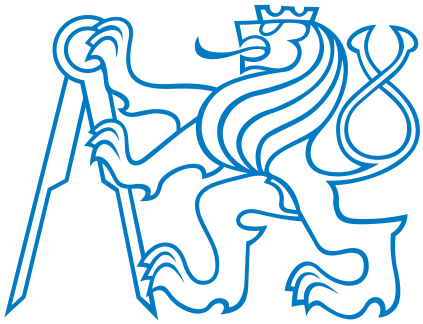
Classification of construction and demolition waste fragments using computer vision

Tomáš Zbíral

supervisor: **doc. Ing. Václav Nežerka, Ph.D.**



Czech Technical University in Prague

Faculty of Civil Engineering

Department of Physics

A thesis submitted for the degree of

Bachelor

May, 2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Zbíral** Jméno: **Tomáš** Osobní číslo: **507774**

Fakulta/ústav: **Fakulta stavební**

Zadávající katedra/ústav: **Katedra fyziky**

Studijní program: **Geodézie a kartografie**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Classification of construction and demolition waste fragments using computer vision

Název bakalářské práce anglicky:

Classification of construction and demolition waste fragments using computer vision

Pokyny pro vypracování:

1) Stručně představte teorii strojového učení s důrazem na její význam v počítačovém vidění pro klasifikaci stavebního a demoličního odpadu (CDW).
2) Detailně popište následující modely strojového učení:
- CNN: obecný přehled, výhody při rozpoznávání obrazu,
- GB decision trees: vysvětlení principu, práce se znaky (features),
- MLP: vysvětlení principů, využití pro rozpoznávání obrazu.
3) Porovnejte modely na základě přesnosti, rychlosti a práce s texturami.
4) Vyjmenujte kritéria: přesnost, výpočetní efektivita, škálovatelnost.
5) Sepište výsledky a diskuzi výsledků modelování.
6) Shrňte závěry s důrazem na využití technologie pro třídění CDW a navrhněte "research gap" pro budoucí výzkum.

Seznam doporučené literatury:

[1] Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly Media, Inc.
[2] Dong, Z et al. (2022). Computer vision to recognize construction waste compositions: A novel boundary-aware transformer (BAT) model. Journal of Environmental Management, 305, 114405.
[3] Lin, K. et al. (2022). Deep convolutional neural networks for construction and demolition waste classification: VGGNet structures, cyclical learning rate, and knowledge transfer. Journal of Environmental Management, 318, 115501.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Václav Nežerka, Ph.D. katedra fyziky FSv

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **09.02.2024** Termín odevzdání bakalářské práce: **20.05.2024**

Platnost zadání bakalářské práce: **01.07.2024**

doc. Ing. Václav Nežerka, Ph.D. podpis vedoucí(ho) práce	prof. Ing. Jiří Novák, Ph.D. podpis vedoucí(ho) ústavu/katedry	prof. Ing. Jiří Máca, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. Datum převzetí zadání

Podpis studenta

Acknowledgements

I want to thank the supervisor of this thesis, Václav Nežerka, for his patient consultations and advice during my participation in his projects and during writing this thesis. I would like to thank Jan Trejbal for his contribution during the research. Finally, I would like to thank my family and friends who supported me during my studies.

Honesty Declaration

I declare that this master thesis has been carried out by me and only with the use of materials that are stated in the literature sources.

April 19$^{\text{th}}$, 2024
Tomáš Zbíral

Funding

This work was supported by the European Union's Horizon Europe Framework Programme (call HORIZON-CL4-2021-TWIN-TRANSITION-01-11) under grant agreement No. 101058580 and project RECONMATIC (Automated solutions for sustainable and circular construction and demolition waste management), the Technology Agency of the Czech Republic, grant agreement No. S03010302 (Development of efficient tools to minimize production of construction and demolition waste, its monitoring and reuse).

Abstract

Improper sorting of construction and demolition waste (CDW) leads to significant environmental and economic implications, including inefficient resource use and missed recycling opportunities. This thesis adrress this by developing a machine-learning-assisted procedure for recognizing CDW fragments using an RGB camera. Our approach uniquely leverages selected feature extraction, enhancing classification speed and accuracy. We employed three classifiers: convolutional neural network (CNN), gradient boosting (GB) decision trees, and multi-layer perception (MLP). Notably, our method's extraction of selected features for GB and MLP outperformed the traditional CNN in terms of speed and accuracy, especially for challenging samples with similar textures. Specifically, while convolution resulted in an overall accuracy of 85.9%, our innovative feature extraction approach yielded accuracies up to 92.3%. This study's findings have significant implications for the future of CDW management, offering a pathway for efficient and accurate waste sorting, fostering sustainable resource use, and reducing the environmental impact of CDW disposal. Supplementary materials, including datasets, codes, and models, are provided, promoting transparency and reproducibility.

Abstrakt

Nesprávné třídění stavebního a demoličního odpadu (CDW) vede k významným environmentálním a ekonomickým důsledkům, včetně neefektivního využívání zdrojů a nevyužití možností recyklace. Tato práce řeší tento problém vývojem postupu pro rozpoznávání fragmentů CDW pomocí RGB kamery za použití strojového učení. Náš přístup jedinečným způsobem využívá extrakci vybraných vlastností, čímž zvyšuje rychlost a přesnost klasifikace. Použili jsme tři klasifikátory: konvoluční neuronovou síť (CNN), rozhodovací stromy s gradientním posilováním (GB) a vícevrstvý perceptron (MLP). Je pozoruhodné, že extrakce vybraných vlastností naší metodou pro GB a MLP překonala tradiční CNN z hlediska rychlosti a přesnosti, zejména u náročných vzorků s podobnými texturami. Konkrétně, zatímco konvoluce vedla k celkové přesnosti 85,9%, náš inovativní přístup k extrakci vlastností přinesl přesnost až 92,3%. Výsledky této studie mají významný dopad na budoucnost nakládání s CDW, protože nabízejí cestu k efektivnímu a přesnému třídění odpadu, podporují udržitelné využívání zdrojů a snižují dopad likvidace CDW na životní prostředí. K dispozici jsou doplňkové materiály, včetně souborů dat, kódů a modelů, které podporují transparentnost a reprodukovatelnost.

The complete code developed during the work on this thesis is available on codeocean capsule.

Contents

Chapter 1

Introduction to Machine Learning for Classification Tasks

Machine learning (ML) has experienced a massive boom over the last few years as it appears helpful in various industries and parts of life. Based on the data report provided by the AI Index Steering Committee [1], it can be concluded that there is great interest in ML. Figure 1.1 demonstrates that artificial intelligence (AI) has become popular within the research community as well as in different fields of engineering including geodesy [2, 3], waste management [4], and commercial activities (Figure 1.2). The exponential growth started around 2013, however, even in 2005 the number of papers was double that in 1998. AI provides us with fast methods of achieving goals that would otherwise be tedious. While the prospects of these technologies appear promising, users must possess a comprehensive understanding of the algorithms to select the most appropriate one for their specific requirements, thereby optimizing outcomes. This section covers ML algorithms' fundamentals and will describe their pros and cons. In addition, algorithms that have been used in this study will be given special attention so they can be fully understood.

ML algorithms are commonly divided into three basic types: (i) supervised, (ii) unsupervised, and (iii) reinforcement learning.

1.1 Supervised learning

Supervised learning is a subcategory of ML characterized by using labeled data to train an algorithm. Labeled data comprise samples annotated with the correct output, facilitating the algorithm's learning process. In this study, a material name was paired with each image of a material. Labels can be of either a quantitative nature or represent a category. Considering that we can only receive those two types of labels, supervised learning is a well-suited solution for basically three types of tasks [5]: (i) classification, (ii) regression, and (iii) ranking.

Classification tasks are typically distinguished as binary, a multi-label classification, and several unique labels characterize it. The classification task aims to predict a category label characterized by discrete values based on input data. Sorting construction and demolition waste (CDW) is a multi-label classification problem as its goal is to distinguish a waste of different

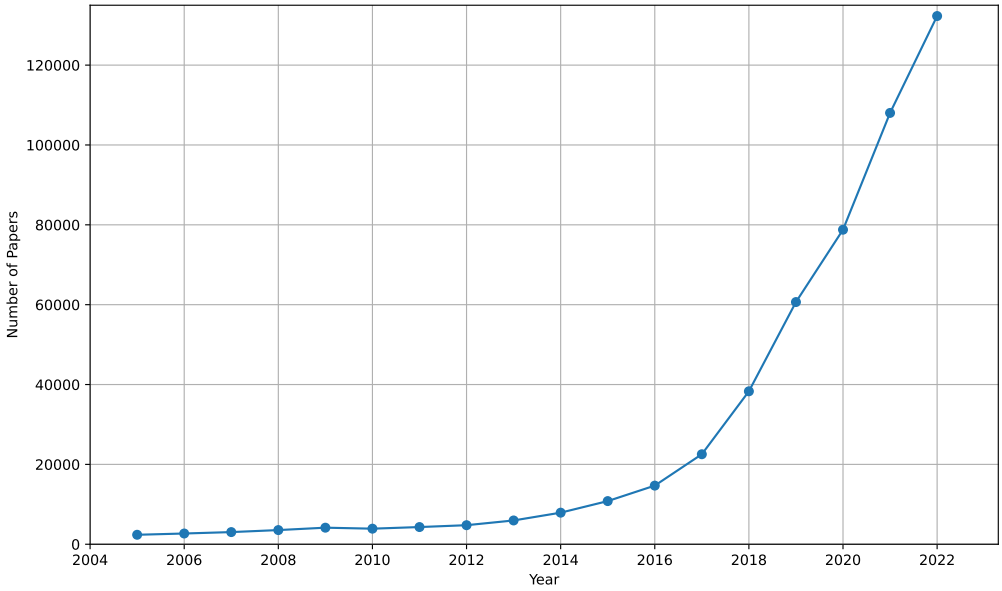


Figure 1.1: Number of papers published with keywords of AI, artificial intelligence, ML, and deep learning by year based on the data from Web of Science.

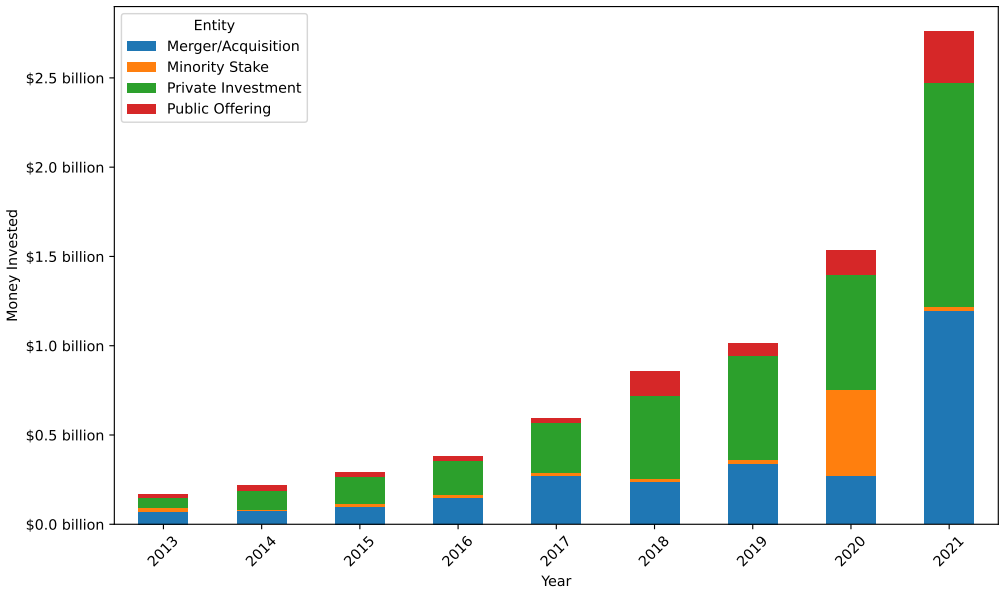


Figure 1.2: Anual global corporate investment in AI based on data from AIIndex report 2023 [1].

materials, i.e., classify material so it can be sorted and efficiently reused. The core idea of a classification task is to recognize a pattern that distinguishes between labels. Multiple ML models used for classification tasks have been used over time. Among the most used there are the following models:

- Decision trees
- Random forests
- Support vector machines (SVM)
- K-nearest neighbors
- Neural networks/deep learning
- Gradient boosting machines

This study employs three distinct approaches to address the specified task. Application of gradient boosting decision tree, multi-layer perceptron (MLP) [6] a type of artificial neural network (ANN), and convolutional neural network (CNN) [7] a foundational algorithm within deep learning (DL). A gradient boosting (GB) decision tree combines decision tree algorithms with GB techniques, and its mechanism will be described further. Regression tasks and their application are beyond this study's scope and will not be described in depth. It should be noted that, unlike classification tasks, regression tasks typically involve labels represented by real values [5].

1.1.1 Decision trees

The structure of a decision tree model helps to understand its principle (Figure 1.3). This Diagram consists of three types of nodes and branches. The root node represents the first decision node in a model. A decision node is a node that contains an if-statement that tests given data for a defined property and leads the algorithm toward another decision node or a leaf node. A leaf node is a node where a label is assigned to the data. Branches represent the outcome of a decision node, and they can represent both positive and negative outcomes of a decision node. During the classification process, data are tested in decision nodes and assigned to a leaf node that represents a label. However, it is even more critical to understand the process of training a decision tree model.

The decision tree tries to find the best if-statement for a node by evaluating how good a split was. Various indicators are used for this purpose, such as the Gini index, Shannon entropy, chi-square, and reduction in variance. Scikit-learn decision tree model [8] have been used in this study, and since this model uses mostly the Gini index and Shannon entropy to decide how good the split was, those will be described mathematically. If a target is a classification outcome taking on values $0, 1, \ldots, k-1$ for node m, let

$$p_{mk} = \frac{1}{n_m} \sum_{y \in Q_m} I\left(y = k\right) \tag{1.1}$$

be the proportion of class k observations in node m. If m is a decision node, predict probe for this region is set to p_{mk}, where Q_m represents data at node m with n_m samples. The Gini index is then defined as

$$G\left(Q_m\right) = \sum_k p_{mk}\left(1 - p_{mk}\right) \tag{1.2}$$

and the Shannon entropy is defined as

$$E\left(Q_m\right) = -\sum_k p_{mk} \log\left(p_{mk}\right). \tag{1.3}$$

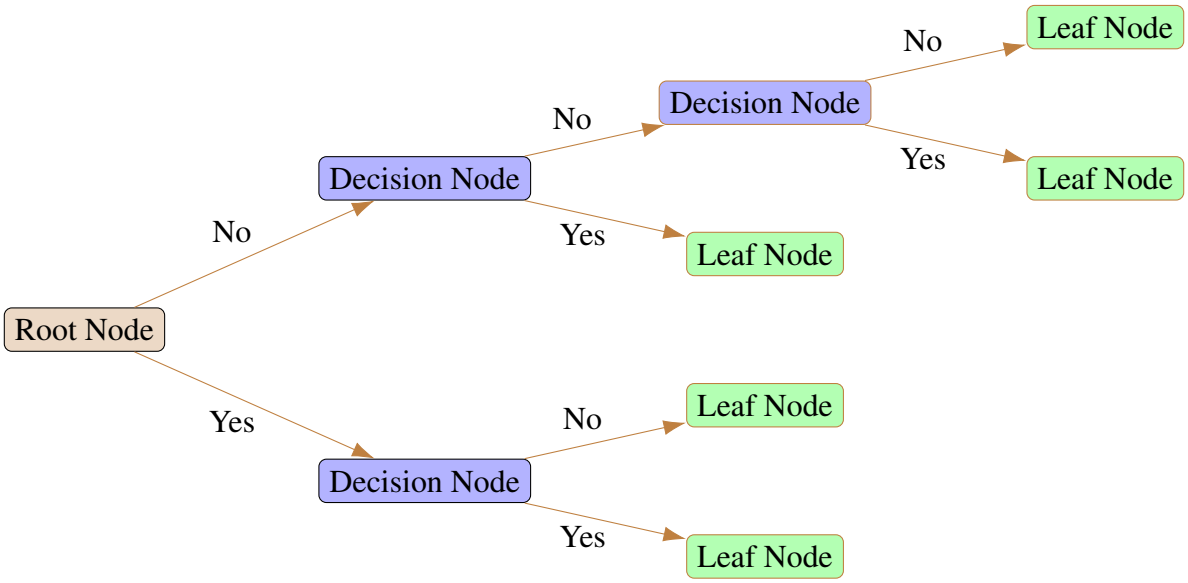


Figure 1.3: Diagram with fundamental parts of a decision tree model.

Every model has its pros and cons. Decision trees are renowned for their interpretability as they can process both categorical and numerical data. They are versatile, and we can use them both for classification tasks and regression tasks. On the other hand, many decision tree models employ greedy algorithms, selecting the optimal split for the current node, potentially compromising the effectiveness of future splits. Decision tree models are susceptible to overfitting, and they may be slow to make predictions when it comes to vast and complex datasets. To mitigate the limitations associated with greedy algorithms, GB techniques were employed in the algorithms utilized in this study.

1.1.2 Gradient boosting

GB is a powerful machine learning technique that constructs prediction models using an ensemble of weak predictive models, typically decision trees. This approach builds the model incrementally in a stage-wise fashion and optimizes an arbitrary differentiable loss function, thereby enhancing the model's generalization capability. The fundamental principle of GB involves sequentially adding predictive models to the ensemble, each designed to correct its predecessor. Let $f\left(x_1, x_2, \ldots, x_n\right)$ be an n-variable function. Gradient ∇ is then defined as:

$$\nabla f = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right] \tag{1.4}$$

Applying the gradient ∇ to a multivariable function indicates the direction of the steepest ascent within the function's domain. Each predictor is trained using the gradient of the loss function. Building a prediction model begins with a base model, usually a very simple model, and computes residuals. Subsequent decision trees are then trained to predict these residuals, and those residuals are added to the earlier models' predictions, making the final predictions more accurate. This procedure is repeated multiple times until stopping criteria are met, usually several trees or tolerable errors in predictions. There are numerous GB algorithms, among the most used: (i) XGBoost [9], (ii) LightGBM [10], (iii) CatBoost [11], (iv) AdaBoost [12], (v) HistoGradient Boosting [8]. The effectiveness of GB is attributed to its broad applicability and adaptability to various loss functions, making it a popular choice among models. However, careful tuning of hyperparameters and controlling overfitting are essential for leveraging the full potential of GB models to deliver highly accurate predictions.

1.1.3 Artificial neural networks and deep learning

ANN is a computational model inspired by the functioning of the human brain [13]. The human brain consists of neurons and their connections, forming a biological neural network. An ANN is comprised of units termed 'neurons' with associated weights and biases facilitating information processing. An ANN is composed of multiple layers: an input layer, one or more hidden layers, and an output layer (Figure 1.5). DL's definition is unclear. Generally, ANN with multiple hidden layers can be considered a DL algorithm. DL is typically characterized by layered structure and its ability to learn feature hierarchies without human intervention. The model learns to identify the right features in data by itself, progressing through layers to understand increasingly abstract concepts.

Before delving into the description of these concepts, it is essential first to define the notation of ANN elements. The general notation for activation values is $a_i^{(l)}$, where the subscript i denotes the neuron's position within its layer, and the superscript l denotes the layer of the neuron. It is important to note that the superscript represents layer indexing, not exponentiation. The notation for weights is $w_{j,k}^{(l)}$, where j denotes the position of the originating neuron within its layer, k denotes the position of the target neuron within its layer, and l denotes the layer to which the weight is applied. Notation of biases is similar to the notation of an activation value, $b_i^{(l)}$ is the bias for a neuron i in a layer l. Activation value is a value distributed by its neuron as an output into the next layers. Activation values are calculated as

$$a_i^{(l)} = \sigma\left(\sum_{j=0}^{n-1} a_j^{(l-1)} w_{j,i}^{(l)} - b_i^{(l)}\right), \tag{1.5}$$

where σ is an activation function that maps input values to the interval $(0, 1)$, where n represents the number of neurons in a layer $l-1$. It is important to note that there are multiple activation functions, including the sigmoid function, the rectified linear unit (ReLU) function, and the hyperbolic tangent function ($\tanh$), each offering distinct characteristics beneficial for different ANN configurations. Weight represents how strong the connection between two neurons is and is applied while calculating the activation of a target neuron. A bias is a constant added to the product of activations and weights, shifting the activation function along the y-axis, enabling the model to fit the data better. The process of classification with ML algorithms works on that

principle. We calculate all the activation values, and once we reach an output layer, the neuron with the highest activation value in the output layer is considered the result. During training, the goal is to identify the optimal set of parameters (weights and biases) that maximize the model's accuracy on the validation dataset.

At the beginning of the training process, all weights and biases are initialized to random values before input into the ANN alongside the training data. The efficiency of those weights and biases is evaluated using the cost function. Let a_i^{c} denote the calculated activation values in the output layer, and let a_i^{t} denote the target activation values (also known as the ground truth). The cost value for one training example is then calculated as

$$C = \sum_{i=0}^{n-1} \left(a_i^{\mathrm{c}} - a_i^{\mathrm{t}}\right)^2. \tag{1.6}$$

The cost value for the entire training dataset is computed as the mean of all individual costs. It follows that greater accuracy in the results correlates with a lower cost value. Since the cost value is derived as the mean of all individual costs, minimizing this function is expected to yield improved accuracy across the training dataset. To minimize the function most efficiently, weights and biases are adjusted in the direction opposite to the gradient, denoted by $-\nabla$ defined by Equation 1.4. Gradient estimation is predominantly performed using the backpropagation method, which has demonstrated efficiency across a wide range of tasks [14].

1.1.3.1 Backpropagation

Backpropagation employs the chain rule [15] in ANNs to propagate errors backward from the output to input layers. Consider the notation for the weighted sum component of the activation value as defined in Equation (1.5)

$$z_j^{(l)} = \sum_{j=0}^{n-1} a_j^{(l-1)} w_{j,k}^{(l)} - b_k^{(l)}. \tag{1.7}$$

The chain rule quantifies the influence of weight adjustments on the cost value. On Figure 1.4, we can see that weight $w_{j,k}^{(l)}$ influence $z_j^{(l)}$ directly, which cause change in $a_k^{(l)}$ value and this change in $a_k^{(l)}$ consequently affects the cost value, C_k. This allows to calculate change in C_k caused by change to $w_{j,k}^{(l)}$.

$$\frac{\partial C_k}{\partial w_{j,k}^{(l)}} = \frac{\partial z_j^{(l)}}{\partial w_{j,k}^{(l)}} \frac{\partial a_k^{(l)}}{\partial z_j^{(l)}} \frac{\partial C_k}{\partial a_k^{(l)}}$$

$$\frac{\partial C_k}{\partial a_k^{(l)}} = 2\left(a_k^{(l)} - y_k\right)$$

$$\frac{\partial a_k^{(l)}}{\partial z_j^{(l)}} = \sigma'\left(z_j^{(l)}\right)$$

$$\frac{\partial z_j^{(l)}}{\partial w_{j,k}^{(l)}} = a_j^{(l-1)}$$

This differentiation yields the final expression for the partial derivative of the cost with respect to a weight

$$\frac{\partial C_k}{\partial w_{j,k}^{(l)}} = a_j^{(l-1)} \sigma'\left(z_j^{(l)}\right) 2\left(a_k^{(l)} - y_k\right). \tag{1.8}$$

This way, the influence of the change of $w_{j,k}^{(l)}$ to the cost of one training example C_k is computed. Since the cost value is of one iteration for the training dataset is computed as an average of costs for all training examples, its derivative $\frac{\partial C}{\partial w_{j,k}^{(l)}}$ requires averaging expression above for whole training dataset. This is formally expressed as

$$\frac{\partial C}{\partial w_{j,k}^{(l)}} = \frac{1}{n_\mathrm{e}} \sum_{k=0}^{n_\mathrm{e}-1} \frac{\partial C_k}{\partial w_{j,k}^{(l)}}, \tag{1.9}$$

where n_e is number of training examples. Using the same idea as we have used for partial derivative $\frac{\partial C_k}{\partial w_{j,k}^{(l)}}$, we can then express partial derivatives with respect to $b_k^{(l)}$ and $a_j^{(l-1)}$.

$$\frac{\partial C_k}{\partial b_k^{(l)}} = 1\sigma'\left(z_j^{(l)}\right) 2\left(a_k^{(l)} - y_k\right) \tag{1.10}$$

$$\frac{\partial C_k}{\partial a_j^{(l-1)}} = w_{j,k}^{(l)} \sigma'\left(z_j^{(l)}\right) 2\left(a_i^{(l)} - y_k\right) \tag{1.11}$$

Iterating through the ANN in reverse order allows for the computation of the cost function's sensitivity to all weights and biases.

1.1.4 Multi-layer perceptron (MLP)

The MLP is a fundamental ML algorithm from the class of ANNs characterized by multiple fully connected hidden layers of neurons between the input and output layers. Those hidden

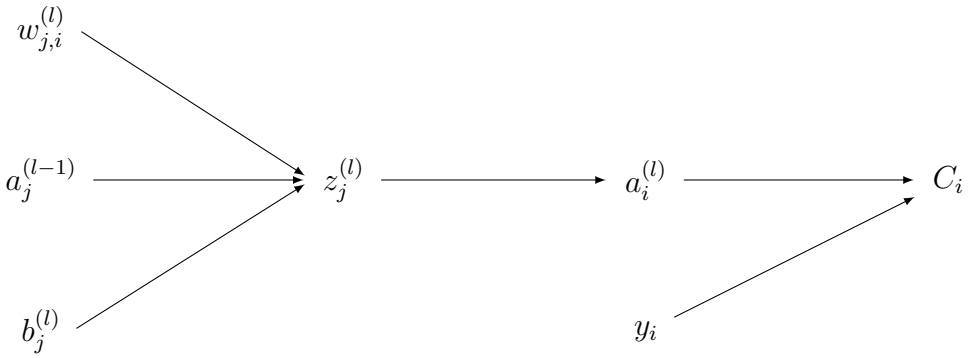


Figure 1.4: Chain rule application for computing cost function of a single neuron.

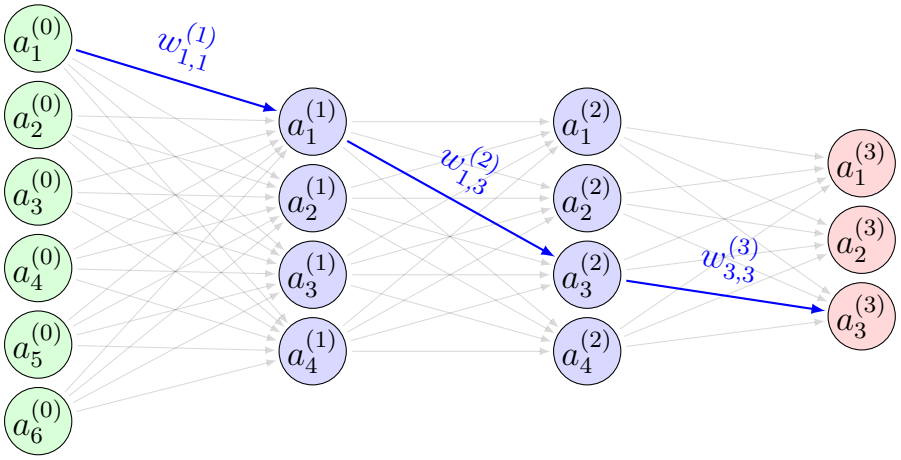


Figure 1.5: ANN consisting of 4 layers with applied general notation.

layers allow MLPs to capture complex nonlinear relationships in data by leveraging multiple layers of computation. This capability and MLPs wide range of use led to the decision to implement this algorithm for CDW recognition. The architecture of an MLP is often described by the number of neurons in each layer (*hidden_layer_sizes* in scikit-learn MLPClassifier [8]), with the connectivity pattern typically being fully connected. This means every neuron in one layer connects to every neuron in the subsequent layer.

Evaluation of data and training of the MLP is achieved by the algorithms described in Section 1.1.3. While powerful, MLPs come with challenges, such as the risk of overfitting, especially with very deep networks or insufficient data. Techniques such as dropout or early stopping come in handy to overcome those limitations. Early stopping is a method of regularization used to avoid overfitting in ANN training. It involves monitoring model performance on a validation dataset after each epoch. If the model's performance on a validation dataset starts to deteriorate (validation error increases or stops decreasing for several epochs). It is possible to save the model after every epoch to choose the best model that is not overfitted. Furthermore, machine learning frameworks and their models come with multiple hyperparameters (number of neurons in each hidden layer described above, learning rate, regularization, and others). Despite those limitations, MLPs is one of the DL algorithms, perhaps the most intuitive one. It illustrates the power of layered ANNs that can work with complex and high-dimensional data.

1.1.5 Convolutional neural networks (CNN)

Over the last couple of years, DL techniques have made tremendous progress in computer vision, especially in object recognition [16]. CNNs are a special kind of DL that is particularly efficient for processing grid-like data such as images. CNNs are mainly known for their high efficiency in recognizing patterns directly from pixels of images with minimal preprocessing. CNNs power to recognize patterns directly from images was first shown in an ILSVRC 2012 challenge, where CNN of the name AlexNet achieved a top-5 error rate of around 16.4% while the runner-up–ANN called SuperVision had a top-5 error rate of around 26.2% [17]. CNNs can automatically learn spatial hierarchies of features from input images, which is possible using three main types of layers: convolutional layers, pooling layers, and fully connected layers.

1.1.5.1 Convolutional layers

Formally, for complex-valued functions f, g defined on set $\mathbb{Z}$, discrete convolution (such as convolution performed in computer vision) of f and g is given by [18]:

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]\, g[n-m]. \tag{1.12}$$

Convolutional layers are the cornerstone of CNNs, and they play a critical role in CNN's ability to learn from grid-like data. Convolution performed on matrices (grids) is well described in Figure 1.6. For each convolutional layer, there is a kernel filter is applied to the input matrix. Kernel slides across the width and depth of the input matrix, computing the dot product of kernel values and input values, producing a matrix that gives the responses of convolution at every position. In convolutional layers, adjusting kernel size, striding, and padding is common practice. Striding controls the amount of overlap between neighboring regions processed by the kernel. Padding refers to adding values around the border of the input, ensuring that all the pixels in the input layer are considered equally and allowing us to control output size. Following convolution, outputs are passed through a non-linear function such as ReLU or similar functions. This enables a CNN to understand more complex non-linear patterns. Training of a CNN is done using backpropagation described in Section 1.1.3.1 with a slight change for convolutional and pooling layers. Each kernel or pooling window in those layers is optimized, but the principle has not been changed.

CNN's role is to learn the spatial hierarchies of features from the input data (whole images for our case). The first layers tend to detect edges, textures, or other simple patterns, and deeper layers combine those patterns to detect more complex patterns, such as shapes or specific objects. This principle has been well described in a seminal paper [19] where the authors presented Figure 1.7. Unlike in fully connected networks, each neuron in a convolutional layer is connected only to a small region of the input volume. The spatial extent of this connectivity is equivalent to the filter size. This design takes advantage of the spatial structure in the input data, ensuring that the network architecture assumes that only nearby input features are relevant for computing the output. This makes CNNs less prone to overfitting and allows CNNs to be deeper with fewer parameters. By learning to recognize patterns anywhere in an input image, convolutional layers give CNNs a significant advantage in tasks involving images and grid-like data, enabling efficient extraction of features.

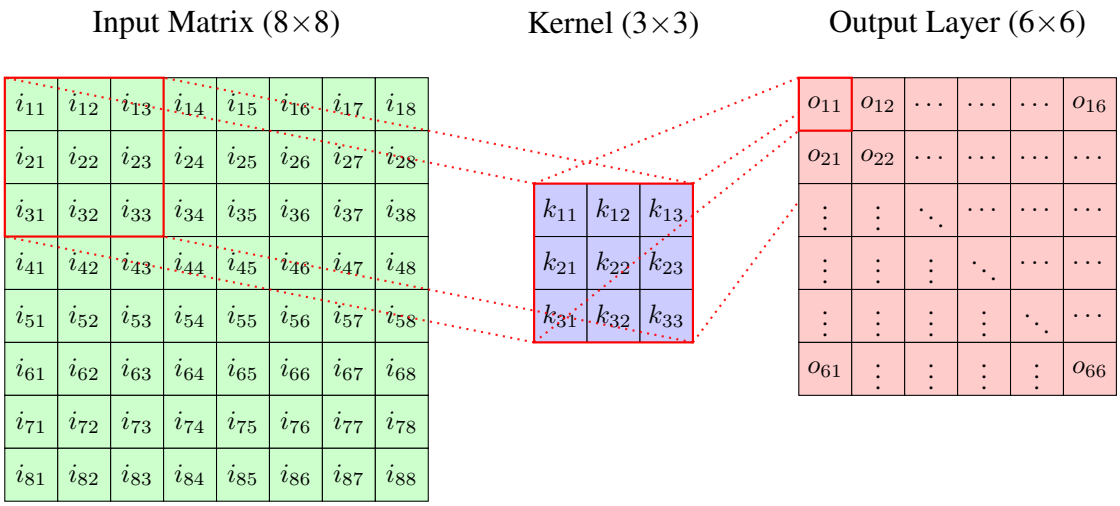


Figure 1.6: The principle of calculating the output of a convolutional layer using the kernel.

1.1.5.2 Pooling layers

There are many challenges to face during the training of CNNs, such as long training time and overfitting. Pooling layers are usually placed between successive convolutional layers within CNN architecture. They serve multiple purposes: They reduce the size of the input data for subsequent layers, drastically reducing computation cost and memory usage of a user, they help to achieve invariance to minor translations such as small shifts, rotations, etc., which then resolve in models being more robust to variations in the position of features and they help with feature aggregation. Max pooling helps aggregate the most present feature in a layer, and average pooling helps aggregate the average features. Correctly applying pooling layers helps to abstract higher-level features from the input.

The pooling operation slides a window of a size we can set as a hyperparameter over the input matrix performing one of the pooling types. For each subwindow in a matrix, one value is calculated from multiple values (4 for 2×2 window, 9 for 3×3 window, etc.), significantly reducing the matrix size. Three widely used pooling types are max pooling, average pooling, and L2 norm pooling. The process of max pooling is shown in Figure 1.8. When using max pooling, the max value is retrieved from a window and is passed to the output afterward. With average pooling, the output is calculated as an average value from the window, and with L2 norm pooling, the output value is computed as the square root of the sum of the squares of the elements in the window.

1.1.5.3 Fully connected layers

In CNNs, fully connected layers serve as a classifier for features extracted using convolutional and pooling layers. The operational principles of these layers are detailed in Section 1.1.3. Typically, fully connected layers require vectorized input; however, the output from convolutional and pooling layers is grid-like. To reconcile this discrepancy, the grid-like data are flattened into a vector form, thus fulfilling the input requirements of the fully connected layers. This transformation ensures that the spatial information encoded in the grid is converted into a

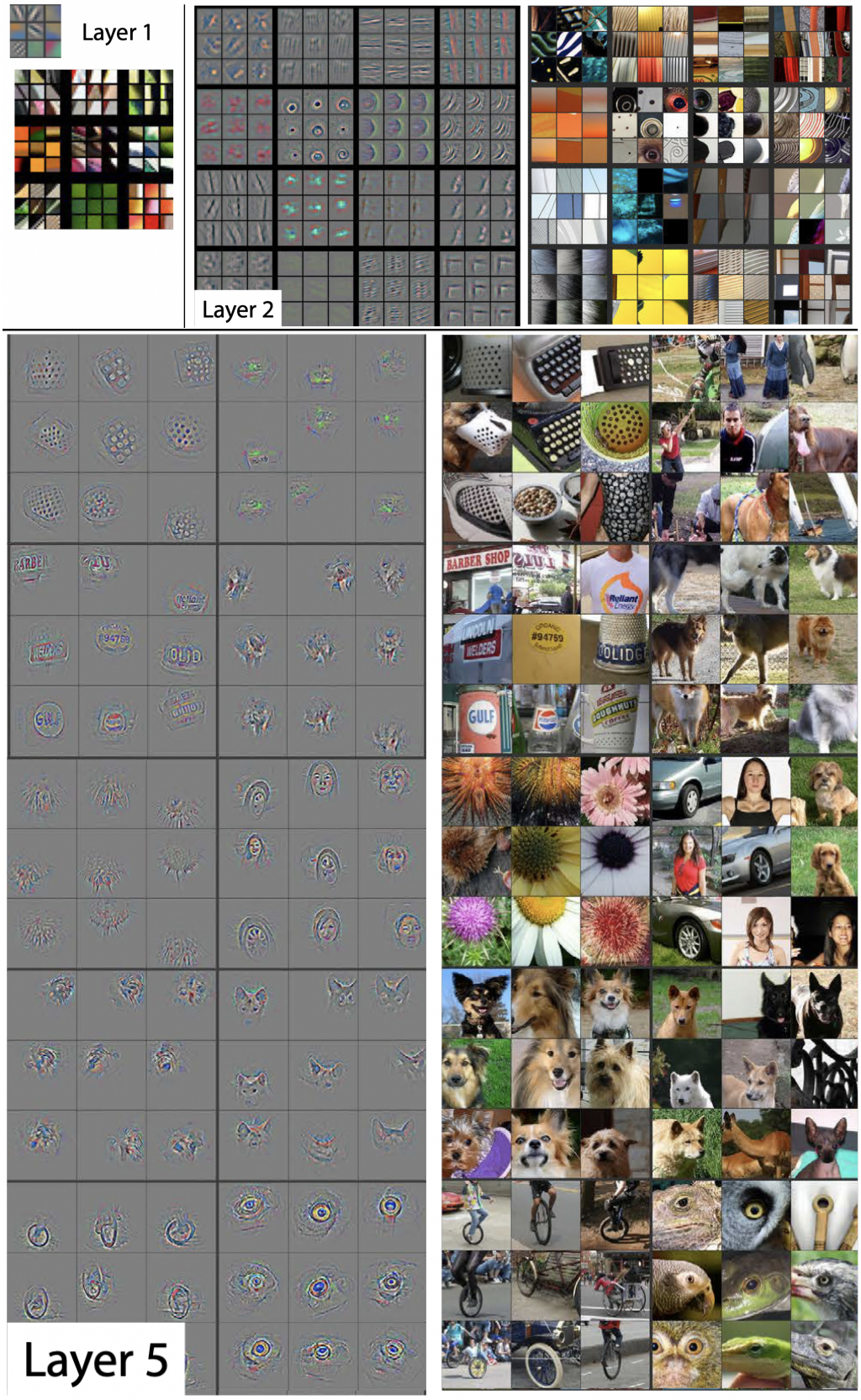


Figure 1.7: Visualization of features in a fully trained model, presented in a paper by Zeiler and Fergus [19].

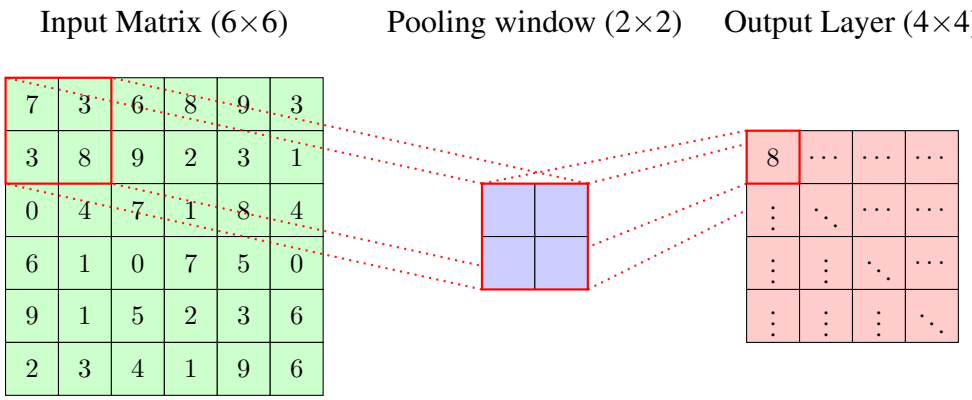


Figure 1.8: The principle of calculating the output of a pooling layer using the 2×2 max pooling window.

format suitable for classification.

Chapter 2

Machine learning algorithms for CDW classification: convolution versus extraction of selected features

Based on: V. Nežerka, T. Zbíral, J. Trejbal, Machine-learning-assisted classification of construction and demolition waste fragments using computer vision: convolution versus extraction of selected features, doi: 10.1016/j.eswa.2023.121568

2.1 Introduction

The construction industry has a significant socio-economic role as it generates around 25% of the global GDP and employs 7% of the population [20]. In the EU, 18 mil. people were employed in the construction sector in 2020 [21]. However, the sector is responsible for the enormous consumption of raw materials and large production of waste. Globally, it is estimated that the construction industry consumes over 30–40% of all natural resources extracted [22, 23], generates around 25–40% of the total solid waste [24], and emits up to 25% of anthropogenic CO_2 [25]. In 2020, the production of CDW in the EU was estimated to be around 747.3 mil. tons, which amounts to approximately 1685 kg per capita[1].

In order to pursue sustainable development, it is imperative to manage waste in a prudent and cost-effective manner and adopt the principles of circular economy [26, 27]. Following this direction, the European Parliament and the Commission issued Directive No 98/2008 which required the EU member states to increase the overall recycling of waste to at least 70% by weight from 2020. Even though the rate of CDW recycling in the EU is almost constant, at about 90% on average[2], the lion's share is downcycled. At the global scale, rapidly developing countries, such as China with 2 bn tons/year, are even bigger CDW producers than all the EU states combined [28].

The most commonly recycled CDW materials, besides soils, are concrete and ceramics, mostly used for embankments, backfills, fillings, or beddings under foundation slabs or pavings.

[1] https://ec.europa.eu/eurostat/databrowser/view/env_wasgen/default/bar
[2] https://ec.europa.eu/eurostat/databrowser/view/cei_wm040/default/table

Less frequently, the recycled fragments are used as aggregates in the production of new concrete mixes or the finest fractions as micro-fillers [29, 30, 31, 32]. The major limiting factor in the crushed CDW valorization in applications such as concrete manufacturing is improper sorting [33]. Yongbo Su [34] carried out a multi-agent evolutionary game study and concluded that research into CDW classification holds the greatest potential to promote CDW recycling and reuse. Davis et al. [35] pointed out that the automatic classification of CDW materials would significantly reduce the costs associated with sorting.

At the pre-sorting stage, methods exploiting gravitational, magnetic, inertial, electrostatic, or buoyancy forces are very efficient in separating specific types of materials from a heterogeneous CDW mix [36, 37]. Leveraging big data in CDW management offers promising advancements. Yuan et al. [38] utilized a dataset of 4.27 million truckloads of construction waste to estimate waste composition based on bulk density. Such techniques can significantly refine sorting processes and promote sustainable resource utilization.

Despite recent progress in advanced methods based on research into the development of various sensors (image, spectroscopic, spectral, UV sensitive, etc.) [36, 39], sorting of the remaining fragments is at the industrial scale most commonly accomplished manually and cannot be done properly due to their similarity. Therefore, it is desirable to replace manual sorting with robotic vision-based technologies such as RGB cameras, hyperspectral imaging, or X-ray sensors assisted with machine learning. This approach has been first employed for the purpose of municipal waste separation [39, 40, 41, 42] and the extensive development led to the sorting accuracy exceeding 90% [43].

The robotic vision-based technology has also started to find its way into the CDW sorting [44, 45]. However, automatic CDW recognition encountered its limitations in terms of accuracy and boundary identification. The latter issue was addressed by Dong et al.[46], who proposed a boundary-aware model with the ability to distinguish and segment individual materials within structural debris. CNNs are specialized for image recognition, leveraging their ability to identify hierarchical patterns in visual data. Their design enables them to dissect images into components, enhancing classification accuracy, especially in intricate tasks like CDW sorting. For instance, Xiao et al. [47] utilized CNNs to effectively classify different CDW materials, underscoring the potential of this approach in the domain. They classified different CDW materials (wood, brick, rubber, rock, concrete) with an accuracy exceeding 80%. Ku et al. [48] built a robotic line that automatically recognized and classified the basic materials within CDW using hyperspectral and 3D cameras with an accuracy of about 90%. Machine-learning classification was also employed by Lin et al. [49], who recognized visually different CDW fragments and achieved an accuracy ranging between 75 and 80%. The closest to our goal is the study by Hoong et al. [33], who employed neural networks for the classification of recycled aggregates. They constructed a library of 36,000 images of individual aggregate grains and their model achieved accuracies of up to 97%.

While previous studies have employed CNN-based models for CDW classification, our research distinguishes itself in two primary ways. Firstly, we focus on the efficient extraction of features describing the textures captured using ordinary RGB cameras, a method not extensively explored in prior work. Secondly, we provide a comprehensive comparison between CNN and other machine-learning models, specifically GB models and MLP, showcasing the efficacy of feature extraction in enhancing both speed and accuracy. This paper presents a unique approach to CDW fragment recognition, emphasizing the power of feature extraction. We pro-

vide extensive datasets, computer codes, and pre-trained models, ensuring our methodology is transparent, reproducible, and can be built upon by other researchers or industry stakeholders.

2.2 Methodology

The capabilities and limitations of the selected feature extraction methods and machine-learning models are demonstrated on four types of CDW fragments. These were chosen because they are the most common fragments found in mixed debris from demolition sites in the Czech Republic: light-colored aerated autoclaved concrete (AAC), asphalt conglomerates, ceramics (roof tiles and bricks), and concrete. These materials not only represent a significant portion of the total waste but also pose a challenge in terms of their similarity, making their accurate classification crucial for efficient recycling and waste management.

2.2.1 Collection of datasets

The 1920×1280 px images of ∼30–250mm fragments were taken from a distance of about 70 cm using a handheld digital single-lens reflex camera (Canon EOS 70D with a Canon zoom lens EF-S 17-85 IS USM) in a CDW collection and sorting yard near Kladno, Czech Republic (Figure 2.1). The images were captured in a shade to minimize variations in illumination and to ensure consistent image quality. Importantly, the CDW fragments were used in their natural state from the yard, without any presorting or cleaning, reflecting the real-world conditions of such waste. In a potential industrial deployment, techniques like air-flow cleaning could be introduced on conveyor belts to minimize dirt and dust, enhancing the image clarity. The fragments were placed on the ground while taking the images, or directly on the CDW piles.

Unlike clean structural elements, whose classification has been tackled in other studies [50, 51, 52, 53, 54, 55], recognition of CDW fragments is a more challenging task as their surface can be contaminated with dust and residues of other materials. Randomly selected samples of CDW fragments are presented in Figure 2.2, showing similar textures, especially in the case of AAC and concrete. The complete image datasets used for training of machine-learning classifiers and validation are open and provided as supplementary material [56].

The acquired image datasets were manually split to individual material classes. The annotated images within each class were divided into training and testing sets in a 4:1 ratio. Since the shape of fragments cannot be the key for classification and the classifiers were trained to recognize the CDW textures, 200×200 px regions (image subsets) were manually extracted for training and testing of the selected classifiers (Figure 2.3). The summary of these training/testing data is provided in Table 2.1.

2.2.2 Extraction of features

Images represent a high-dimensional input space with $D = N \times N \times C$ features, where $N \times N$ is the image subset size (px) and C is the number of color channels (equal to 3). Such large inputs can be tackled using CNNs, yet reducing the input space by extracting informative numeric features that describe the CDW texture (Figure 2.4) allows to use simple and efficient

Figure 2.1: The site for collecting images, a CDW collection and sorting yard near Kladno, Czech Republic.

Figure 2.2: Examples of image datasets for the examined CDW materials.

Table 2.1: Summary of extracted 200×200 px image subsets used for testing and training of selected classifiers.

Material (class)	Number of training images	Number of testing images
AAC	939	235
Asphalt	902	226
Ceramics	620	155
Concrete	825	206

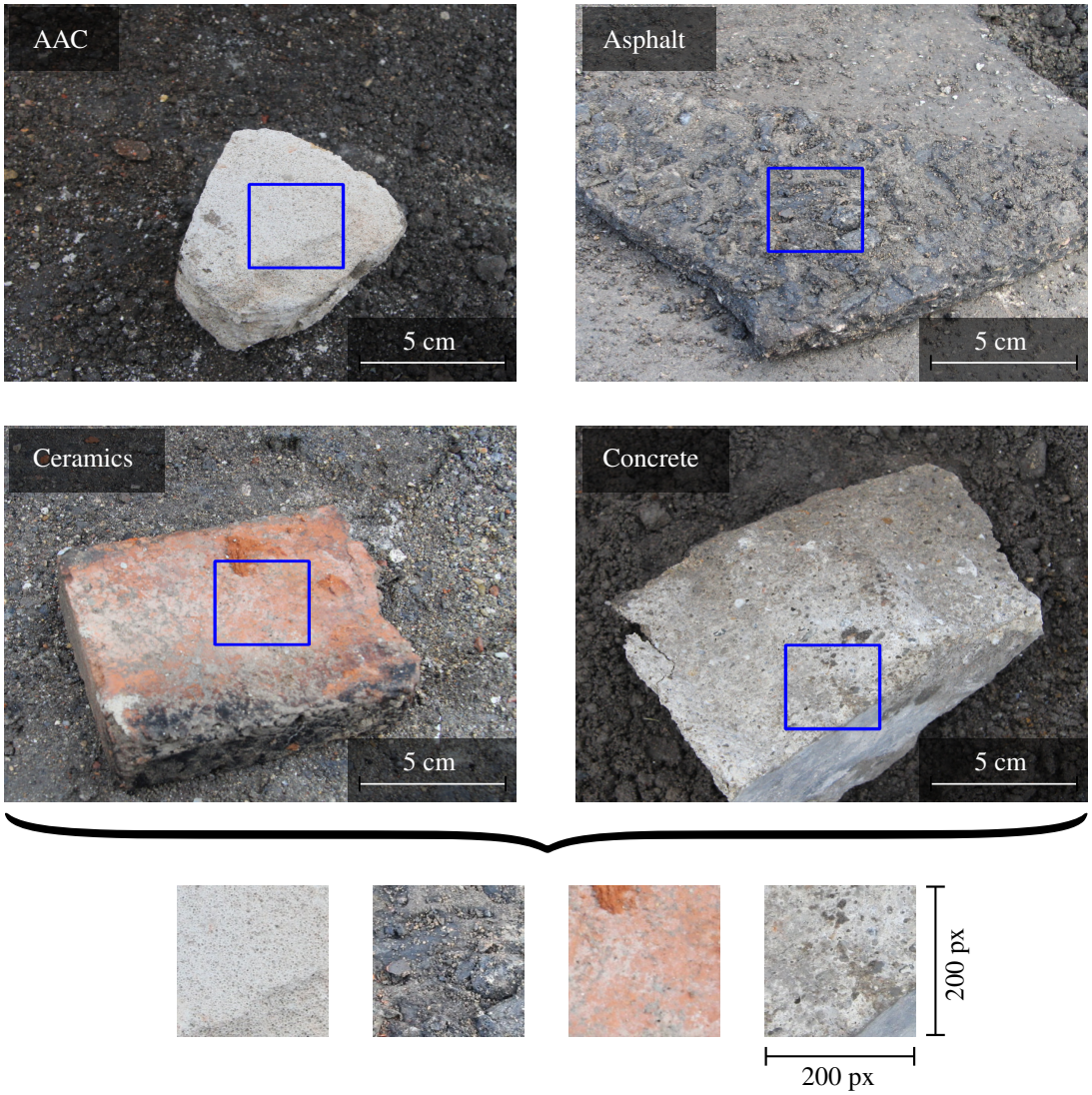


Figure 2.3: Manual extraction of 200×200 px regions (image subsets) used for training and testing of selected classifiers.

classification algorithms. In this study, we scrutinize the GB and MLP models for such a classification based on extracted features.

The following metrics are proposed to describe the color and texture of CDW fragments, reducing the input space to $D = 4$: (i) mean intensity, (ii) mean intensity of a selected color channel, (iii) Shannon entropy, and (iv) mean intensity gradient. To calculate these quantities, local coordinates (i, j) are introduced for image subsets (Figure 2.5). The 3-dimensional matrix of intensities for individual color channels, $I(C, i, j)$, was reduced to a single-channel matrix $I(1, i, j) \equiv I_{\text{gray}}(i, j)$, representing a gray-scale image, as

$$I_{\text{gray}}(i, j) = 0.299\, I_{\text{red}}(i, j) + 0.587\, I_{\text{green}}(i, j) + 0.114\, I_{\text{blue}}(i, j), \tag{2.1}$$

where $I_{\text{red}}(i, j)$, $I_{\text{green}}(i, j)$, and $I_{\text{blue}}(i, j)$ represent the matrices of intensities for the red, green, and blue channel, respectively. The weights for individual channels follow luma encoding that reflects different human vision sensitivity to particular colors.

Figure 2.4: Visualization of the image subset characteristics for individual materials (classes) as pairwise scatter plots; marginal distributions of each feature for each class are plotted on the diagonal.

2.2.2.1 Mean intensity

Mean intensity, $\overline{I_{\text{gray}}}$, is strongly influenced by the illumination of a captured scene and cannot be considered a reliable feature if constant illumination is not ensured for all (training, testing, and classified) images. Since this proof-of-the-concept study is intended as a cookbook for CDW fragments recognition on conveyor belts in an indoor environment, $\overline{I_{\text{gray}}}$ can be considered as one of the relevant features for classification and is calculated as

$$\overline{I_{\text{gray}}} = \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{I_{\text{gray}}(i,j)}{N^2}. \tag{2.2}$$

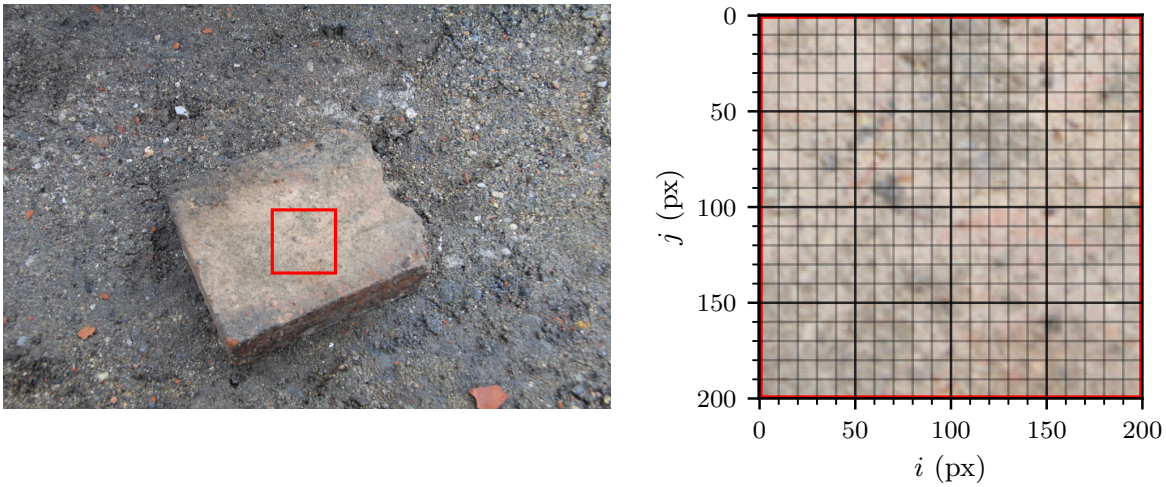


Figure 2.5: Local coordinates (i, j) for a subset of pixels (right) arbitrarily located within an image of a CDW fragment (left).

2.2.2.2 Mean intensity of red color

The color distribution is one of the key features and many machine-learning models for material recognition were based purely on color-based classification [51]. It was found during a preliminary analysis that for CDW materials, it is sufficient to focus on the predominance of a specific color. Given the orange/reddish color of ceramic fragments, the mean intensity of the red channel, $\overline{I_{\text{red}}}$, relative to the mean intensity (brightness) was selected as the most appropriate color-related label and its value was calculated as

$$\overline{I_{\text{red}}} = \sum_{i=1}^{N}\sum_{j=1}^{N} \frac{I_{\text{red}}(i,j)}{N^2} \frac{1}{\overline{I_{\text{gray}}}}. \tag{2.3}$$

2.2.3 Shannon's entropy

Many distinct CDW materials have similar colors and color-based labeling may fail [52, 57, 58]. To evaluate the randomness of a texture pattern as an additional feature, Shannon's entropy appears to be the most easy-to-calculate measure [59, 60, 61]. It was first proposed by Claude Shannon in 1948 to evaluate the average level of uncertainty in a signal as [62, 63]

$$H = -\sum_{I_{\text{gray}}=0}^{255} P(I_{\text{gray}}) \log_2 P(I_{\text{gray}}), \tag{2.4}$$

where $P(I_{\text{gray}}) \in [0, 255]$ (8-bit images) is the frequency of gray pixels' intensity. High values of H indicate higher uncertainty (randomness) of the signal (image).

2.2.3.1 Mean intensity gradient

Mean intensity gradient ($\overline{\nabla_I}$) was proposed by Pan et al. [64] as an indicator of stochastic pattern quality in regard to digital image correlation measurements. It evaluates the frequency and intensity of irregularities within an image. Such a measure is directly related to the texture roughness, being another crucial feature used for material classification [65]. In this study, the mean intensity gradient was calculated as

$$\overline{\nabla_I} = \sum_{i=1}^{N}\sum_{j=1}^{N}|\nabla I_{\text{gray}}(i,j)|\frac{1}{N^2}, \tag{2.5}$$

where $|\nabla I_{\text{gray}}(i,j)| = \sqrt{I_i(i,j)^2 + I_j(i,j)^2}$ is the modulus of local intensity gradient and I_i and I_j are the i-directional and j-directional derivatives of $I_{\text{gray}}(i,j)$ at each pixel location (i,j). The differentiation was accomplished using a Sobel operator with a 3×3 kernel [66].

2.2.4 Classifiers

The machine-learning models used for classification are only briefly introduced in the following sections, along with a presentation of input parameters for each model. Detailed descriptions and analyses of the models are beyond the scope of this paper. The curious reader is referred to comprehensive books on machine learning such as ones by Geron [67] and Murphy [68].

The choice of classifiers in this study was driven by the aim to span a spectrum of algorithmic complexity and to capture the strengths of different types of models. Specifically:

1. GB is renowned for its efficiency in classification tasks. It excels in handling structured data and can seamlessly navigate the non-linear relationships between features, making it a robust choice for our dataset [69].

2. MLP, as a basic form of artificial neural networks (ANNs), bridges the gap between traditional machine learning and deep learning techniques [70]. Its inclusion allowed us to gauge the efficiency of a simpler neural network architecture in the context of CDW recognition.

3. CNN was incorporated as a benchmark due to its inherent layered analysis capabilities. By automatically extracting features, CNNs detect edges and intricate patterns. Its performance provides insights into how deep learning techniques interpret the visual features in the CDW fragments.

The performance of individual classifiers was tested on a custom-built desktop computer equipped with an Intel 4 core i3-8350K CPU, 16 GB RAM, 250 GB SSD hard drive, Windows 10 operating system, and Python 3.10.9. The Python codes and pre-trained models are provided along with this paper [71].

2.2.4.1 Gradient boosting

GB is a machine learning algorithm that typically uses decision trees [72] as its base models [73]. The decision tree is a flowchart-like tree structure where each internal node tests an

attribute, and the connected branches represent an outcome of the test. Analogically to leaves, the terminal nodes hold class labels [74].

At each iteration, GB trains a weak decision tree model on the residual errors between the true and predicted labels of the previous iteration. The final prediction is made by adding up the predictions of all the decision trees, where the contribution of each tree depends on its weight, determined by the improvement in the loss function after adding the tree to the ensemble. The loss function is minimized using gradient descent. The algorithm usually outperforms random forest classifiers in terms of speed and accuracy of the predictions [75, 76].

The GB classifier used in this study was implemented in the Scikit-Learn v.1.1.3 Python package. Standardization of features was performed using the *preprocessing.StandardScaler* class. Cross-validation was accomplished using the *model_selection.StratifiedShuffleSplit* class that provides randomly selected indices to split datasets into test/train data and preserves the percentage of samples for each class. The input parameters for the *ensemble.GradientBoostingClassifier* model class were defined as summarized in Table 2.2. The optimum parameters were selected based on the prediction accuracy and speed. The optimization was done using the Scikit-learn's *model_selection.GridSearchCV* class that provides an exhaustive search over specified values of model parameters (learning rate $\in$ [0.2, 0.8], maximum depth $\in$ [3, 5], and a number of estimators $\in$ [100, 200]). Other input parameters were kept in their default settings.

Table 2.2: Summary of input parameters for the GB classifier implemented in Scikit-Learn v.1.1.3 (*ensemble.GradientBoostingClassifier* model class).

Input parameter	Keyword argument	Value	Note
Random state	random_state	0	Fixing the random state ensures deterministic behavior during fitting
Learning rate	learning_rate	0.4	Learning rate shrinks the contribution of each tree
Maximum depth	max_depth	4	Maximum depth of individual regression estimators, limiting the number of nodes in decision trees
Number of estimators	n_estimators	125	Number of boosting stages to perform

2.2.4.2 Multi-layer perception

MLP is a type of artificial neural network that consists of an input layer, a specified number of hidden layers, and an output layer [77, 78]. The input layer represents the features of the input data, while the output layer represents the predicted probability for all classes. The hidden layers are used to learn the non-linear transformations of the input features that lead to the final prediction. In our implementation, the MLP model consists of a single hidden layer; this hidden layer consists of neurons, where each neuron applies a weighted sum of the input features and a bias term, followed by an activation function, such as sigmoid or hyperbolic tangent (tanh). The weights and biases are learned through backpropagation, where the gradients of the loss function are computed to update the weights and biases using gradient descent.

Also, the MLP classifier was implemented in the Scikit-Learn v.1.1.3 Python package. The training procedure was similar to that of the GB model: the standardization of features was performed using the *preprocessing.StandardScaler* class and *model_selection.StratifiedShuffleSplit* class was used for cross-validation. The input parameters for the *neural_network.MLPClassifier* model class were defined as summarized in Table 2.3. The search for optimum parameters

was also accomplished using the Scikit-learn's *model_selection.GridSearchCV* class, searching over specified values of model parameters (learning rate ∈ {adaptive, constant ∈ [0.005, 0.015, 0.05]}, solver ∈ {stochastic gradient descent, stochastic gradient-based optimizer (adam) [79]}, activation ∈ {rectified linear unit function (ReLU), hyperbolic tan function (tanh)}, and a hidden layer size ∈ [5, 100]). Other input parameters were kept in their default settings.

Table 2.3: Summary of input parameters for the MLP classifier implemented in Scikit-Learn v.1.1.3 (*neural_network.MLPClassifier* model class).

Input parameter	Keyword argument	Value	Note
Random state	random_state	0	Fixing the random state ensures deterministic behavior during fitting
Learning rate	learning_rate_init	0.015	Controls the step-size in updating neuron weights
Maximum number of iterations	max_iter	800	Number of epochs (how many times each data point is used)
Learning rate schedule	learning_rate	'constant'	Selected constant learning rate
Solver	solver	'adam'	Weight optimization using the Adam algorithm [79]
Neuron activation function	activation	'tanh'	Activation function for the hidden layer
Hidden layer size	hidden_layer_sizes	(20,)	Single hidden layer with 20 neurons

2.2.4.3 Convolutional neural network

CNN is a type of artificial neural network that is designed for the analysis of data with a grid-like topology (e.g., images) [80, 81, 82]. It consists of several layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layer applies a convolution operation to the input image, where the convolution kernel slides over the image and computes the dot product between the kernel and the local patch of the image to extract features. The convolutional layer is followed by an activation function that applies non-linear transformations to the output of the convolution. The pooling layer reduces the spatial dimensions of the output of the convolutional layer by applying a pooling operation, such as max pooling, that takes the maximum value of a local patch. The fully connected layer combines the features learned by the convolutional and pooling layers and makes the final prediction. The weights and biases of the convolutional and the fully connected layers are adjusted during the network training through backpropagation, exploiting the gradient descent algorithm.

Unlike GB and MLP classifiers, CNN takes the whole image as input. Since the model in our study was trained on 200×200 px 3-channel (RGB) images, images for classification having a different size were rescaled to 200×200 px using an interpolation function. The CNN classifier was implemented in the Tensorflow Keras v.2.10.0 Python package, provided by the *models.Sequential* class.

Different architectures of CNNs with various number of filters for the convolutional layers have been tested. The selected model includes three convolutional layers, each followed by a max pooling layer, a flatten layer, and two dense layers. The first and third convolutional layers have 32 3×3 filters, a stride of 1, and a ReLU activation function. The second convolutional layer has 64 3×3 filters and the same activation function. The max pooling layers downsample

the feature maps by a factor of two to make the model more efficient. The flatten layer converts the 2D feature maps into a 1D vector. The two dense layers consist of 256 units with a ReLU activation function, followed by an output layer with four neurons corresponding to the individual CDW classes.

The selected model architecture is described in detail in Table 2.4. During the training process, the model achieved 100% accuracy on the training data (α_{train}) after 30 epochs, but the maximum accuracy on the testing data ($\alpha_{\text{test}} = 80\%$) was reached after 11 epochs, suggesting potential overfitting (Figure 2.6). The model trained after 11 epochs was adopted for the future CDW classification.

Table 2.4: Architecture of the CNN models; the individual layers were implemented in the Tensorflow Keras v.2.10.0 Python package, the *layers* class.

Layer	Keras class	Purpose
Convolutional layer (32 filters, size 3×3)	*Conv2D(32, (3, 3), 1, activation='relu', input_shape=(200, 200, 3))*	Extract features from the input images
Maximum pooling layer (2×2 pool)	*MaxPooling2D()*	Downsample the feature maps from the previous layer
Convolutional layer (64 filters, size 3×3)	*Conv2D(64, (3, 3), 1, activation='relu')*	Extract features from the previous layer
Maximum pooling layer (2×2 pool)	*MaxPooling2D()*	Downsample the feature maps from the previous layer
Convolutional layer (32 filters, size 3×3)	*Conv2D(32, (3, 3), 1, activation='relu')*	Extract features from the previous layer
Flattening layer	*Flatten()*	Flattens the 2D feature map into a 1D array
Fully connected layer (256 neurons)	*Dense(256, activation='relu')*	Take the flattened vector from the previous layer as input
Output layer (4 neurons)	*Dense(4)*	Values of individual neurons represent probabilities that the input belongs to each of the possible classes

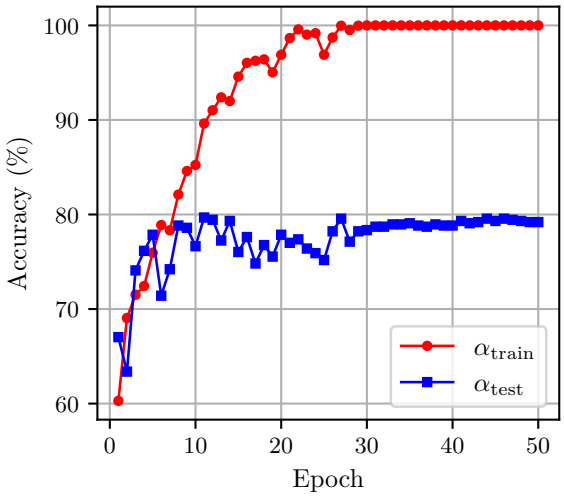


Figure 2.6: Training and testing accuracy as a function of epoch recorded during CNN training.

2.2.4.4 Model Evaluation Metrics

To evaluate the performance of our multi-class classification models, we primarily utilize accuracy and the weighted F-score.

Let P_c^{true} be the number of true positives for class c, N_c^{true} the true negatives, P_c^{false} the false positives, and N_c^{false} the false negatives. Accuracy, denoted by α, measures the proportion of all correct predictions across all four classes:

$$\alpha = \frac{\sum_{c=1}^{4} P_c^{\text{true}} + N_c^{\text{true}}}{\sum_{c=1}^{4} P_c^{\text{true}} + N_c^{\text{true}} + P_c^{\text{false}} + N_c^{\text{false}}}. \tag{2.6}$$

The precision P_c and recall R_c for each class are respectively defined as:

$$P_c = \frac{P_c^{\text{true}}}{P_c^{\text{true}} + P_c^{\text{false}}} \quad \text{and} \quad R_c = \frac{P_c^{\text{true}}}{P_c^{\text{true}} + N_c^{\text{false}}} \tag{2.7}$$

The F-score for class c, denoted as F_c, offers a balance between P_c and R_c. It is described as the harmonic mean of P_c and R_c:

$$F_c = \frac{2P_cR_c}{P_c + R_c} \tag{2.8}$$

For our multi-class problem, the weighted F-score, F_{weighted}, is calculated by averaging the F-score of each class, weighted by the proportion of samples from that class:

$$F_{\text{weighted}} = \sum_{c=1}^{4} w_c F_c \tag{2.9}$$

where w_c denotes the weight (proportion of samples) for the c^{th} class.

We employ both α and F_{weighted} in this study to evaluate the performance of our classifiers, providing a comprehensive view of their efficacy, especially in light of the minor class imbalance present in our dataset.

2.3 Results and discussion

The performance of individual classifiers is represented through confusion matrices (Figure 2.7), alongside the results of "manual" classification. This manual classification was accomplished using an online survey[1] by five experts on building materials from the Faculty of Civil Engineering, Czech Technical University in Prague.

While accuracy provides a general measure of correctness, the weighted F-score offers a more balanced measure between precision (how many selected items are relevant) and recall (how many relevant items are selected). For instance, GB and MLP classifiers achieved an accuracy of 82.5% with F-scores of 82.4%, indicating a harmonious balance between precision

[1] https://rm.fsv.cvut.cz/cdw/

and recall. The CNN classifier achieved an accuracy of 82.1% and an F-score of 82.3%, further demonstrating the model's consistent performance. In comparison, human experts achieved an accuracy of 87.2% and an F-score of 87.5%, outperforming the machine classifiers slightly.

Both machine-learning classifiers and human experts had difficulties distinguishing between image samples of AAC, asphalt, and concrete. This demonstrates the inherent difficulty in differentiating these materials visually, particularly when they share similar characteristics like a grayish color and texture. In contrast, ceramics (bricks, roof tiles, etc.) were recognized with an impressive accuracy of over 96% by both groups. A potential enhancement to the classification process could be the integration of a basic weight measurement device. Given the significant differences in density between the grayish materials, weight can be a distinguishing factor. Moreover, if a dual-camera setup were employed, the segmentation technique would permit volume estimation from visual data, further refining the differentiation process.

Despite the commendable performance of human experts, there are inherent limitations to relying on manual sorting. Prolonged concentration can lead to lapses in attention, impacting the consistency of the sorting process [83]. Furthermore, machine classifiers, especially when deployed on standard office computers, can process samples at a rate that outpaces human capability by orders of magnitude.

In recent literature, Davis et al. [35] reported accuracy levels between 80% and 97% for the CNN-based classification of general waste. Their categories included paper, glass, plastic, metal, cardboard, and non-recyclables. Although their work achieved an accuracy of up to 95.7% for CDW, it's crucial to note that the objects they classified had more distinct shapes than the CDW fragments. Xian et al. [84] reported a perfect accuracy of 100% in their classification of CDW on a conveyor belt. They employed a high-cost near-infrared hyperspectral camera and a dataset with distinct categories like foam, plastic, brick, concrete, and wood. Introducing more challenging materials such as asphalt conglomerates or AAC, often found in CDW, could potentially reduce this high accuracy even with advanced hardware.

A study on the performance of the individual classifiers in terms of speed and accuracy is presented as a function of subset size in Figure 2.8. As larger subsets contained more information, the accuracy of models increased. This phenomenon was most significant in the case of CNN, for which the image subsets had to be rescaled to 200×200 px to have the same size as images used for training. Similar findings were reported by Dimitrov and Golparvar-Fard. [52], who developed a system for vision-based material recognition and monitoring of construction progress, employing the SVM classifier [85].

In our study, the GB and MLP models that utilized feature extraction, exhibited similar speed and accuracy, both superior to CNN, especially for small subsets. Unlike CNN, both models approached their maximum accuracies at approximately 150×150 px subset size. The classification speed of GB and MLP classifiers, including feature extraction, was about 15× higher compared to CNN.

The practical demonstration of the image subset classification is provided in Figure 2.9. Here, the randomly selected CDW fragments from the testing dataset were localized using the Rembg[1] Python package based on the U^2-Net deep neural network [86]. An auxiliary script was designed to extract image subsets from the unmasked regions. The accuracy of the CNN

[1] https://github.com/danielgatis/rembg

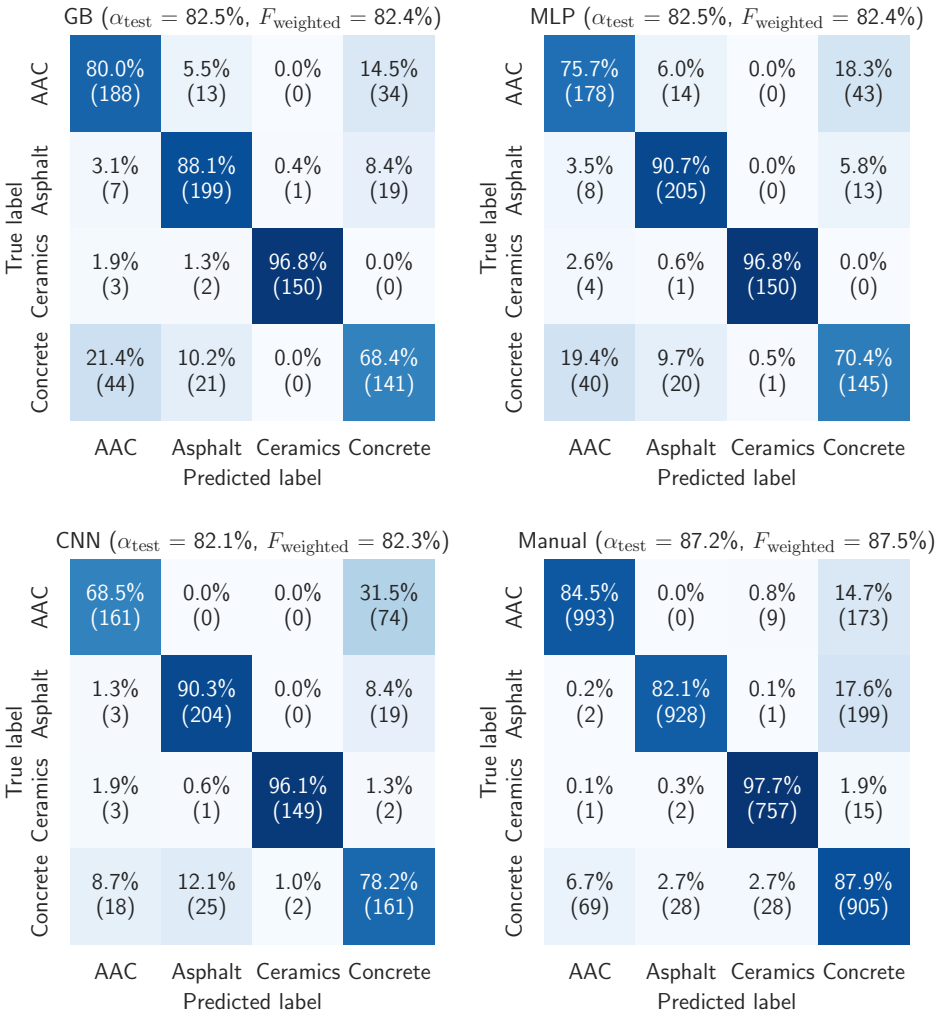


Figure 2.7: Confusion matrices for different classifiers and comparison of their performance with manual classification done by five experts on building materials from the FCE CTU in Prague.

classifier was compromised by the small size (135×135 px) subsets placed over the region of interest; however, even despite this shortcoming, even the CNN classified the fragments correctly with high confidence. Nearly 100% confidence was reached by the GB and MLP classifiers.

This demonstration shows that the accuracy reached for individual subsets is improved by placing a higher number of these over the samples. The accuracy was tested on a comprehensive dataset containing 2664 images of CDW fragments [56]; the summary of reached accuracies for individual classifiers is provided in Table 2.5. Classification of several samples per a CDW fragment led to overall accuracy ranging between 85.9% (CNN) and 92.3% (GB), reaching the accuracy reported by other authors dealing with the classification of clean building materials. In a study by Mahami et al. [55], the authors managed to classify eleven construction materials using CNN (VGG16 network [87]) and reached up to 97.35% accuracy, yet, their dataset did not contain contaminated materials having similar textures, such as fragments of AAC and concrete in our study.

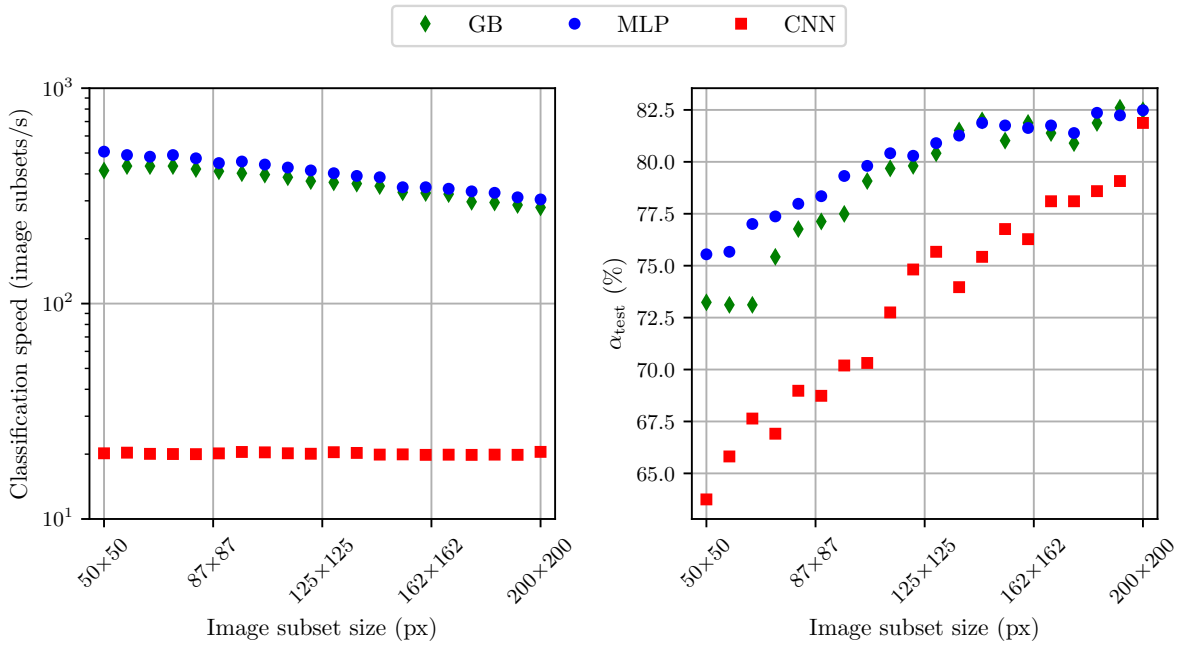


Figure 2.8: Speed (left) and accuracy (right) reached by individual classifiers on the validation (testing) datasets for different sizes of image subsets that were extracted by cropping the redundant portion of the images.

Table 2.5: Accuracy of different classifiers when recognizing whole CDW fragments by classifying several (>4) 200×200 image subsets with a 70 px overlap (Figure 2.10).

Classifier	AAC (582 images)	Asphalt (741 images)	Ceramics (572 images)	Concrete (769 images)	Complete dataset (2664 images)
GB	86.9%	93.9%	99.1%	89.7%	92.3%
MLP	89.4%	93.8%	98.4%	85.2%	91.3%
CNN	56.7%	97.2%	99.0%	87.5%	85.9%

Our models, especially the Gradient Boosting and Multi-Layer Perceptron classifiers, demonstrated competitive performance when compared to previous studies, as summarized in Table 2.6. Notably, while our dataset size was comprehensive, the nature of our CDW images, which included contaminated materials with similar textures, made the classification task more challenging.

It should be noted that all the images for both training and testing datasets were taken using the same camera and similar conditions, which can compromise the robustness of the classification models. The goal of this proof-of-the-concept study is to demonstrate the capabilities of the proposed low-cost lightweight procedures that could be implemented in CDW sorting and recycling plants for CDW recognition on conveyor belts. For particular industrial applications, new site-specific training datasets should be acquired, optimally involving auxiliary data (weight, acoustic emissions, etc.) from other sensors. Fusion of RGB cameras with different sensors could significantly increase the accuracy, especially in the case of lightweight AAC which is often confused with fragments of concrete that also have a fine texture and grayish color.

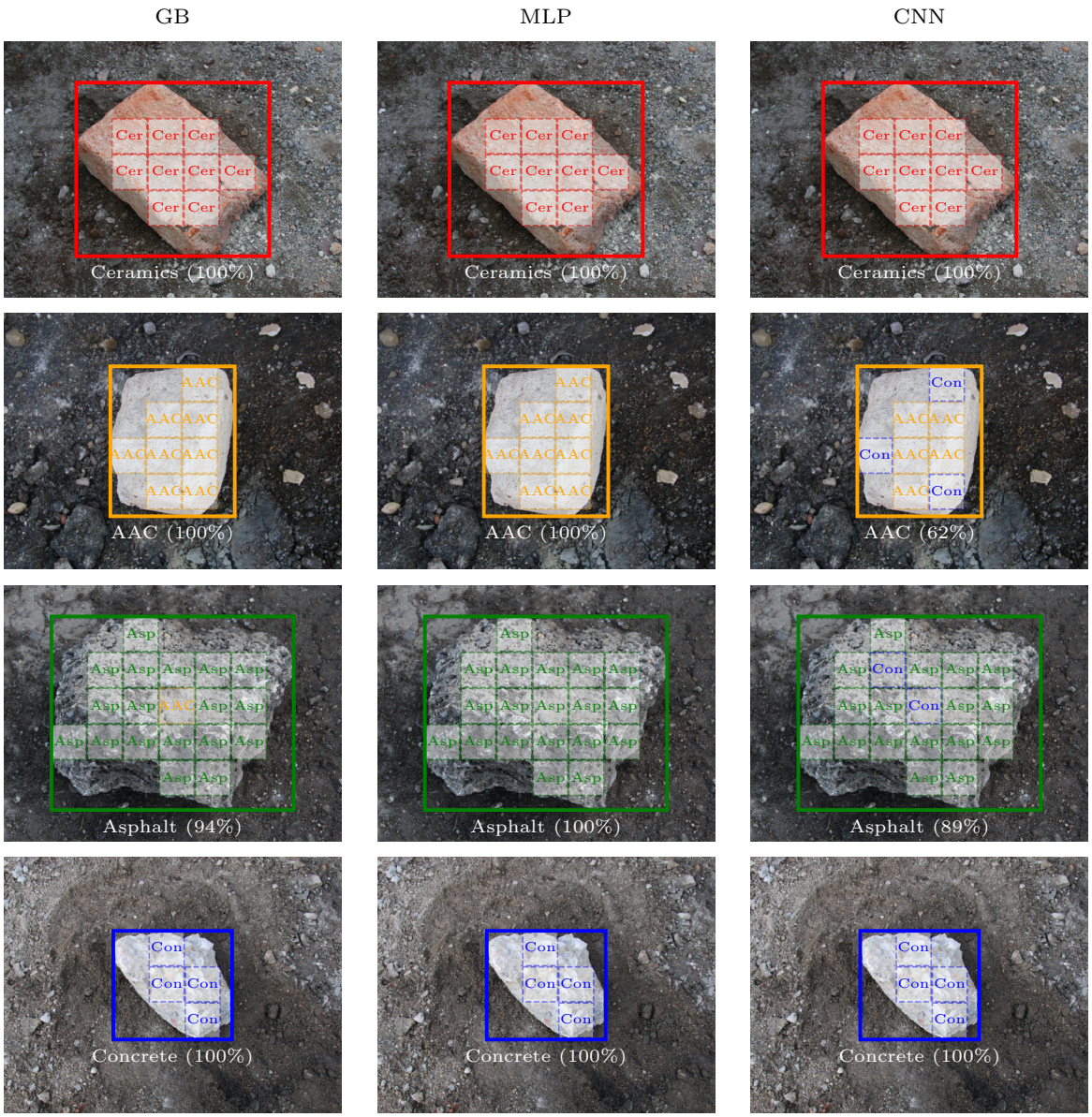


Figure 2.9: Localization of whole CDW fragments and their classification based on texture recognition using different classifiers; the size of image subsets 135×135 px.

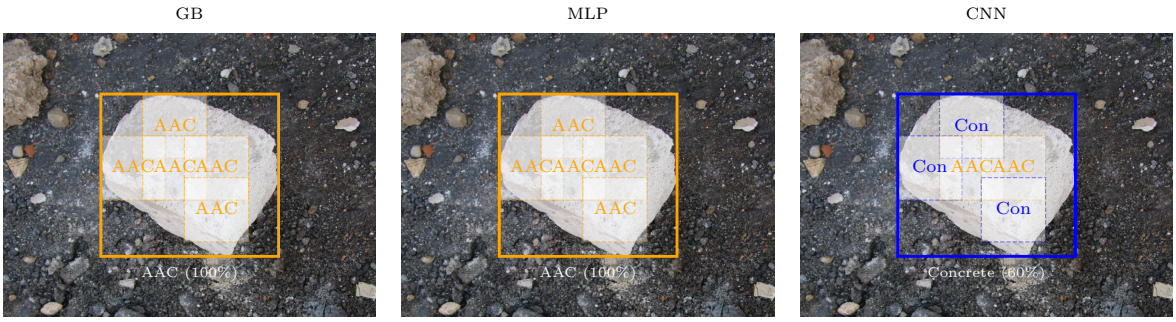


Figure 2.10: A typical misclassification of AAC fragments by CNN during a comprehensive validation of the classification algorithms; size of image subsets 200×200 px with a 70 px overlap.

Table 2.6: Comparison of the current study with previous significant works focused on machine-learning-based recognition of construction materials in terms of model performance, data type, and dataset size.

Reference	Model	Accuracy	Dataset type and size	Dataset size
This study (GB)	GB	92.3%	CDW images	2664
This study (MLP)	MLP	91.3%	CDW images	2664
This study (CNN)	CNN	85.9%	CDW images	2664
[35]	CNN	80-97%	Images of conatiners with bulk CDW	2283
[84]	CNN	100%	Hyperspectral images of very diverse materials	250
[52]	SVM	Up to 97.1%	Point cloud patches (images of construction surfaces)	3740
[55]	CNN (VGG16)	97.35%	Images of clean very diverse materials	1231
[38]	BD-P model	90.2%	Bulk density (truck loads)	4.27 mil.
[33]	CNN (Custom ResNet34)	97%	Images of recycled aggregates	36000
[49]	CNN (CVGGNet-16)	76.6%	Images of diverse clean bulk materials	2836 (bofore augmentation)

2.3.1 Application procedure

Our developed machine-learning-assisted method for CDW fragment recognition is designed for easy integration into existing CDW sorting systems. Here, we outline the potential application procedure:

1. **Image Acquisition**: Using high-resolution cameras, images of CDW fragments on conveyor belts or sorting platforms are captured. Ideally, this would be integrated into a continuous flow system where CDW moves along a conveyor.

2. **Preprocessing**: The captured images undergo preprocessing, which may include cleaning using air-flow or other mechanisms to enhance clarity, and then they are fed into the model.

3. **Density Estimation**: For individual fragments on the conveyor belt, a weight measurement system can be integrated to estimate the density of each fragment. This can assist in further refining the classification, especially for fragments with similar appearances but different densities (e.g., AAC and concrete).

4. **Classification**: The preprocessed images are classified in real-time using a trained model. The model identifies the type of CDW fragment and can potentially direct its sorting into appropriate bins or sections.

5. **Post-processing**: Based on classifications, automated mechanisms or manual laborers can be directed to ensure correct sorting or further refinement.

6. **Feedback Loop**: The system can be designed to continuously learn from any misclassifications through a feedback mechanism, enhancing accuracy over time.

This proposed application procedure is modular and can be customized based on the specific requirements of the CDW sorting facility, available resources, and desired accuracy levels.

2.4 Conclusion

Proper sorting of construction and demolition waste (CDW) fragments is essential for its further valorization. In this study, we demonstrated the potential of machine-learning models for the recognition and classification of CDW fragments using computer vision-based algorithms. The approach was tested on four types of CDW material fragments commonly found in mixed debris from demolition sites: aerated autoclaved concrete (AAC), asphalt conglomerates, ceramics (roof tiles and bricks), and concrete fragments. For that purpose, we examined three machine-learning classification models, gradient boosting (GB), multi-layer perception (MLP), and convolutional neural network (CNN).

In contrast to CNN, having the 200×200×3 px RGB images as its input, GB and MLP were trained on classifying the CDW texture based on four extracted features: (i) mean intensity, (ii) mean intensity of the red color channel, (iii) Shannon entropy, and (iv) mean intensity gradient, reducing the input space from $D = 120,000$ to $D = 4$. In the case of CNN, the feature extraction was accomplished using convolutional layers. The GB and MLP classifiers outperformed CNN not only in terms of speed (for a single image subset ∼300 s^{-1} vs. ∼20 s^{-1}), but also accuracy, especially when classifying images of sizes below 200×200 px, on which the models were trained.

Despite the high similarity of the recognized textures and contamination of the CDW fragments with dust, the examined classifiers exhibited accuracy over 82.1% for 200×200 px image subsets, slightly below the average accuracy reached by experts on building materials (87.2%). The accuracy reached up to 92.3% (GB) when classifying the whole fragments by placing several subsets over the samples. The lowest overall accuracy was reached when using CNN because the model often misclassified AAC for concrete. All the models were most accurate when classifying fragments of ceramics (98.4–99.1%) because of their distinct reddish color.

However, this study comes with certain limitations. All images, both for training and testing datasets, were acquired under similar conditions using the same camera, which might affect the

robustness of the classifiers in more varied settings. Moreover, while the study showcases the capabilities of low-cost procedures for CDW recognition, it underscores the need for acquiring new site-specific training datasets for specific industrial applications; optimally on a conveyor belt. The integration of additional sensors or data sources could further enhance accuracy and reliability.

The links to image datasets, computer codes, and pre-trained models used in this study are open and are provided as supplementary material. We believe that the findings can promote the developments in robotics-assisted sorting of CDW fragments, enabling its efficient use in the production of new materials and products and reduction of the environmental burden associated with CDW disposal.

References

[1] AI Index Steering Committee. AI Index Report 2023. Technical report, Stanford Institute for Human-Centered Artificial Intelligence (HAI), 2023.

[2] Roberto Pierdicca and Marina Paolanti. Geoai: A review of artificial intelligence approaches for the interpretation of complex geomatics data. *Geoscientific Instrumentation, Methods and Data Systems*, 11(1):195–218, Jun 2022.

[3] Shan Liu, Kenan Li, Xuan Liu, and Zhengtong Yin. Geospatial ai in earth observation, remote sensing, and giscience, 2023.

[4] Anushka G. Satav, Sunidhi Kubade, Chinmay Amrutkar, Gaurav Arya, and Ashish Pawar. A state-of-the-art review on robotics in waste sorting: scope and challenges. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, 17(6):2789–2806, 2023.

[5] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[6] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[7] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[9] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[11] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[12] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[13] Magdi Zakaria, AS Mabrouka, and Shahenda Sarhan. Artificial neural network: a brief overview. *neural networks*, 1:2, 2014.

[14] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[15] Gottfried Wilhelm Freiherr von Leibniz. *The early mathematical manuscripts of Leibniz: Translated from the Latin texts published by Carl Immanuel Gerhardt with critical and historical notes*. 1920.

[16] Hyeon-Joong Yoo. Deep convolution neural networks in computer vision: a review. *IEIE Transactions on Smart Processing and Computing*, 4(1):35–43, 2015.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[18] Steven B Damelin and Willard Miller. *The mathematics of signal processing*. Number 48. Cambridge University Press, 2012.

[19] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[20] Masoud Norouzi, Marta Chàfer, Luisa F. Cabeza, Laureano Jiménez, and Dieter Boer. Circular economy in the building and construction sector: A scientific evolution analysis. *Journal of Building Engineering*, 44:102704, 2021.

[21] Gabriel Luiz Fritz Benachio, Maria do Carmo Duarte Freitas, and Sergio Fernando Tavares. Circular economy in the construction industry: A systematic literature review. *Journal of Cleaner Production*, 260:121046, 2020.

[22] Amos Darko, Albert P.C. Chan, Michael A. Adabre, David J. Edwards, M. Reza Hosseini, and Ernest E. Ameyaw. Artificial intelligence in the AEC industry: Scientometric analysis and visualization of research activities. *Automation in Construction*, 112:103081, 2020.

[23] Callun Keith Purchase, Dhafer Manna Al Zulayq, Bio Talakatoa O'Brien, Matthew Joseph Kowalewski, Aydin Berenjian, Amir Hossein Tarighaleslami, and Mostafa Seifan. Circular economy of construction and demolition waste: A literature review on lessons, challenges, and benefits. *Materials*, 15:76, 2021.

[24] Mohammed Haneef Abdul Nasir, Andrea Genovese, Adolf A. Acquaye, S.C.L. Koh, and Fred Yamoah. Comparing linear and circular supply chains: A case study from the construction industry. *International Journal of Production Economics*, 183:443–457, 2017.

[25] Amirreza Mahpour. Prioritizing barriers to adopt circular economy in construction and demolition waste management. *Resources, Conservation and Recycling*, 134:216–227, 2018.

[26] Tuomo Joensuu, Harry Edelman, and Arto Saari. Circular economy practices in the built environment. *Journal of Cleaner Production*, 276:124215, 2020.

[27] Benjamin I. Oluleye, Daniel W.M. Chan, Abdullahi B. Saka, and Timothy O. Olawumi. Circular economy research on building construction and demolition waste: A review of current trends and future research directions. *Journal of Cleaner Production*, 357:131927, 2022.

[28] Lina Zheng, Huanyu Wu, Hui Zhang, Huabo Duan, Jiayuan Wang, Weiping Jiang, Biqin Dong, Gang Liu, Jian Zuo, and Qingbin Song. Characterizing the generation and flows of construction and demolition waste in China. *Construction and Building Materials*, 136:405–413, 2017.

[29] R. Hlůžek, J. Trejbal, V. Nežerka, P. Demo, Z. Prošek, and P. Tesárek. Improvement of bonding between synthetic fibers and a cementitious matrix using recycled concrete powder and plasma treatment: from a single fiber to FRC. *European Journal of Environmental and Civil Engineering*, 26:3880–3897, 2020.

[30] Z. Prošek, J. Trejbal, V. Nežerka, V. Goliáš, M. Faltus, and P. Tesárek. Recovery of residual anhydrous clinker in finely ground recycled concrete. *Resources, Conservation and Recycling*, 155:104640, 2020.

[31] J. Valentin, J. Trejbal, V. Nežerka, T. Valentová, and M. Faltus. Characterization of quarry dusts and industrial by-products as potential substitutes for traditional fillers and their impact on water susceptibility of asphalt concrete. *Construction and Building Materials*, 301:124294, 2021.

[32] V. Nežerka, Z. Prošek, J. Trejbal, J. Pešta, J.A. Ferriz-Papi, and P. Tesárek. Recycling of fines from waste concrete: Development of lightweight masonry blocks and assessment of their environmental benefits. *Journal of Cleaner Production*, 385:135711, 2023.

[33] Jean David Lau Hiu Hoong, Jérôme Lux, Pierre-Yves Mahieux, Philippe Turcry, and Abdelkarim Aït-Mokhtar. Determination of the composition of recycled aggregates using a deep learning-based image analysis. *Automation in Construction*, 116:103204, 2020.

[34] Yongbo Su. Multi-agent evolutionary game in the recycling utilization of construction waste. *Science of The Total Environment*, 738:139826, 2020.

[35] Peter Davis, Fayeem Aziz, Mohammad Tanvi Newaz, Willy Sher, and Laura Simon. The classification of construction waste material using a deep convolutional neural network. *Automation in Construction*, 122:103481, 2021.

[36] Sathish Paulraj Gundupalli, Subrata Hait, and Atul Thakur. A review on automated sorting of source-separated municipal solid waste for recycling. *Waste Management*, 60:56–74, 2017.

[37] Taillard Vincent, Mercier Guy, Pasquier Louis-César, Blais Jean-François, and Martel Richard. Physical process to sort construction and demolition waste (c&dw) fines components using process water. *Waste Management*, 143:125–134, 2022.

[38] Liang Yuan, Weisheng Lu, and Fan Xue. Estimation of construction waste composition based on bulk density: A big data-probability (BD-p) model. *Journal of Environmental Management*, 292:112822, 2021.

[39] Weisheng Lu and Junjie Chen. Computer vision for solid waste sorting: A critical review of academic research. *Waste Management*, 142:29–43, 2022.

[40] Kemal Özkan, Semih Ergin, Şahin Işık, and İdil Işıklı. A new classification scheme of plastic wastes based upon recycling labels. *Waste Management*, 35:29–35, 2015.

[41] Zhaokun Wang, Binbin Peng, Yanjun Huang, and Guanqun Sun. Classification for plastic bottles recycling based on image recognition. *Waste Management*, 88:170–181, 2019.

[42] Shuang Liang and Yu Gu. A deep convolutional neural network to simultaneously localize and recognize waste types in images. *Waste Management*, 126:247–257, 2021.

[43] Jianfei Yang, Zhaoyang Zeng, Kai Wang, Han Zou, and Lihua Xie. GarbageNet: A unified learning framework for robust garbage classification. *IEEE Transactions on Artificial Intelligence*, 2:372–380, 2021.

[44] Zeli Wang, Heng Li, and Xiaoling Zhang. Construction waste recycling robot for nails and screws: Computer vision technology and neural network approach. *Automation in Construction*, 97:220–228, 2019.

[45] Zeli Wang, Heng Li, and Xintao Yang. Vision-based robotic system for on-site construction and demolition waste sorting and recycling. *Journal of Building Engineering*, 32:101769, 2020.

[46] Zhiming Dong, Junjie Chen, and Weisheng Lu. Computer vision to recognize construction waste compositions: A novel boundary-aware transformer (BAT) model. *Journal of Environmental Management*, 305:114405, 2022.

[47] Wen Xiao, Jianhong Yang, Huaiying Fang, Jiangteng Zhuang, and Yuedong Ku. Classifying construction and demolition waste by combining spatial and spectral features. *Proceedings of the Institution of Civil Engineers—Waste and Resource Management*, 173:79–90, 2020.

[48] Yuedong Ku, Jianhong Yang, Huaiying Fang, Wen Xiao, and Jiangteng Zhuang. Deep learning of grasping detection for a robot used in sorting construction and demolition waste. *Journal of Material Cycles and Waste Management*, 23:84–95, 2020.

[49] Kunsen Lin, Tao Zhou, Xiaofeng Gao, Zongshen Li, Huabo Duan, Huanyu Wu, Guanyou Lu, and Youcai Zhao. Deep convolutional neural networks for construction and demolition waste classification: VGGNet structures, cyclical learning rate, and knowledge transfer. *Journal of Environmental Management*, 318:115501, 2022.

[50] Zhenhua Zhu and Ioannis Brilakis. Parameter optimization for automated concrete detection in image data. *Automation in Construction*, 19:944–953, 2010.

[51] Hyojoo Son, Changmin Kim, and Changwan Kim. Automated color model-based concrete detection in construction-site images by using machine learning algorithms. *Journal of Computing in Civil Engineering*, 26:421–433, 2012.

[52] Andrey Dimitrov and Mani Golparvar-Fard. Vision-based material recognition for automated monitoring of construction progress and generating building information modeling from unordered site image collections. *Advanced Engineering Informatics*, 28:37–49, 2014.

[53] Kevin K. Han and Mani Golparvar-Fard. Appearance-based material classification for monitoring of operation-level construction progress using 4d BIM and site photologs. *Automation in Construction*, 53:44–57, 2015.

[54] Alex Braun and André Borrmann. Combining inverse photogrammetry and BIM for automated labeling of construction site images for machine learning. *Automation in Construction*, 106:102879, 2019.

[55] Hadi Mahami, Navid Ghassemi, Mohammad Tayarani Darbandy, Afshin Shoeibi, Sadiq Hussain, Farnad Nasirzadeh, Roohallah Alizadehsani, Darius Nahavandi, Abbas Khosravi, and Saeid Nahavandi. Material recognition for automated progress monitoring using deep learning methods, 2020.

[56] Václav Nežerka, Jan Trejbal, and Tomáš Zbíral. Dataset of construction and demolition waste images: aerated autoclaved concrete (AAC), asphalt, ceramics, and concrete, February 2023.

[57] Frédéric Bosché, Mahmoud Ahmed, Yelda Turkan, Carl T. Haas, and Ralph Haas. The value of integrating scan-to-BIM and scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components. *Automation in Construction*, 49:201–213, 2015.

[58] V. Nežerka and J. Trejbal. Assessment of aggregate-bitumen coverage using entropy-based image segmentation. *Road Materials and Pavement Design*, pages 1–12, 2019.

[59] Yue Wu, Yicong Zhou, George Saveriades, Sos Agaian, Joseph P. Noonan, and Premkumar Natarajan. Local shannon entropy measure with statistical tests for image randomness. *Information Sciences*, 222:323–342, 2013.

[60] Jakub Antoš, Václav Nežerka, and Michael Somr. Assessment of 2D-DIC stochastic patterns. *Acta Polytechnica CTU Proceedings*, 13:1–10, 2017.

[61] F. N. M. de Sousa Filho, V. G. Pereira de Sá, and E. Brigatti. Entropy estimation in bidimensional sequences. *Physical Review E*, 105:054116, 2022.

[62] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:623–656, 1948.

[63] Yue Wu, Joseph P. Noonan, and Sos Agaian. A novel information entropy based randomness test for image encryption. In *2011 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2011.

[64] Bing Pan, Zixing Lu, and Huimin Xie. Mean intensity gradient: An effective global parameter for quality assessment of the speckle patterns used in digital image correlation. *Optics and Lasers in Engineering*, 48:469–477, 2010.

[65] Liang Yuan, Jingjing Guo, and Qian Wang. Automatic classification of common building materials from 3d terrestrial laser scan data. *Automation in Construction*, 110:103017, 2020.

[66] Mark S. Nixon and Alberto S. Aguado. Image processing. In *Feature Extraction and Image Processing for Computer Vision*, pages 83–139. Elsevier, 2020.

[67] Aurélien Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., 2022.

[68] Kevin P Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022.

[69] Zhi-Hua Zhou. Decision trees. In *Machine Learning*, pages 79–102. Springer Singapore, 2021.

[70] Sam Yu-Chieh Ho, Tsair-Wei Chien, Mei-Lien Lin, and Kang-Ting Tsai. An app for predicting patient dementia classes using convolutional neural networks (CNN) and artificial neural networks (ANN): Comparison of prediction accuracy in microsoft excel. *Medicine*, 102:e32670, 2023.

[71] Tomáš Zbíral and Václav Nežerka. Python codes for machine-learning-based classification of construction and demolition waste fragments, February 2023.

[72] Steven L. Salzberg. C4.5: Programs for machine learning by j. ross quinlan. morgan kaufmann publishers, inc., 1993. *Machine Learning*, 16:235–240, 1994.

[73] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29:1189–1232, 2001.

[74] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38:367–378, 2002.

[75] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Boosting and additive trees. In *The Elements of Statistical Learning*, pages 337–387. 2008.

[76] S. Madeh Piryonesi and Tamer E. El-Diraby. Using machine learning to examine impact of type of performance indicator on flexible pavement deterioration modeling. *Journal of Infrastructure Systems*, 27:04021005, 2021.

[77] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[78] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[79] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[80] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. Convolutional deep networks for visual data classification. *Neural Processing Letters*, 38:17–27, 2012.

[81] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[82] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84–90, 2017.

[83] Chaz Firestone. Performance vs. competence in human-machine comparisons. *Proceedings of the National Academy of Sciences*, 117:26562–26571, 2020.

[84] Wen Xiao, Jianhong Yang, Huaiying Fang, Jiangteng Zhuang, and Yuedong Ku. Development of online classification system for construction waste based on industrial camera and hyperspectral camera. *PLOS ONE*, 14:e0208706, 2019.

[85] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[86] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar Zaiane, and Martin Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. *Pattern Recognition*, 106:107404, 2020.

[87] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.