characteristics. Unlike basic edge detection, which distinguishes edge pixels, HOG takes this a step further by considering the directionality of edges, which is achieved by analyzing the gradient and orientation of the edges.

During the process of analysis, the image is segmented into smaller sections. For each of these segments, the gradients and their orientations are calculated, leading to the generation of individual histograms for each segment. These histograms are formulated on the basis of the gradients and the orientations of pixel values within the segment.

The overall procedure encompasses three phases: initial data preparation, calculation of gradients, and the ascertainment of orientations derived from the gradient data. Each alteration is assessed in both dimensions of a 2-D object, i.e., along the X and Y axes. The variations between these two dimensions form the foundation of the traditional Pythagorean methodology used in calculating gradient magnitude. Furthermore, the angle ϕ, indicating direction, is ascertained using these calculated metrics.

Application of the HOG method has enabled the extraction of 1296 distinct features from smaller regions, each with a defined cell size of 32×32.

The process of extracting HOG features is detailed in Algorithm 4.

Algorithm 4 Extract HOG Features

```
procedure EXTRACTHOGFEATURES ($image$)
    $gradients \leftarrow$ COMPUTEGRADIENTS($image$)
    $cells \leftarrow$ DIVIDEINTOCELLS($image$)
    $hogFeatures \leftarrow []$
    for each $cell$ in $cells$ do
        $histogram \leftarrow$ INITIALIZEHISTOGRAM
        for each $pixel$ in $cell$ do
            $gradient \leftarrow gradients[pixel]$
            $bin \leftarrow$ FINDORIENTATIONBIN($gradient$)
            ADDTOHISTOGRAM($histogram, bin, gradient.magnitude$)
        end for
        $normalizedHistogram \leftarrow$ NORMALIZEHISTOGRAM($histogram$)
        APPENDTOFEATUREVECTOR($hogFeatures, normalizedHistogram$)
    end for
    return $hogFeatures$
end procedure
```

2.3. Extreme Gradient Boosting—XGBoost

The eXtreme Gradient Boosting (XGBoost) algorithm is a highly efficient and scalable implementation of Gradient Boosting machines, a type of machine learning algorithm for regression and classification problems. XGBoost was developed by Chen and Guestrin (2016) [59] and has gained popularity in machine learning competitions due to its performance and speed.

Gradient Boosting is a machine learning technique for classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees. The method was further refined by Friedman (2000, 2001, 2002) [60–62] through the introduction of techniques such as stochastic Gradient Boosting and improvements in loss function optimization.

XGBoost is a refined and enhanced version of Gradient Boosting [63]. The core of the XGBoost algorithm involves sequentially adding predictors to an ensemble, each one correcting its predecessor. However, unlike traditional Gradient Boosting, XGBoost utilizes a more regularized model formalization to control overfitting, which gives it better performance.

2.4. Key Features of XGBoost

The key features of XGBoost include:

- Regularization: This includes L1 (Lasso Regression) and L2 (Ridge Regression) regularization, which helps in reducing overfitting.
- Handling Sparse Data: XGBoost is designed to handle sparse data from the ground up.
- Tree Pruning: XGBoost uses a depth-first approach for tree pruning, unlike the level-wise approach, making it more efficient.
- Handling Missing Values: XGBoost has an in-built routine to handle missing values.
- System Optimization: The system is optimized for distributed computing and can handle large datasets efficiently.

XGBoost was selected for its numerous benefits over the original model. It integrates regularization principles into the training process and supports parallel computation. The model also comes equipped with capabilities to address missing data and includes built-in procedures for cross-validation and tree pruning. Collectively, these features enable XGBoost potentially to outperform its predecessors [64].

In our analysis, eXtreme Gradient Boosting was configured with the following parameters:

- Number of Boosting Stages (M) = 50;
- Loss Function = 'log-loss';
- Maximum Depth of Trees = 3;
- Learning Rate = 0.1;
- Minimum Samples per Leaf = 1.

2.5. Loss Functions

Loss functions play a main role in the realm of machine learning (ML) modeling. They are the guiding metrics that an AI model aims to minimize during the training process. The choice of a loss function significantly influences the behavior and performance of the model, as it dictates how the model's predictions are evaluated against the actual outcomes.

The primary purpose of loss functions in AI models is to quantify the difference between the predicted values and the actual values. During the training phase, AI models use algorithms to adjust their parameters. The direction and magnitude of these adjustments are determined based on the loss function. A well-chosen loss function ensures that the model learns the correct patterns and makes accurate predictions.

While standard loss functions for classification tasks are practical in many scenarios, they are not one-size-fits-all solutions [65–67]. In particular, these functions may not perform optimally in situations with imbalanced data or when certain types of errors have more severe consequences than others.

In multi-class classification tasks, edge classes often represent categories with fewer samples or higher misclassification costs. Standard loss functions may not adequately penalize errors in these edge classes, leading to suboptimal model performance for these critical categories:

- Addressing Class Imbalance: Loss function modification can help address the class imbalance by assigning higher penalties for errors in underrepresented classes. This ensures that the model does not overlook these classes during the learning process.
- Focusing on Critical Errors: In many real-world applications, certain misclassifications are more costly than others. Customized loss functions can be designed to impose heavier penalties for specific types of critical errors, thereby reducing their occurrence.
- Improving Model Sensitivity: Modifying loss functions can improve the model's sensitivity towards edge classes, enhancing its ability to detect and correctly classify instances belonging to these classes.

Loss functions are more than just a measure of a machine learning model's error. They are instrumental in shaping the learning process. Customizing loss functions, particularly in the context of multi-class classification, can lead to significant improvements in model performance, especially for edge classes. This customization enables the development of AI models that are not only accurate but also aligned with the specific needs and priorities of their application domains.

2.6. XGBoost's Default Loss Function for Multi-Class Classification

XGBoost, a popular machine learning algorithm for boosted trees, uses different loss functions depending on the nature of the problem. For multi-class classification tasks, where the objective is to categorize instances into one of three or more classes, XGBoost defaults to the 'multi:softprob' loss function.

The 'multi:softprob' function is designed for multi-class classification problems. It calculates the probability of each class for an instance and uses these probabilities to minimize the loss during the training process.

The loss for a single instance is given by the negative log-likelihood of the true class [68]:

$$L(y,\hat{y}) = -\sum_{k=1}^{K} y_k \log(\hat{y}_k) \tag{5}$$

Here, K represents the total number of classes, y_k is a binary indicator (0 or 1) if class label k is the correct classification for the observation, and $\hat{y}_k$ is the predicted probability of the observation being of class k.

Pseudocode for 'multi:softprob'

The following pseudocode (Algorithm 5) outlines the basic approach of the 'multi:softprob' function during the training phase of XGBoost.

Algorithm 5 Pseudocode for multi:softprob in XGBoost

```
for each iteration do
    for each training instance $i$ do
        Calculate predictions $\hat{y}_i$ for all classes
        Compute probabilities using the softmax function
        for each class $k$ do
            Calculate $L(y_i, \hat{y}_i)$ for class $k$
        end for
        Update the model to minimize the loss
    end for
end for
```

This pseudocode simplifies the complex process behind XGBoost's 'multi:softprob' function while still capturing the essence of predicting class probabilities and updating the model to minimize loss.

2.7. *Weighted Softmax Loss Variant 1*

The 'WeightedSoftmaxLossVariant1' function extends the traditional Softmax Loss Function by introducing class-specific weights. This modification is beneficial in addressing class imbalances and emphasizing specific class pairings over others.

The function processes predictions y_{pred} and true labels y_{true} in a multistep approach, starting with a softmax transformation and followed by the application of class-specific weights in the gradient and Hessian computation. See (Algorithm 6).

Algorithm 6 Weighted Softmax Loss Function—Variant 1

```
Require: $y, \hat{y}$
  $\hat{y} \leftarrow \exp(\hat{y} - \max(\hat{y}))$
  $\hat{y} \leftarrow \hat{y} / \sum(\hat{y})$
  $grad \leftarrow$ zero matrix with shape of $\hat{y}$
  $hess \leftarrow$ zero matrix with shape of $\hat{y}$
  $weights \leftarrow \{(0,1) : 0.1, (1,0) : 0.1, (0,2) : 0.17, (2,0) : 0.17, (1,2) : 0.1, (2,1) : 0.1\}$
  for $i = 0$ to length of $y - 1$ do
      for $j = 0$ to number of columns in $\hat{y} - 1$ do
          $weight \leftarrow weights[\min(y[i], j), \max(y[i], j)]$
          if $weight$ is not set then
              $weight \leftarrow 0$
          end if
          $prob \leftarrow \hat{y}[i, j]$
          $grad[i, j] \leftarrow weight \times (prob - (y[i] == j))$
          $hess[i, j] \leftarrow weight \times prob \times (1 - prob) + 0.02$
      end for
  end forreturn $grad, hess$
```

2.7.1. Softmax Transformation

The softmax transformation is defined as [69]:

$$\hat{y} = \frac{\exp(\hat{y} - \max(\hat{y})}{\sum \exp(\hat{y} - \max(\hat{y}))} \tag{6}$$

2.7.2. Gradient and Hessian Initialization

Both gradient grad and Hessian hess matrices are initialized as zero matrices of the same dimensions as y_{pred}.

2.7.3. Weight Assignment

The function utilizes a predefined dictionary of weights for class pairs. The weights for some of the class pairings are as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$;
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.17$;
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$.

These weights are used to scale the impact of each class pair differently, allowing for a customized response to the imbalance or specific importance of class combinations.

2.7.4. Gradient and Hessian Computation

For each instance and class, the gradient and Hessian are computed as follows [70]:

$$\text{grad}_{ij} = w_{ij} \cdot (p_{ij} - \mathbb{1}(y_i == j)) \tag{7}$$

$$\text{hess}_{ij} = w_{ij} \cdot p_{ij} \cdot (1 - p_{ij}) + \lambda \tag{8}$$

where p_{ij} is the predicted probability of instance i belonging to class j, $\mathbb{1}$ is the indicator function, and λ is a regularization term to prevent overfitting.

By incorporating class-specific weights, the 'WeightedSoftmaxLossVariant1' function provides a nuanced approach to handling classification tasks. This functionality is particularly advantageous in datasets where certain class combinations are more critical or where there is a significant class imbalance. The flexibility to adjust weights as needed allows the function to be tailored to the specific requirements of various datasets and classification challenges.

2.8. Weighted Softmax Loss Function—Variant 2

The 'WeightedSoftmaxLossVariant2' function is a modification of the earlier described 'WeightedSoftmaxLossVariant1'. This variant introduces a slight change in the weights assigned to class pairs, refining the approach towards class-specific penalization in classification tasks.

The methodological steps in 'WeightedSoftmaxLossVariant2' remain largely the same as in Variant 1, with the primary difference lying in the weight assignment for class pairs.

Modified Weight Assignment

The key modification in Variant 2 is in the weight assignment. The new weights for the class pairs are as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$;
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.18$;
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$.

The adjustment in weights, particularly for the class pairs (0, 2) and (2, 0), signifies a refined approach to how the function penalizes or prioritizes different class pairs.

The 'WeightedSoftmaxLossVariant2' function demonstrates how subtle changes in the weight configuration can lead to significant differences in the behavior of loss functions. By adjusting the weights for specific class pairs, the function becomes more adaptable to the nuances of different classification problems, particularly where specific class pairs require more attention due to imbalance or other specific considerations.

2.9. Weighted Softmax Loss Function—Variant 3

The 'WeightedSoftmaxLossVariant3' function introduces another iteration of refinements in the Weighted Softmax Loss approach. Similar to its predecessors, this variant adjusts the weights assigned to specific class pairs, refining the model's sensitivity to different class interactions.

The fundamental methodology of 'WeightedSoftmaxLossVariant3' remains consistent with the previous versions, with the primary distinction being the further adjustment of the class pair weights.

Modified Weight Assignment

The new weights in 'WeightedSoftmaxLossVariant3' are as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$;
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.19$;
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$.

This modification, particularly for the class pairs (0, 2) and (2, 0), indicates a continued refinement in the model's approach to handling these class interactions.

'WeightedSoftmaxLossVariant3' further exemplifies the adaptive nature of weighted loss functions in machine learning classification tasks. By incrementally adjusting the weights for specific class pairs, this function allows for a more fine-tuned response to class imbalances and specific classification dynamics. Such iterative refinements highlight the importance of customizing loss functions to suit the specific needs of diverse datasets and classification challenges.

2.10. Weighted Softmax Loss Function—Variant 4

The 'WeightedSoftmaxLossVariant4' represents the penultimate iteration in the series of modifications to the Weighted Softmax Loss function. This version continues to adjust the weighting strategy for class pairs, reflecting an ongoing refinement to address the specific nuances of classification tasks better.

The overall structure and methodology of 'WeightedSoftmaxLossVariant4' remain consistent with its predecessors, with the critical difference being the updated weight values for certain class pairs.

Modified Weight Assignment

In 'WeightedSoftmaxLossVariant4', the weights for class pairs are adjusted as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$;
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.20$;
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$.

This change, especially for the class pairs (0, 2) and (2, 0), indicates a continual refinement in the approach toward handling these specific class interactions.

The progression to 'WeightedSoftmaxLossVariant4' underscores the importance of continuous optimization in loss function design, particularly in the context of class imbalances and specific classification dynamics. By fine-tuning the weights assigned to different class pairs, this variant offers an even more nuanced approach to addressing the complexities inherent in diverse datasets and classification challenges.

2.11. Weighted Softmax Loss Function—Variant 5

'WeightedSoftmaxLossVariant5' represents the culmination of a series of refinements in the Weighted Softmax Loss function series [71]. This final variant further fine-tunes the weighting system for class pairs, aiming to optimize the classification performance in machine learning models.

The methodology for 'WeightedSoftmaxLossVariant5' follows the established pattern of its predecessors, with modifications confined to the weights assigned to the class pairs.

Modified Weight Assignment

The weights for class pairs in 'WeightedSoftmaxLossVariant5' are adjusted as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.21$
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$

These adjustments, especially for the class pairs (0, 2) and (2, 0), reflect an ongoing effort to optimize the model's sensitivity and response to specific class dynamics.

With 'WeightedSoftmaxLossVariant5', the series of modifications to the Weighted Softmax Loss function reaches a refined state, demonstrating a meticulous approach to enhancing classification models. By incrementally adjusting the weights for class pairs, this variant offers a sophisticated balance, addressing class imbalances and specific classification challenges with heightened precision. This iterative process highlights the value of continual optimization in loss function design, catering to the complex needs of diverse and evolving datasets.

2.12. *Weighted Softmax Loss Function with Edge Penalty*

The 'WeightedSoftmaxLossWithEdgePenalty' function introduces an advanced modification to the standard Weighted Softmax Loss function by incorporating an extra penalty for edge class errors. This enhancement aims to address the challenges posed by class imbalances, particularly when errors in extreme classes (or edge classes) have a more significant impact.

The function maintains the core structure of the Weighted Softmax Loss but introduces an additional penalty factor for errors involving the edge classes.

2.12.1. Modified Weight Assignment

The function assigns weights to class pairs and introduces an extra penalty for errors involving the edge classes.

The weights for class pairs in 'WeightedSoftmaxLossWithEdgePenalty' are adjusted as follows:

- Classes 0 and 1: $w_{0,1} = w_{1,0} = 0.1$;
- Classes 0 and 2: $w_{0,2} = w_{2,0} = 0.17$;
- Classes 1 and 2: $w_{1,2} = w_{2,1} = 0.1$.

2.12.2. Algorithm of Weighted Softmax Loss Function with Edge Penalty

The algorithm for the 'WeightedSoftmaxLossWithEdgePenalty' function is presented as follows (Algorithm 7).

Algorithm 7 Weighted Softmax Loss with Edge Penalty

```
Require: $y, \hat{y}$
  Initialize $grad$ and $hess$ as zero matrices of the same shape as $\hat{y}$
  Define class pair weights
  Set $extra\_penalty \leftarrow 1.2$
  for $i = 0$ to $len(y) - 1$ do
      for $j = 0$ to $columns(\hat{y}) - 1$ do
          $weight \leftarrow$ weight for the pair $(\min(y[i], j), \max(y[i], j))$
          $prob \leftarrow \hat{y}[i, j]$
          $penalty \leftarrow extra\_penalty$ if $y[i]$ or $j$ is an edge class, else 1
          $grad[i, j] \leftarrow penalty \times weight \times (prob - (y[i] == j))$
          $hess[i, j] \leftarrow penalty \times weight \times prob \times (1 - prob) + 0.02$
      end for
  end forreturn $grad, hess$
```

The 'WeightedSoftmaxLossWithEdgePenalty' function represents a significant advancement in addressing class imbalance and the specific challenges posed by edge classes in classification tasks. By applying an extra penalty to errors involving these classes, the function aims to improve model sensitivity and accuracy in scenarios where edge class errors are particularly costly or significant.

2.13. Adaptive Weighted Softmax Loss Function

The Adaptive Weighted Softmax Loss function introduces a dynamic approach to weight assignment in the context of classification errors, combined with a focal loss modification [72] to further enhance model performance, especially in scenarios with imbalanced classes or hard-to-classify instances.

This function comprises two significant components: the computation of adaptive weights based on class-specific errors and the incorporation of the focal loss concept into the softmax loss calculation.

2.13.1. Computing Adaptive Weights

Adaptive weights [73] are computed based on the average error of each class. An additional focus is given to specified class pairs with a focus multiplier. In this implementation, the focus_pairs are set to [(0, 2), (2, 0)], indicating a heightened emphasis on the errors between these class pairs. The focus_multiplier is set to 5.0, significantly amplifying the weight adjustment for these pairs.

2.13.2. Focal Loss Modification

The focal loss [74,75] modification adjusts the contribution of each sample to the loss based on the correctness of its classification, thereby focusing more on difficult or misclassified samples.

2.13.3. Algorithm of Adaptive Weighted Softmax Loss Function

The pseudocode for computing adaptive weights is presented in Algorithm 8, and the Adaptive Weighted Softmax Loss Function is outlined in Algorithm 9.

Algorithm 8 Compute Adaptive Weights

```
Require: $y, \hat{y}, focus\_pairs, focus\_multiplier$
  $n\_classes \leftarrow columns(\hat{y})$
  Initialize $class\_errors$ as a zero vector of length $n\_classes$
  for $i = 0$ to $n\_classes - 1$ do
      $class\_indices \leftarrow$ indices where $y = i$
      $class\_errors[i] \leftarrow$ mean absolute error of $\hat{y}[class\_indices, i]$ from 1
  end for
  Normalize $class\_errors$ by its maximum value
  Initialize $weights$ as an empty dictionary
  for $i = 0$ to $n\_classes - 1$ do
      for $j = i$ to $n\_classes - 1$ do
          if $i = j$ then
              $weights[(i,j)] \leftarrow 0.1 + 0.1 \times class\_errors[i]$
          else
              $avg\_error \leftarrow (class\_errors[i] + class\_errors[j])/2$
              $weights[(i,j)] \leftarrow weights[(j,i)] \leftarrow avg\_error$
              if $(i,j)$ in $focus\_pairs$ or $(j,i)$ in $focus\_pairs$ then
                  $weights[(i,j)] \leftarrow weights[(j,i)] \leftarrow avg\_error \times focus\_multiplier$
              end if
          end if
      end for
  end forreturn $weights$
```

Algorithm 9 Adaptive Weighted Softmax Loss with Focal Modification

```
Require: $y, \hat{y}, gamma = 2$
  Compute $\hat{y}$ using the softmax transformation
  Initialize $grad$ and $hess$ as zero matrices of the same shape as $\hat{y}$
  Compute adaptive weights using $y$ and $\hat{y}$
  for $i = 0$ to $len(y) - 1$ do
      for $j = 0$ to $columns(\hat{y}) - 1$ do
          $weight \leftarrow$ weight for the pair $(\min(y[i], j), \max(y[i], j))$
          $prob \leftarrow \hat{y}[i, j]$
          $focal\_mod \leftarrow (1 - prob)^{g}amma$
          $grad[i, j] \leftarrow focal\_mod \times weight \times (prob - (y[i] == j))$
          $hess[i, j] \leftarrow focal\_mod \times weight \times prob \times (1 - prob) + 0.02$
      end for
  end forreturn $grad, hess$
```

The Adaptive Weighted Softmax Loss function with Focal Modification offers a sophisticated approach to handling complex classification scenarios. By dynamically adjusting weights according to class-specific errors and incorporating the focal loss mechanism, this function aims to improve model accuracy and robustness, particularly in cases of class imbalance or where certain classes are more challenging to classify correctly.

3. Numerical Experiments

All features obtained through the five extraction methods were integrated into a single dataset. However, a distinction was made to identify the drill to which each feature set belongs. This design choice was critical for maintaining the integrity of the data and ensuring that the classification model accurately reflects the characteristics specific to each drill.

In our experiments, we opted not to apply feature selection for the XGBoost classifier. This decision was grounded in several considerations specific to the nature of our data and the inherent characteristics of XGBoost. Here are the key reasons for this approach:

- Inherent Feature Handling Capabilities of XGBoost: XGBoost is well-known for its ability to handle a large number of features efficiently. It automatically assigns a score to each feature based on its importance, effectively doing an internal form of feature selection during the learning process. Given this capability, we believed that an additional explicit feature selection step might not significantly improve the performance.
- Complexity of the Data: The dataset in our study was complex, with features extracted from five different methods. Each feature potentially carried unique information that could be crucial for accurate classification. We wanted to ensure that the model had access to all available information before making any decision to exclude features.
- Avoiding Potential Loss of Information: Feature selection, especially if not done carefully, can lead to the loss of important information that could be valuable for the model. Given the critical nature of our task—drill-wear analysis—we could not afford to lose potentially subtle yet important signals that might be present in the less prominent features.
- Computational Resources: We had access to sufficient computational resources to handle the complexity and size of our dataset without the need for feature reduction. This allowed us to train the XGBoost model on the full set of features without concerns about computational efficiency or training time.
- Ensuring Model Robustness: By using the complete set of features, we aimed to develop a model that is robust and can generalize well across different scenarios. Reducing the feature space might lead to a model that is overly optimized for the specific characteristics of the training data, potentially reducing its effectiveness on new, unseen data.

Thus, considering the strengths of XGBoost in handling high-dimensional data, along with the desire to preserve the integrity and richness of our dataset, we concluded that feature selection was not a necessary preprocessing step for our specific application. The results of our experiments would provide insights into the effectiveness of this approach.

In the course of this study, a conscious decision was made to not engage in hyperparameter optimization. This approach was grounded in our intent to demonstrate that modifications to the loss function alone can optimize the classifier for the specific problem at hand. Hyperparameter optimization, by its nature, is not deterministic and often involves elements of randomness. In our methodology, we sought to ensure that the comparison of different loss function methods was as reliable and consistent as possible. Therefore, to maintain stability and determinism in our experiments, elements prone to nondeterminism and randomness, such as feature selection and hyperparameter optimization, were deliberately omitted. This approach allowed us to focus on the direct impact of the loss function modifications, ensuring that any observed improvements in classifier performance could be attributed specifically to these changes rather than to variations in other parameters of the model.

Industry experts who designed the entire experimental setup imposed a specific process for validation (cross-validation). The experiment involved five drills, and a five-fold cross-validation was applied. However, in this particular experiment, a 'fold' corresponds to one drill. This means that in each step of the validation process, one drill is used as the test set, while data from the remaining four drills are utilized for training the model. This procedure is repeated five times, once for each drill.

Industry experts mandated this five-stage cross-validation process. It was specifically chosen to ensure that the model's performance is robust and generalizable across different drills. This approach also addresses potential biases that might arise if the model is trained and tested on data from the same drill. By separating the test and training sets based on the drill, the validation process provides a more reliable assessment of the model's effectiveness and accuracy in real-world conditions.

Additionally, it is essential to note that the custom loss functions were implemented using Python 3.9. For the classifier, the implementation of XGBoost used was DLMC XGBoost [76]. This library is an optimized distributed Gradient Boosting code renowned for its efficiency, flexibility, and portability. It implements machine learning algorithms under the Gradient Boosting framework, providing a parallel tree boosting technique, also known as GBDT (Gradient Boosting Decision Tree) or GBM (Gradient Boosting Machine). The use of DLMC XGBoost in our experiments ensures that our model not only leverages a state-of-the-art algorithm for boosted tree learning but also benefits from the latest advancements in distributed computing for large-scale machine learning tasks.

All experiments were performed on hardware (Ubuntu operating system) with the following specifications:

- Processor: AMD RYZEN THREADRIPPER 2990WX (32C 64T) 4.3 GHz;
- Motherboard: AsRock X399 TAICHI;
- Memory: 8 × ADATA XPG SPECTRIX DDR4 16 GB D41 3000 MHz (128 GB RAM);
- Graphics Card: 2 × Nvidia GeForce RTX Titan 24 GB GDDR6 (48 GB RAM);
- Drive SSD: 2 × WD BLACK 1 TB WDS100T3X0C1TB (PCIE);
- Drive HDD: 1 × WD RED PRO 8 TB WD8003FFBX 3.5″ (SATA);
- Power Supply: BE QUIET! DARK POWER PRO 11 1000 W;
- Cooling: BE QUIET! Silent Loop BW003 280 mm;
- Network: 10GbE SFP+.

4. Results and Discussion

The experimental evaluation of various XGBoost loss functions, with a particular focus on the reduction in critical errors in classification tasks, has been presented below. Table 3 presents a comprehensive comparative analysis of these loss functions.

Table 3. Comparative analysis of different XGBoost loss functions focusing on the reduction in critical errors.

Loss Function for XGBoost	**Green-Red Error**	**Red-Green Error**	**Total Critical Errors**	**Accuracy**	**Time**
Default Softmax Loss	552	598	1150	64.29%	199 s
Weighted Softmax Loss V1	683	457	1140	62.73%	215 s
Weighted Softmax Loss V2	603	458	1061	61.44%	214 s
Weighted Softmax Loss V3	544	429	973	61.08%	217 s
Weighted Softmax Loss V4	500	422	922	60.52%	223 s
Weighted Softmax Loss V5	436	417	853	59.73%	218 s
Weighted Softmax Loss With Edge Penalty	460	413	873	60.59%	232 s
Adaptive Weighted Softmax Loss	406	318	724	56.08%	473 s

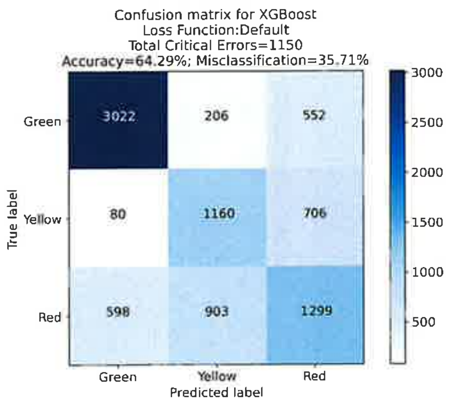
The Default Softmax Loss Function, serving as a baseline, showed a total of 1150 critical errors (552 Green-Red errors and 598 Red-Green errors), with an overall accuracy of 64.29%. This standard metric sets the stage for evaluating the enhancements achieved through the implementation of weighted loss functions.

A noticeable trend in the data is the gradual reduction in the total count of critical errors as we progress from Weighted Softmax Loss Variant 1 to Variant 5. Notably, Variant 1 exhibited a slightly higher Green-Red error count (683) compared to the Default Softmax Loss but a reduced Red-Green error count (457), leading to a marginally decreased overall accuracy of 62.73%.

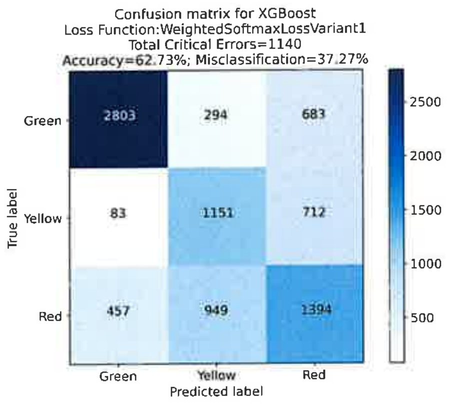
Variants 2 through 5 consistently showed a decrease in both Green-Red and Red-Green errors, culminating in Variant 5, which recorded the lowest total critical errors (853) among these variants. Despite this improvement, there was a noticeable decrease in overall accuracy, with Variant 5 reaching 59.73%.

The Weighted Softmax Loss with Edge Penalty and the Adaptive Weighted Softmax Loss introduced more nuanced approaches to handling critical errors. The former resulted in a total of 873 critical errors and an accuracy of 60.59%, while the latter achieved the most substantial reduction in critical errors, totaling 724 with an accuracy of 56.08%.

These findings are further elucidated in the confusion matrices presented in Figures 5–8. These matrices provide a detailed view of the classification performance across different classes, offering a visual representation of the trade-offs between specific error types and overall accuracy.



(**a**)



(**b**)

Figure 5. Confusion matrix analysis for XGBoost with Default Loss Function and Variant 1: (**a**) presents the classification outcomes using the Default Loss Function, while (**b**) illustrates results from Weighted Softmax Loss Function Variant 1.

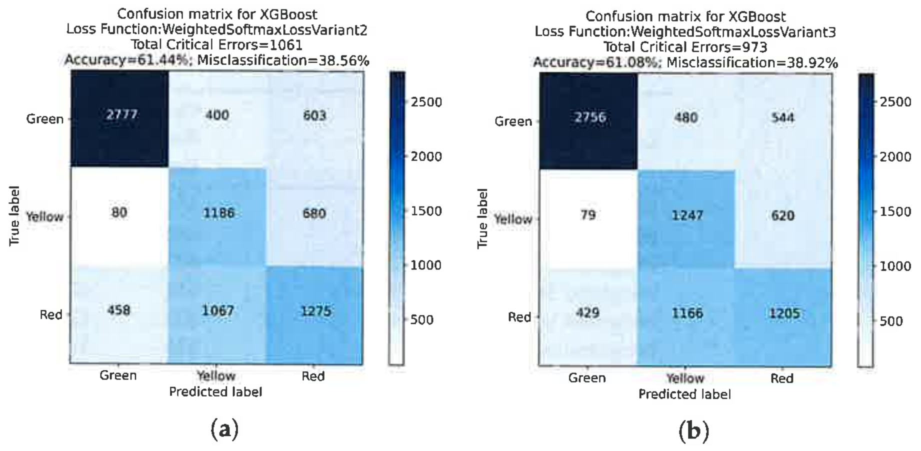


Figure 6. Confusion matrix analysis for XGBoost with Weighted Softmax Loss Function Variant 2 and Variant 3: (**a**) presents the classification outcomes using the Weighted Softmax Loss Function Variant 2, while (**b**) illustrates results from Weighted Softmax Loss Function Variant 3.

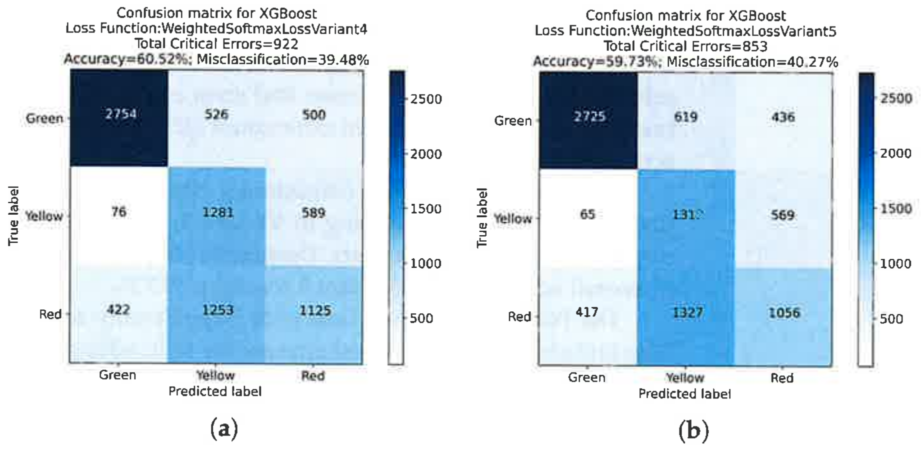


Figure 7. Confusion matrix analysis for XGBoost with Weighted Softmax Loss Variant 4 and Variant 5: (**a**) presents the classification outcomes using the Weighted Softmax Loss Variant 4, while (**b**) illustrates results from Weighted Softmax Loss Variant 5.

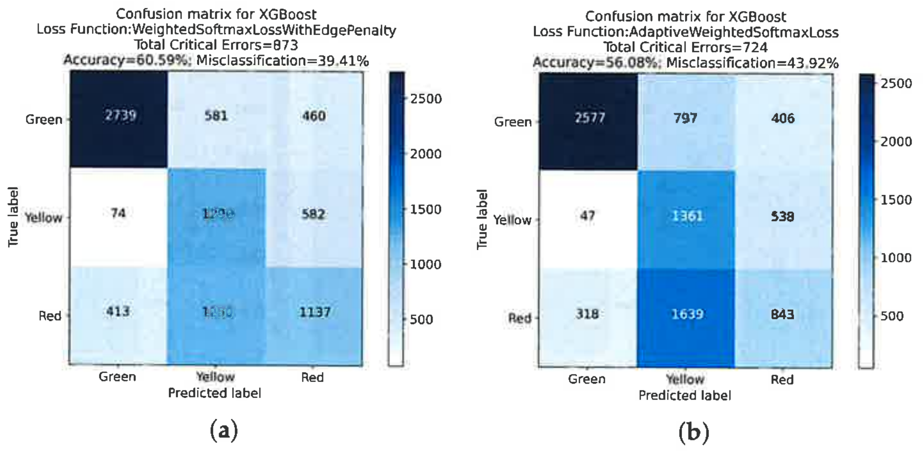


Figure 8. Confusion matrix analysis for XGBoost with Weighted Softmax Loss Function with Edge Penalty and Adaptive Weighted Softmax Loss Function: (**a**) presents the classification outcomes using the Weighted Softmax Loss Function with Edge Penalty, while (**b**) illustrates results from Adaptive Weighted Softmax Loss Function.

The exploration of different loss functions in the XGBoost algorithm reveals a complex balance between reducing critical errors and maintaining overall classification accuracy. The insights gained from this analysis, mainly when considered alongside the detailed results in the confusion matrices, are invaluable for guiding the choice of loss function in practical applications, depending on the specific requirements of the classification task at hand.

Another crucial aspect to consider is the computational time, which reflects the efficiency and practical applicability of these algorithms in real-world scenarios.

Looking at the "Time" column in Table 3, we note a consistent trend in the computational time required for each variant of the loss function. The Default Softmax Loss function, serving as our baseline, required 199 s. As we progressed through the Weighted Softmax Loss Variants 1 through 5, there was a slight but consistent increase in computation time, culminating in Variant 5, which took 218 s. This increment can be attributed to the additional complexity introduced by the weighted mechanism in these variants.

The Weighted Softmax Loss with Edge Penalty and the Adaptive Weighted Softmax Loss Function exhibited more significant increases in computational time, 232 s and 473 s, respectively. The additional time for the Edge Penalty variant might be due to the extra computations required for applying penalties to the edge classes. However, the Adaptive Weighted Softmax Loss Function, which showed the most substantial reduction in critical errors, also required the longest computational time. This increase in time is likely due to the dynamic nature of the adaptive weights calculation and the incorporation of the focal loss mechanism, which adds to the computational complexity.

Below, a detailed analysis of the performance metrics for different loss functions applied to the XGBoost algorithm is presented. These metrics include precision, sensitivity (recall), F1 score, and specificity, evaluated for each class (Green, Yellow, and Red).

The detailed analysis presented in Tables 4–11 plays a crucial role in understanding the nuances of our study. These tables collectively provide a comprehensive evaluation of the performance metrics for various loss functions applied in the XGBoost algorithm, offering insights into their effectiveness in the context of drill-wear analysis for melamine-faced chipboard.

Table 4. Performance metrics for XGBoost using Default Softmax Loss Function.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	81.68%	79.95%	80.80%	85.71%
Yellow	51.12%	59.61%	55.04%	83.15%
Red	50.80%	46.39%	48.50%	78.03%

Table 5. Performance metrics for XGBoost using Weighted Softmax Loss Function Variant 1.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	83.85%	74.15%	78.70%	88.62%
Yellow	48.08%	59.15%	53.04%	81.11%
Red	49.98%	49.79%	49.88%	75.64%

Table 6. Performance metrics for XGBoost using Weighted Softmax Loss Function Variant 2.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	83.77%	73.47%	78.28%	88.66%
Yellow	44.70%	60.95%	51.58%	77.71%
Red	49.84%	45.54%	47.59%	77.59%

Table 7. Performance metrics for XGBoost using Weighted Softmax Loss Function Variant 3.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	84.44%	72.91%	78.25%	89.30%
Yellow	43.10%	64.08%	51.54%	74.98%
Red	50.87%	43.04%	46.62%	79.67%

Table 8. Performance metrics for XGBoost using Weighted Softmax Loss Function Variant 4.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	84.69%	72.86%	78.33%	89.51%
Yellow	41.86%	65.83%	51.18%	72.96%
Red	50.81%	40.18%	44.87%	80.98%

Table 9. Performance metrics for XGBoost using Weighted Softmax Loss Function Variant 5.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	84.97%	72.09%	78.00%	89.84%
Yellow	40.27%	67.42%	50.42%	70.43%
Red	51.24%	37.71%	43.45%	82.45%

Table 10. Performance metrics for XGBoost using Weighted Softmax Loss Function with Edge Penalty.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	73.37%	72.46%	72.91%	81.08%
Yellow	70.72%	66.29%	68.44%	89.89%
Red	38.30%	40.61%	39.42%	71.88%

Table 11. Performance metrics for XGBoost using Adaptive Weighted Softmax Loss Function.

Class	Precision	Sensitivity	F1 Score	Specificity
Green	69.80%	68.17%	68.98%	79.71%
Yellow	75.03%	69.94%	72.39%	90.15%
Red	27.91%	30.11%	28.97%	68.72%

Table 4 presents the baseline performance metrics using the Default Softmax Loss Function. This sets the stage for comparison with other loss function variants. It is observed that while the Green class shows relatively high precision and sensitivity, the Yellow and Red classes have lower values, indicating potential areas for improvement.

Tables 5–9 depict the results for Weighted Softmax Loss Function Variants 1 through 5. A noticeable trend across these variants is the increasing precision for the Green class but with a decrease in sensitivity, suggesting a more conservative classification. The Yellow and Red classes, however, show mixed results, indicating the complexities involved in balancing precision and recall for these classes.

In Table 10, we observe the performance metrics for the Weighted Softmax Loss Function with Edge Penalty. This variant shows a significant improvement in precision and recall for the Yellow class, suggesting a more balanced classification across the classes. However, the Red class continues to be challenging in terms of lower precision and recall.

Lastly, Table 11 focuses on the Adaptive Weighted Softmax Loss Function. While this variant demonstrates a notable shift in performance with a significant improvement for the Yellow class, it highlights the ongoing challenge with the Red class, which exhibits the lowest precision and recall.

The discussions surrounding these tables emphasize the trade-offs inherent in optimizing loss functions for multi-class classification tasks. The variation in performance metrics across different classes underscores the complexity of achieving a balance between precision and recall and highlights the importance of selecting a loss function that aligns with the specific needs and objectives of the task at hand.

In conclusion, the exploration of different XGBoost loss functions reveals significant variations in performance metrics across different classes. While some loss functions improve precision or recall for certain classes, they often do so at the expense of other metrics or classes. This underscores the challenge of finding a one-size-fits-all loss function for multi-class classification tasks and highlights the importance of choosing a loss function that aligns with specific objectives and class priorities of the given task.

4.1. Advantages and Limitations of the Proposed Work

4.1.1. Advantages

The proposed approach of applying custom loss functions in XGBoost for drill-wear analysis in melamine-faced chipboard presents several significant advantages:

- Enhanced Accuracy for Critical Classes: By customizing loss functions, the model demonstrates improved classification accuracy, especially for critical edge classes, which are essential in industrial applications for maintaining production quality.
- Flexibility in Addressing Class Imbalance: The adaptive nature of the proposed loss functions effectively addresses the challenges posed by imbalanced datasets, a common issue in real-world scenarios.
- Context-Specific Model Optimization: Tailoring loss functions according to the specific needs of the application allows for a more nuanced and effective model compared to standard approaches.
- Improved Decision-Making in Industrial Settings: The refined predictions offered by the model facilitate better decision-making processes, crucial in high-stakes industrial environments such as furniture manufacturing.

4.1.2. Limitations

Despite its advantages, the proposed approach also exhibits certain limitations that should be considered:

- Increased Computational Complexity: Custom loss functions, particularly those involving adaptive weights and focal modifications, demand higher computational resources, potentially impacting the efficiency of the model.
- Overfitting Risks: The model's heightened sensitivity to specific classes might lead to overfitting, particularly when dealing with small or highly specific datasets.
- Dependency on Expert Knowledge: The effectiveness of the approach relies heavily on domain expertise for accurately defining and tuning the custom loss functions.
- Limited Generalizability: While effective in the specific context of drill-wear analysis, the approach may not be directly applicable or as effective in other domains without significant modifications.

5. Conclusions

This study advances the understanding of custom loss functions in the XGBoost algorithm, specifically tailored for the complex task of drill-wear analysis in melamine-faced chipboard processing. The exploration reveals that the Adaptive Weighted Softmax Loss Function markedly reduces critical errors in multi-class classification, setting a new benchmark compared to the baseline Default Softmax Loss Function. The investigation into various Weighted Softmax Loss Function variants illustrates a nuanced trade-off between minimizing critical errors and maintaining overall classification accuracy. This highlights the complexity and intricacies involved in optimizing these aspects simultaneously.

While the Adaptive Weighted Softmax Loss Function demonstrates superior performance in reducing critical errors, it comes at the cost of increased computational time. This trade-off between performance and efficiency is a critical consideration in practical applications. For scenarios where time efficiency is a priority, simpler models such as the Default Softmax or the earlier variants of the Weighted Softmax Loss might be preferable.

However, for applications where the reduction in critical errors is paramount and computational resources are not a constraint, the Adaptive Weighted Softmax Loss Function presents an effective solution despite its higher computational demand.

Furthermore, the research underscores the significance of selecting loss functions that align with the specific challenges and objectives of classification tasks, especially in scenarios with high sensitivity to critical errors. For practitioners in the field of industrial machine learning, this study offers valuable insights into the choice of loss functions based on the specific requirements and priorities of the task at hand.

In summary, the findings of this research contribute significantly to the practical application of machine learning in industrial settings. By demonstrating the effectiveness of custom loss functions in addressing specialized classification challenges, this study paves the way for the development of more accurate and reliable classification models in various industrial applications.

Author Contributions: Conceptualization, M.B.; data acquisition, A.J.; formal analysis, M.B.; investigation M.B.; methodology, M.B.; software, M.B.; project administration J.K.; supervision J.K.; visualization B.Ś.; writing—original draft, M.B. and J.K.; writing—review and editing, all the authors (M.B., J.K., B.Ś. and A.J.). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset available on request from the authors.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Byrne, G.; Dornfeld, D.; Inasaki, I.; Ketteler, G.; König, W.; Teti, R. Tool condition monitoring (TCM)—The status of research and industrial application. *CIRP Ann.* **1995**, *44*, 541–567. [CrossRef]
2. Liu, T.I.; Jolley, B. Tool condition monitoring (TCM) using neural networks. *Int. J. Adv. Manuf. Technol.* **2015**, *78*, 1999–2007. [CrossRef]
3. Mohamed, A.; Hassan, M.; M'Saoubi, R.; Attia, H. Tool condition monitoring for high-performance machining systems—A review. *Sensors* **2022**, *22*, 2206. [CrossRef]
4. Schueller, A.; Saldaña, C. Indirect Tool Condition Monitoring Using Ensemble Machine Learning Techniques. *J. Manuf. Sci. Eng.* **2023**, *145*, 011006. [CrossRef]
5. Lemaster, R.L.; Tee, L.B.; Dornfeld, D.A. Monitoring tool wear during wood machining with acoustic emission. *Wear* **1985**, *101*, 273–282. [CrossRef]
6. Kun, W.; Qi, S.; Cheng, W.; Chunjie, L. Influence of pneumatic pressure on delamination factor of drilling medium density fiberboard. *Wood Res.* **2015**, *60*, 429–440.
7. Szwajka, K.; Trzepieciński, T. Effect of tool material on tool wear and delamination during machining of particleboard. *J. Wood Sci.* **2016**, *62*, 305–315. [CrossRef]
8. Szwajka, K.; Trzepieciński, T. An examination of the tool life and surface quality during drilling melamine faced chipboard. *Wood Res.* **2017**, *62*, 307–318.
9. Śmietańska, K.; Podziewski, P.; Bator, M.; Górski, J. Automated monitoring of delamination factor during up (conventional) and down (climb) milling of melamine-faced MDF using image processing methods. *Eur. J. Wood Wood Prod.* **2020**, *78*, 613–615. [CrossRef]
10. Swiderski, B.; Antoniuk, I.; Kurek, J.; Bukowski, M.; Gorski, J.; Jegorowa, A. Tool Condition Monitoring for the Chipboard Drilling Process Using Automatic, Signal-based Tool State Evaluation. *BioResources* **2022**, *17*, 5349–5371. [CrossRef]
11. Bukowski, M.; Kurek, J.; Antoniuk, I.; Jegorowa, A. Decision Confidence Assessment in Multi-Class Classification. *Sensors* **2021**, *21*, 3834. [CrossRef]
12. Król, P.M.; Szymona, K. Methodology evaluation of computer vision small-dimension hole localization. *Wood Mater. Sci. Eng.* **2023**, *18*, 184–192. [CrossRef]
13. Osowski, S.; Kurek, J.; Kruk, M.; Górski, J.; Hoser, P.; Wieczorek, G.; Jegorowa, A.; Wilkowski, J.; Śmietańska, K.; Kossakowska, J. Developing automatic recognition system of drill wear in standard laminated chipboard drilling process. *Bull. Pol. Acad. Sci. Tech. Sci.* **2016**, *64*, 633–640. [CrossRef]
14. Kuo, R. Multi-sensor integration for on-line tool wear estimation through artificial neural networks and fuzzy neural network. *Eng. Appl. Artif. Intell.* **2000**, *13*, 249–261. [CrossRef]
15. Jemielniak, K.; Urbański, T.; Kossakowska, J.; Bombiński, S. Tool condition monitoring based on numerous signal features. *Int. J. Adv. Manuf. Technol.* **2012**, *59*, 73–81. [CrossRef]

16. Panda, S.; Singh, A.; Chakraborty, D.; Pal, S. Drill wear monitoring using back propagation neural network. *J. Mater. Process. Technol.* **2006**, *172*, 283–290. [CrossRef]
17. Jegorowa, A.; Górski, J.; Kurek, J.; Kruk, M. Initial study on the use of support vector machine (SVM) in tool condition monitoring in chipboard drilling. *Eur. J. Wood Wood Prod.* **2019**, *77*, 957–959. [CrossRef]
18. Nasir, V.; Cool, J.; Sassani, F. Intelligent machining monitoring using sound signal processed with the wavelet method and a self-organizing neural network. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3449–3456. [CrossRef]
19. Nasir, V.; Sassani, F. A review on deep learning in machining and tool monitoring: Methods, opportunities, and challenges. *Int. J. Adv. Manuf. Technol.* **2021**, *115*, 2683–2709. [CrossRef]
20. Ibrahim, I.; Khairuddin, A.; Abu Talip, M.; Arof, H.; Yusof, R. Tree species recognition system based on macroscopic image analysis. *Wood Sci. Technol.* **2017**, *51*, 431–444. [CrossRef]
21. Kurek, J.; Swiderski, B.; Jegorowa, A.; Kruk, M.; Osowski, S. Deep learning in assessment of drill condition on the basis of images of drilled holes. In Proceedings of the 8th International Conference on Graphic and Image Processing (ICGIP 2016), Tokyo, Japan, 29–31 October 2016; Volume 10225, pp. 375–381. [CrossRef]
22. Kurek, J.; Wieczorek, G.; Kruk, B.; Jegorowa, A.; Osowski, S. Transfer learning in recognition of drill wear using convolutional neural network. In Proceedings of the 18th International Conference on Computational Problems of Electrical Engineering (CPEE), Kutna Hora, Czech Republic, 11–13 September 2017; pp. 1–4. [CrossRef]
23. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Świderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J. Classifiers Ensemble of Transfer Learning for Improved Drill Wear Classification using Convolutional Neural Network. *Mach. Graph. Vis.* **2019**, *28*, 13–23. [CrossRef]
24. Kurek, J.; Antoniuk, I.; Górski, J.; Jegorowa, A.; Świderski, B.; Kruk, M.; Wieczorek, G.; Pach, J.; Orłowski, A.; Aleksiejuk-Gawron, J. Data Augmentation Techniques for Transfer Learning Improvement in Drill Wear Classification Using Convolutional Neural Network. *Mach. Graph. Vis.* **2019**, *28*, 3–12. [CrossRef]
25. Wieczorek, G.; Chlebus, M.; Gajda, J.; Chyrowicz, K.; Kontna, K.; Korycki, M.; Jegorowa, A.; Kruk, M. Multiclass image classification using gans and cnn based on holes drilled in laminated chipboard. *Sensors* **2021**, *21*, 8077. [CrossRef] [PubMed]
26. Jegorowa, A.; Kurek, J.; Antoniuk, I.; Dołowa, W.; Bukowski, M.; Czarniak, P. Deep learning methods for drill wear classification based on images of holes drilled in melamine faced chipboard. *Wood Sci. Technol.* **2021**, *55*, 271–293. [CrossRef]
27. Jegorowa, A.; Antoniuk, I.; Kurek, J.; Bukowski, M.; Dołowa, W.; Czarniak, P. Time-efficient approach to drill condition monitoring based on images of holes drilled in melamine faced chipboard. *BioResources* **2020**, *15*, 9611. [CrossRef]
28. Kurek, J.; Antoniuk, I.; Świderski, B.; Jegorowa, A.; Bukowski, M. Application of Siamese Networks to the Recognition of the Drill Wear State Based on Images of Drilled Holes. *Sensors* **2020**, *20*, 6978. [CrossRef]
29. Kurek, J.; Osowski, S. Support vector machine for fault diagnosis of the broken rotor bars of squirrel-cage induction motor. *Neural Comput. Appl.* **2010**, *19*, 557–564. [CrossRef]
30. Jegorowa, A.; Kurek, J.; Kruk, M.; Górski, J. The Use of Multilayer Perceptron (MLP) to Reduce Delamination during Drilling into Melamine Faced Chipboard. *Forests* **2022**, *13*, 933. [CrossRef]
31. Jegorowa, A.; Górski, J.; Kurek, J.; Kruk, M. Use of nearest neighbors (k-NN) algorithm in tool condition identification in the case of drilling in melamine faced particleboard. *Maderas. Cienc. Y Tecnol.* **2020**, *22*, 189–196. [CrossRef]
32. Kurek, J. Hybrid Approach Towards the Assessment of a Drill Condition Using Deep Learning and the Support Vector Machine. In Proceedings of the 22nd International Computer Science and Engineering Conference (ICSEC), Chiang Mai, Thailand, 21–24 November 2018; pp. 1–5. [CrossRef]
33. Amal Asselman, M.K.; Aammou, S. Enhancing the prediction of student performance based on the machine learning XGBoost algorithm. *Interact. Learn. Environ.* **2023**, *31*, 3360–3379. [CrossRef]
34. Ben Jabeur, S.; Stef, N.; Carmona, P. Bankruptcy Prediction using the XGBoost Algorithm and Variable Importance Feature Engineering. *Comput. Econ.* **2023**, *61*, 715–741. [CrossRef]
35. Wang, T.; Bian, Y.; Zhang, Y.; Hou, X. Classification of earthquakes, explosions and mining-induced earthquakes based on XGBoost algorithm. *Comput. Geosci.* **2023**, *170*, 105242. [CrossRef]
36. Lei, Y.; Shen, Z.; Tian, F.; Yang, X.; Wang, F.; Pan, R.; Wang, H.; Jiao, S.; Kou, W. Fire risk level prediction of timber heritage buildings based on entropy and XGBoost. *J. Cult. Herit.* **2023**, *63*, 11–22. [CrossRef]
37. Ibrahim, A.A.; Elzaridi, K.M.A. XGBoost algorithm for orecasting electricity consumption of Germany. *AURUM J. Eng. Syst. Archit.* **2023**, *7*, 99–108. [CrossRef]
38. Naik, D.; Kiran, R. A Novel Sensitivity-based Method for Feature Selection. *J. Big Data* **2021**, *8*, 128 . [CrossRef]
39. Asheghi, R.; Hosseini, S.A.; Saneie, M.; Shahri, A.A. Updating the neural network sediment load models using different sensitivity analysis methods: A regional application. *J. Hydroinform.* **2020**, *22*, 562–577. [CrossRef]
40. Yeung, D.; Cloete, I.; Shi, D.; Ng, W. *Sensitivity Analysis for Neural Networks*; Springer: Berlin/Heidelberg, Germany, 2010. [CrossRef]
41. Abbaszadeh Shahri, A.; Shan, C.; Larsson, S. A Novel Approach to Uncertainty Quantification in Groundwater Table Modeling by Automated Predictive Deep Learning. *Nat. Resour. Res.* **2022**, *31*, 1351–1373. [CrossRef]
42. Ghaderi, A.; Abbaszadeh Shahri, A.; Larsson, S. A visualized hybrid intelligent model to delineate Swedish fine-grained soil layers using clay sensitivity. *CATENA* **2022**, *214*, 106289. [CrossRef]
43. Czarniak, P.; Szymanowski, K.; Panjan, P. Characteristic of the wear of a tool coating based on amorphous carbon during chipboard milling. *Ann. Wars. Univ. Life Sci. SGGW For. Wood Technol.* **2020**, *111*, 53–59. [CrossRef]

44. Czarniak, P.; Szymanowski, K.; Panjan, P. Influence of the microstructure of tool coatings based on Ti and Al on the blunting process during chipboard processing. *Ann. Wars. Univ. Life Sci. SGGW For. Wood Technol.* **2020**, *112*, 54–59. [CrossRef]
45. Wieloch, G.; Szymanowski, K. Condition of edges of particle board laminated after saws on a panel saw. *Trieskové A Beztrieskové Obrábanie Dreva = Chip Chipless Woodwork. Process.* **2018**, *11*, 197–204.
46. Pfleiderer. Meblarstwo i Wykończenie Wnętrz. 2023. Available online: https://www.pfleiderer.pl/en/produkty/MEBLARSTWO-I-WYKONCZENIE-WNETRZ/plyty-laminowane (accessed on 17 December 2023).
47. Kruk, M.; Kurek, J.; Osowski, S.; Koktysz, R.; Swiderski, B.; Markiewicz, T. Ensemble of classifiers and wavelet transformation for improved recognition of Fuhrman grading in clear-cell renal carcinoma. *Biocybern. Biomed. Eng.* **2017**, *37*, 357–364. [CrossRef]
48. PyWavelets Development Team. PyWavelets Documentation. 2023. Available online: https://pywavelets.readthedocs.io (accessed on 17 December 2023).
49. Grossmann, A.; Kronland-Martinet, R.; Morlet, J., Reading and understanding continuous wavelet transforms. In *Wavelets*; Springer: Berlin/Heidelberg, Germany, 1990; pp. 2–20. [CrossRef]
50. Büssow, R. An algorithm for the continuous Morlet wavelet transform. *Mech. Syst. Signal Process.* **2007**, *21*, 2970–2979. [CrossRef]
51. ImageNet Project. ImageNet. 2023. Available online: https://www.image-net.org (accessed on 18 December 2023).
52. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385. [CrossRef]
53. Chandola, Y.; Virmani, J.; Bhadauria, H.; Kumar, P. Chapter 4—End-to-end pre-trained CNN-based computer-aided classification system design for chest radiographs. In *Deep Learning for Chest Radiographs*; Academic Press: Cambridge, MA, USA, 2021; pp. 117–140. [CrossRef]
54. Nazir, M.; Jan, Z.; Sajjad, M. Facial expression recognition using histogram of oriented gradients based transformed features. *Clust. Comput.* **2018**, *21*, 539–548. [CrossRef]
55. Déniz, O.; Bueno, G.; Salido, J.; De la Torre, F. Face recognition using Histograms of Oriented Gradients. *Pattern Recognit. Lett.* **2011**, *32*, 1598–1603. [CrossRef]
56. Jafari, F.; Basu, A. Saliency-Driven Hand Gesture Recognition Incorporating Histogram of Oriented Gradients (HOG) and Deep Learning. *Sensors* **2023**, *23*, 7790. [CrossRef] [PubMed]
57. Dias, C.G.; Rodrigues, K.L.; Menegasse, N.C.; Alves, W.A.L.; Silva, L.C.d. Histogram of Oriented Gradients for Rotor Speed Estimation in Three-Phase Induction Motors. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 7503811. [CrossRef]
58. Bhattarai, B.; Subedi, R.; Gaire, R.R.; Vazquez, E.; Stoyanov, D. Histogram of Oriented Gradients meet deep learning: A novel multi-task deep network for 2D surgical image semantic segmentation. *Med. Image Anal.* **2023**, *85*, 102747. [CrossRef]
59. Chen, T.; Guestrin, C. *Xgboost: A Scalable Tree Boosting System*; Association for Computing Machinery: New York, NY, USA, 2016. [CrossRef]
60. Friedman, J.; Hastie, T.; Tibshirani, R. Additive logistic regression: A statistical view of boosting (With discussion and a rejoinder by the authors). *Ann. Stat.* **2000**, *28*, 337–407. [CrossRef]
61. Friedman, J. Greedy Function Approximation: A Gradient Boosting Machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [CrossRef]
62. Friedman, J. Stochastic gradient boosting. *Comput. Stat. Data Anal.* **2002**, *38*, 367–378. [CrossRef]
63. Sharma, N.; Anju; Juneja, A. Extreme Gradient Boosting with Squared Logistic Loss Function. In *Machine Intelligence and Signal Analysis*; Tanveer, M., Pachori, R.B., Eds.; Springer: Singapore, 2019; pp. 313–322. [CrossRef]
64. Python API Reference of Xgboost. Available online: https://xgboost.readthedocs.io/en/stable/python/python_api.html (accessed on 1 December 2023).
65. Mohiuddin, G.; Lin, Z.; Zheng, J.; Wu, J.; Li, W.; Fang, Y.; Wang, S.; Chen, J.; Zeng, X. Intrusion Detection using hybridized Meta-heuristic techniques with Weighted XGBoost Classifier. *Expert Syst. Appl.* **2023**, *232*, 120596. [CrossRef]
66. Vadhwani, D.; Thakor, D. Prediction of extent of damage in vehicle during crash using improved XGBoost model. *Int. J. Crashworthiness* **2023**, *28*, 299–305. [CrossRef]
67. Tian, J.; Tsai, P.W.; Zhang, K.; Cai, X.; Xiao, H.; Yu, K.; Zhao, W.; Chen, J. Synergetic Focal Loss for Imbalanced Classification in Federated XGBoost. *IEEE Trans. Artif. Intell.* **2023**, 1–13. [CrossRef]
68. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 13–17 August 2016; KDD '16; pp. 785–794. [CrossRef]
69. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; Chapter 6.2.2.3 Softmax Units for Multinoulli Output Distributions; MIT Press: Cambridge, MA, USA, 2016; pp. 180–184. [CrossRef]
70. Mushava, J.; Murray, M. Flexible loss functions for binary classification in gradient-boosted decision trees: An application to credit scoring. *Expert Syst. Appl.* **2024**, *238*, 121876. [CrossRef]
71. Legate, G.; Caccia, L.; Belilovsky, E. Re-weighted softmax cross-entropy to control forgetting in federated learning. *arXiv* **2023**, arXiv:2304.05260. [CrossRef]
72. Wang, C.; Deng, C.; Wang, S. Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost. *Pattern Recognit. Lett.* **2020**, *136*, 190–197. [CrossRef]
73. Ye, M.; Zhu, L.; Li, X.; Ke, Y.; Huang, Y.; Chen, B.; Yu, H.; Li, H.; Feng, H. Estimation of the soil arsenic concentration using a geographically weighted XGBoost model based on hyperspectral data. *Sci. Total Environ.* **2023**, *858*, 159798. [CrossRef]
74. Wang, Y.; Wang, X.; Chen, B.; Zhang, R.; She, W.; Tian, Z. A combination of XGBoost and FocalLoss-based cable aging state assessment method. In Proceedings of the 5th International Conference on Information Science, Electrical, and Automation Engineering (ISEAE 2023), Wuhan, China, 24–26 March 2023; SPIE: Bellingham, DC, USA, 2023; Volume 12748, pp. 724–730.

75. Fan, C.; Li, C.; Peng, Y.; Shen, Y.; Cao, G.; Li, S. Fault Diagnosis of Vibration Sensors Based on Triage Loss Function-Improved XGBoost. *Electronics* **2023**, *12*, 4442. [CrossRef]
76. GitHub—Dmlc/Xgboost: Scalable, Portable and Distributed Gradient Boosting (GBDT, GBRT or GBM) Library, for Python, R, Java, Scala, C++ and More. Runs on Single Machine, Hadoop, Spark, Dask, Flink and DataFlow—github.com. Available online: https://github.com/dmlc/xgboost (accessed on 1 January 2024).

Warszawa, 27-01-2025

Michał Bukowski
michal_bukowski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A. Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092. https://doi.org/10.3390/s24041092
mój indywidualny udział w jej powstaniu polegał na:

1. Koncepcji.
2. Metodologii.
3. Badaniu (ang. research).
4. Analizie formalnej.
5. Implementacji.
6. Wizualizacji.
7. Edycji i korekcji artykułu.

[illegible]
Podpis

Warszawa, 27-01-2025

Jarosław Kurek
jaroslaw_kurek@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A. Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092. https://doi.org/10.3390/s24041092
mój indywidualny udział w jej powstaniu polegał na:

1. Przygotowaniu pierwszego szkicu.
2. Nadzorze.
3. Administracji.

[illegible]
Podpis

Warszawa, 27-01-2025

Bartosz Świderski
bartosz_swiderski@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A. Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092. https://doi.org/10.3390/s24041092
mój indywidualny udział w jej powstaniu polegał na:

1. Walidacji.

Bartosz Świderski
Podpis

Warszawa, 27-01-2025

Albina Jegorowa
albina_jegorowa@sggw.edu.pl

Rada Dyscypliny
Informatyka Techniczna i Telekomunikacja

Szkoły Głównej Gospodarstwa
Wiejskiego w Warszawie

Oświadczenie o współautorstwie

Niniejszym oświadczam, że w pracy:
Bukowski, M.; Kurek, J.; Świderski, B.; Jegorowa, A. Custom Loss Functions in XGBoost Algorithm for Enhanced Critical Error Mitigation in Drill-Wear Analysis of Melamine-Faced Chipboard. Sensors 2024, 24, 1092. https://doi.org/10.3390/s24041092
mój indywidualny udział w jej powstaniu polegał na:

1. Akwizycji danych.
2. Zarządzaniu danymi.

A. Jegorowa

Podpis