Lappeenranta-Lahti University of Technology LUT
School of Engineering Science
Computational Engineering and Technical Physics
Computer Vision and Pattern Recognition

Aleksandr Grigorev

UNDERSTANDING MULTIMODAL TIMBER MATCHING NETWORKS VIA ACTIVATION MAPS

Master's Thesis

Examiners: Assoc. Prof. Tuomas Eerola
Assoc. Prof. Igor Klebanov

Supervisors: M.Sc. Fedor Zolotarev
Assoc. Prof. Tuomas Eerola
Prof. Lasse Lensu
Prof. Heikki Kälviäinen

ABSTRACT

Lappeenranta-Lahti University of Technology LUT
School of Engineering Science
Computational Engineering and Technical Physics
Computer Vision and Pattern Recognition

Aleksandr Grigorev

Understanding multimodal timber matching networks via activation maps

Master's Thesis

2021

46 pages, 25 figures, 2 tables.

Examiners: Assoc. Prof. Tuomas Eerola
Assoc. Prof. Igor Klebanov

Keywords: convolutional neural networks, computer vision, encoder-decoder networks, multimodal translation, activation maps, sawmilling, wood defect detection

The use of digital technologies in timber tracing from the raw material to the end product, i.e., from logs to boards, provides benefits for the sawmilling industry such as better product quality prediction, efficient process control, and optimization. However, identification of boards from logs instantly becomes challenging after actual sawing, and that is why an automatic method for board identification was developed in previous studies. This thesis focuses on analyzing the multimodal encoder-decoder networks developed for timber matching to better understand what the network actually learns during the training process. The proposed solution is to compute activation maps and visualize the features learned by the neural network. Activations are further analyzed to estimate their ability to provide useful information based on the data collected from a sawmill. Finally, the use of activation maps for defect detection is studied. The results demonstrate that activation maps can be used to detect knots on both RGB images of boards and laser surface scans of logs in a weakly supervised manner.

PREFACE

This project helped me to take a fresh look at my current field of study and realize the importance of working in a team. I would like to thank my supervisors at LUT University for their help as well as my family for their inspiration and support.

Lappeenranta, May 31, 2021

Aleksandr Grigorev

CONTENTS

LIST OF ABBREVIATIONS

CAM	Class Activation Map
CNN	Convolutional Neural Network
FN	False Negative
FP	False Positive
IoU	Intersection over Union
ReLU	Rectified Linear Unit
RFID	Radio Frequency Identification
SVM	Support Vector Machine
TP	True Positive
XAI	Explainable Artificial Intelligence

1 INTRODUCTION

1.1 Background

Sawmilling is a complex industrial process that includes numerous different stages and variables. The digitalization of such a multi-level and difficult process provides various benefits for the enterprise such as efficient process control, process optimization, and product quality prediction. As a consequence, there is a need for measurement systems, as well as, process and quality control solutions to track the lifecycle of raw materials from the beginning to the end product, i.e., from logs to boards [1].

Timber tracing in a sawmill environment is important since it allows utilization of the measurement data from logs to improve the sawmilling process [1,2]. In practice, methods for wood tracking after each production step are required. For example, without these solutions, it would be challenging to track materials after actual sawing, and that is why a method is needed to identify which board comes from each log.

The latest solution for tracking the lifecycle from logs to boards is to perform board-to-log matching [3]. The proposed method utilizes laser scans of log surfaces and RGB images of boards to construct barcodes using multimodal encoder-decoder networks. The barcodes can be matched together as shown in Fig. 1. In order to collect the data, existing sensors in a sawmill are used.

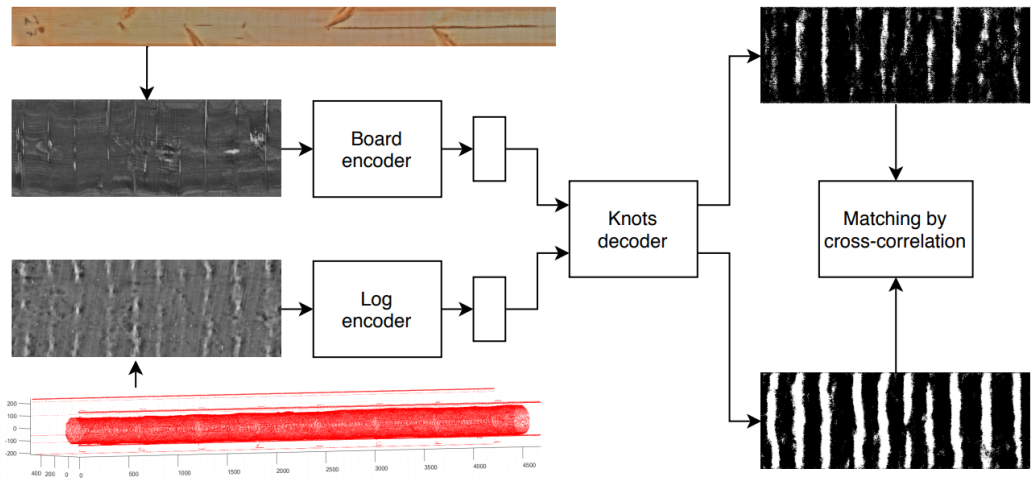

Figure 1. Timber matching using multimodal encoder-decoder networks. [3]

The proposed method relies on convolutional neural networks (CNNs) that have been widely used to solve complex computer vision tasks [4, 5]. A convolutional neural network is a class of deep neural networks where convolutional layers are used to filter inputs and extract useful information for decision making. However, such machine learning models are often perceived as a black box due to the untraceable training of the network, and that is why several techniques have been proposed to explain CNNs and achieve better model predictions [4].

One way to explain a convolutional neural network is to visualize features learned by the network. Nowadays, the most common way to do it is to compute class activation maps (CAMs) since the result is easy to interpret to a human [4]. The examples of different methods are shown in Fig. 2 [6] where the most important regions are red and the least important regions are blue.

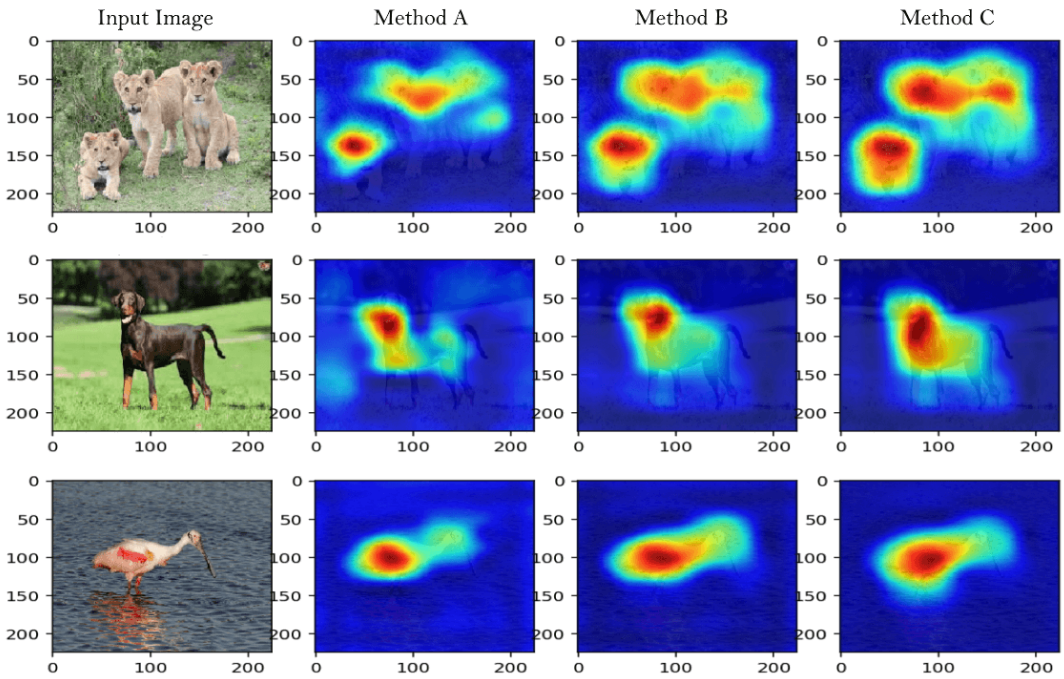

Figure 2. Visualization of the features learned by CNNs using different methods. Red regions correspond to strong activations with high impact on decision making. Blue regions correspond to weak activations with low impact on decision making. [6]

The purpose of this work is to make the existing multimodal encoder-decoder networks [3] more transparent and explainable by applying a method of computing activation maps. Moreover, this work aims to study whether CAMs can be used to detect interesting regions such as knots from both log surface scans and board images.

The Master's thesis is related to the DigiSaw [7] and DeepTimber research projects. The objective of these projects is to enhance the sawmill industry using the capabilities of modern digital systems such as computer vision.

1.2 Objectives and delimitations

This work aims is to develop a method capable of explaining encoder-decoder networks via activation maps. The activation maps are further analyzed and their ability to provide useful information is studied based on the data collected from a sawmill.

The specific objectives are as follows:

1. Analyze the architecture of the multimodal encoder-decoder networks used in timber matching and propose a method to compute activation maps.
2. Design and implement a method to compute activation maps using the data collected from a sawmill.
3. Evaluate the result with respect to color images of boards and laser scans of log surfaces, i.e., what the encoder-decoder networks use for timber matching.
4. Study the correspondence of strong activations and knot locations and develop a weakly supervised method utilizing the activation maps for knot detection.
5. Evaluate the performance of the developed method.

There are no delimitations in this work.

1.3 Structure of the thesis

The rest of the thesis is organized as follows. Chapter 2 presents background information on how machine learning is currently used in sawmilling. Chapter 3 introduces methods used to explain CNNs through visualization. Chapter 4 provides a summary of methods developed in this work. Chapter 5 evaluates the developed method performance for knot detection. Chapter 6 presents discussions about current results and possible future of the project. Finally, the conclusions are drawn from the work done in Chapter 7.

2 MACHINE LEARNING IN SAWMILLING

2.1 Sawmill process

The sawmill process includes a number of steps and basic operations which have remained unchanged for many years. A rough log, generally, goes through various mechanisms of cutting and emerges as a smooth finished product (lumber) [8, 9]. More specifically, the following steps are usually involved in modern sawmills (the most common steps are shown in Fig. 3 [9]):

- Felling trees. Loggers cut and fell trees in different woodlots and load them for transport to a sawmill site where they are sorted.
- Detecting metal objects. Various metal objects such as sign attachments or fencing wire get embedded in standing trees and are covered over by years of growth [8]. In order to prevent ruining a saw blade, it is vital to examine each log for metal contamination. If a log contains metal, it goes through a secondary examination before sawing.
- Debarking. As the first on-site sawmill step, every log is cleared of the bark, impurities, and damages to make their shape more cylindrical, and therefore simplify further sawing. The enterprise usually makes use of debarking machines such as ring debarkers, drum debarkers, flail debarkers, and rosser debarkers [10].
- Evaluating and sorting. The debarked log is estimated with laser scanning, X-ray tomography, or camera viewing for its maximum cut value before the actual cutting starts [11].
- Sawing, edging and trimming. The ready log is fed to a rig saw on a conveyor belt. Typically, different methods of cutting such as live sawing, cant sawing, tangential sawing, and radial sawing are applied [8]. Specific methods are usually selected based on the wood quality and desired result. Moreover, the product goes through edging and trimming stages where it gets its sides cut to fit a rough grade size.
- Grading. The final wood material is graded in a quality control process to ensure that lumber pieces get separated into similar batches [12].
- Drying wood. The product either air-dries or gets placed in kilns for forced drying to avoid mold, pests, or deformation. It is also treated with antiseptics to protect it from moisture and temperature [13].

- Sorting, packaging and loading for delivery. The boards are the final raw material that can be taken to a storage location after sorting and packaging. At this stage, the product is ready to be loaded for delivery.

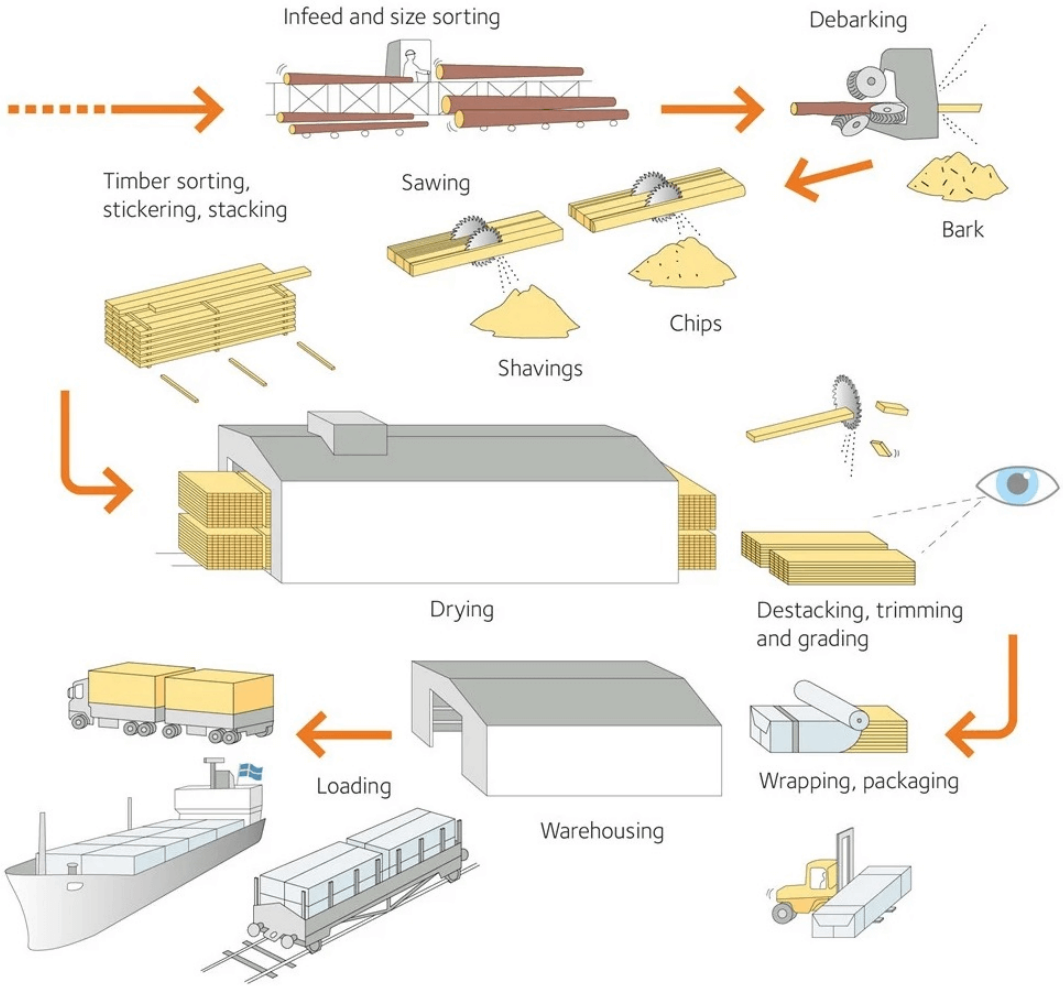

Figure 3. The sawmill process from woodlots to a sawn wood product. [9]

Exact operations and their order vary in each enterprise since the sawmill process mostly depends on the desired result. For example, there are boards of different size, wood chips, or shavings [9]. However, the industry tries to maximize the efficiency of the log usage and the final product quality which is why process automation is always considered to get the optimal amount of wood.

2.2 Timber tracing techniques

Timber quality is an important raw material feature which can be obtained during the whole production process. The enterprise tracks wood in a sawmill through all the stages as it provides several benefits such as efficient process control and optimization, product quality prediction at early stages, and more reliable final material grading [14]. However, in practice, timber tracing requires the development of advanced methods to track wood at the end of each production step. The rough log loses most of its distinctive features after debarking and sawing what makes it difficult to trace timber and define the value of the end product in a sawmill. Debarking removes information about external defects while sawing distorts information about the original shape and size. However, the structure and location of the log knots do not change after sawing considerably [15]. As a result, knots of a board are considered as a feature to make a conclusion about the original log.

In order to trace timber in a sawmill, various approaches have been proposed. Some solutions re-identify logs before sawing (log-to-log matching) while others match boards before and after drying (board-to-board matching) [3]. The traditional approach is to mark each log with paint or serif while transporting and storing. However, mechanical damages and remarkable wood transformations complicate this process. Other techniques such as Radio Frequency Identification tags (RFID-tags) for logs provided high tracing accuracy, but turned out to be too expensive and complex [16].

Nowadays, it is also possible to trace timber in a sawmill from the beginning to the end of the production process using modern measurement tools. Typically, such measuring tools are laser scanners and digital cameras measuring both the raw material and the final product during different production steps [3, 17]. However, information on an individual log or board is lost when the material goes through the production steps because the measurements are not connected with each other. Consequently, there is a great interest to utilize some measurable wood properties to match boards to logs during the whole process.

It was previously shown that methods based on machine vision are more effective in terms of operational costs for timber tracing and achieve high accuracy [18–20]. For example, two-dimensional to one-dimensional projection signals [19] and template matching [21] have been proposed to trace timber using digital camera systems. Moreover, encoder-decoder networks have been proposed recently to match boards to logs using only the existing sawmill sensors such as camera systems and laser scanners [3].

2.3 Defect detection

Knots represent the main varietal timber defect enclosed in the wood trunk. Their quantity, types, sizes, and locations are the main factors defining the value and quality of the end product which is why there are many methods for detecting the knot positions in logs and boards [20]. In order to detect defects, laser scanning is mostly used to describe a log with a 3D model, RGB imaging is utilized to detect knots on boards after sawing, and X-ray tomography is required to obtain information about the rings inside the log. Typically, laser scanning and X-ray tomography are performed after debarking while RGB imaging is required only after sawing.

Moreover, laser scans are useful to observe the material shape and surface, and therefore they allow to detect mechanical damages or defects, e.g., knots and branches, and deformations of boards after sawing or drying. At the same time, camera systems provide useful information about the wood appearance such as minor structure changes before and after drying.

2.3.1 Knot detection from laser scans of log surface

Knot detection from the log surface scans is possible to implement using the data on the structure of a log surface. Laser scanning is often utilized since it allows obtaining a log model in high resolution. Typically, the process of knot detection from log surfaces includes the following steps [22]:

1. Scan the log surface.
2. Create the 3D log model.
3. Detect deformations and rises on the log surface.
4. Select knots among all other defects.

Obviously, laser scanning provides information about external defects only. As a consequence, another method such as RGB imaging or X-ray tomography is required to obtain information about internal defects.

2.3.2 Knot detection from board images

Knot detection from the images of boards is a relatively simple task since knots clearly stand out from the sawn timber background. Moreover, their position, shape, and size do not change significantly during the sawmill process [23].

The state-of-the-art methods extract specific features from the board images to evaluate the knot areas and make the required conclusion about the board. For instance, several methods based on support vector machines (SVM) [15, 24] have been proposed to detect knots from the board images with high accuracy. Similar results were achieved in [25] where the wood defect identification is based on principal component analysis and neural networks. One of the recent solutions to detect locations and classify types of different wood defects has been proposed in [20] where a fully convolutional neural network provided better accuracy and faster detection in comparison to previously developed wood defect detection methods.

2.3.3 Knot detection from X-ray tomography

The X-ray computer tomography is a non-destructive method which allows analyzing the internal wood structure as shown in Fig. 4. With the X-ray tomography, it is possible to observe tree rings and internal defects such as knots before sawing [26]. Moreover, knots have a higher density which is why it is simple to detect them using the X-ray tomography data.

In addition to laser scanning and RGB imaging, X-ray tomography helps to obtain information about moisture, age, and tree species before a log is sawn into multiple boards. Having both X-ray tomography data and laser scans, a 3D model can be constructed to reflect both internal and external log defects [27].

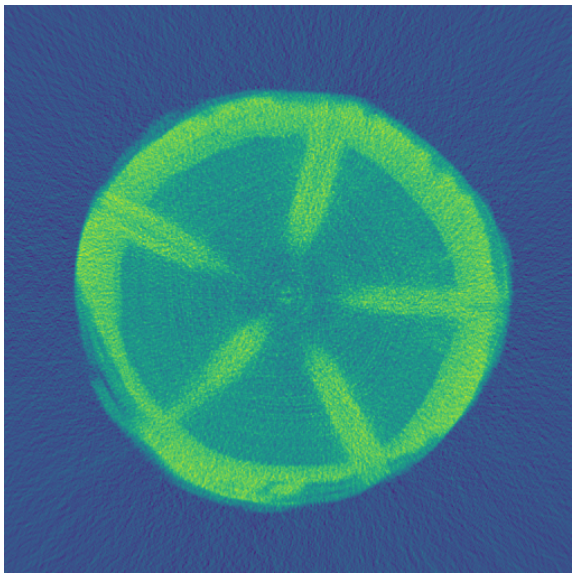

Figure 4. Internal wood structure observed with the X-ray tomography.

3 CONVOLUTIONAL NEURAL NETWORKS

3.1 Network architecture

Convolutional neural networks imitate the visual cortex of the human brain and differ from other machine learning models in that knowledge about images is considered as a specific type of input data [28]. Therefore, it allows to set up a simpler network architecture in comparison to other machine learning models, provides better feature generalization during training and has lower computational requirements [29, 30].

Nowadays, the primary application of CNNs is to solve image-driven pattern recognition tasks. Their performance on various computer vision applications is close to the level of human abilities [5]. For example, CNNs perform well in such tasks as image classification [31], object detection [32], and semantic segmentation [33].

The idea behind CNNs is to use parts of an image to find certain features rather than use the entire image at once. A typical example of the convolutional neural network architecture is shown in Fig. 5 [31, 34]. The architecture represents a sequence of convolutional and pooling layers designed to learn increasingly higher-level features. The input, e.g., an image in the RGB format, is fed to a neural network where the first layers extract primitive features such as dots, lines and edges. The next layers extract complex features such as objects and shapes. An activation function such as ReLU is appended to each convolutional layer to introduce non-linearity into the network. At the end of the sequence, one or multiple fully connected layers are used to map the output features into scores [4].

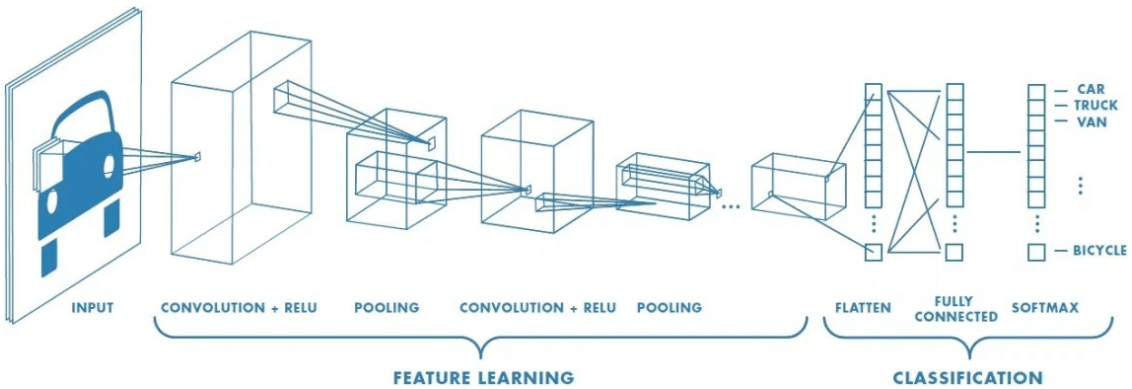

Figure 5. Typical CNN architecture. [34]

Input

The input is usually a three-dimensional grid in the case of RGB images or a one-dimensional grid in the case of grayscale images. The input size I is defined as:

$$I = W \times H \times D \tag{1}$$

where W is an image width, H is an image height, and D is an image depth or the number of image channels.

Convolutional layer

The convolutional layer is the most important layer in CNNs [28]. If the input is a grayscale image, the convolutional layer extracts features using a $N \times N$ convolutional filter. This filter in turn extracts the same feature from the input image at different spatial locations. Therefore, the output of each layer depends on a small region of the input. However, if the input is an RGB image, a convolutional layer consists of multiple feature maps as well as the output. In this case, the filter size will be $N \times N \times K \times M$ where K is the number of input feature maps and M is the number of output feature maps.

Activation function

Activation functions inside a neural network introduce non-linearity into the forward path of the network. Typically, the functions are appended to each convolutional layer and applied to each output node independently. There are various types of the activation functions such as Sigmoid, Tanh, and ReLU for convolutional neural networks. However, ReLU (Rectified Linear Unit) is currently the most commonly used function [35]. It keeps all positive values as is and sets negative values to zero:

$$y = \max(0, x) \tag{2}$$

where x is the input to a neuron.

The main drawback of the ReLU activation function is being unbounded, i.e., activations are not in a reasonable number range during training. It is important to keep the data in a reasonable range since huge numbers cannot be represented accurately with 32-bit or 64-

bit floating points. Besides, high values in feature maps should represent strong feature existence. If a value down the layer chain becomes huge, it is impossible to estimate whether a given feature is actually present. Typically, normalization layers are introduced at regular intervals within the network to solve this issue. Nowadays, batch normalization is usually applied since training is easier to set up and goes faster while the accuracy is higher [35].

Pooling layer

The pooling or sub-sampling layer reduces the dimensionality by transforming several neurons into a single neuron in order to reduce the probability of rapid retraining and computing costs [31]. Typically, this layer is applied to the output of the convolutional layer, and thus it highlights the most important signals based on specific criteria. As an alternative to the pooling layer, padding and stride are also utilized in some applications [36].

Two types of pooling layers are commonly used in convolutional neural networks. The one type is average pooling where the output node is equal to the average value of the input region. The other type is max pooling where the output node is equal to the maximum of the input region. Nowadays, max pooling is widely used for the tasks of computer vision. The pooling layer uses a window of a certain size and applies one of the rules to compute a sub-sampling layer as shown in Fig. 6 [34].

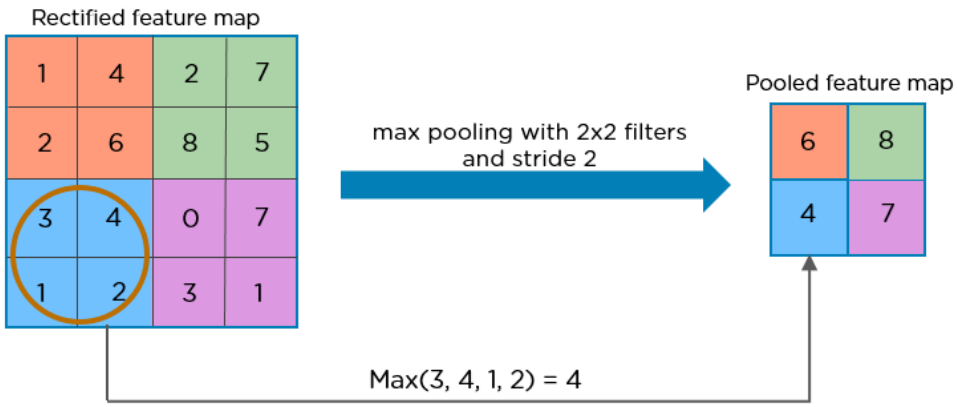

Figure 6. Pooling layer example. [34]

Fully connected layer

The fully connected layer is a one-dimensional layer extracting features over the whole input after its flattening, and therefore each output node depends on all input nodes. The purpose of this layer is to transform all the signals from a network to the one-dimensional form [31].

Softmax

The final layer of a typical convolutional neural network is often a Softmax layer which is required to get the output as a vector of probabilities. In other words, it normalizes the output from the neural network such that the sum of all nodes is 1 and is defined as follows:

$$y = \frac{e^x}{\sum e^x} \tag{3}$$

where x is the output of a neuron.

3.2 Training

The process of training a convolutional neural network typically requires to initialize the weights in all hidden layers and consists of the following steps [37]:

1. Forward propagation or forward pass. Forward propagation sequentially calculates and stores intermediate variables within the computational graph defined by the neural network. It proceeds from the input to the output layer. The output from one layer becomes the input to the next one.
2. Backpropagation or backward pass. Backpropagation sequentially calculates and stores the gradients of intermediate variables and parameters within the neural network in the reversed order. It updates the neuron weights to reduce the error of decision making.

Forward propagation and backpropagation are interdependent in the process of training. Moreover, loss functions are used in training as an error measure to compute gradients and update the neural network weights [38].

3.3 Encoder-decoder networks

Matching the data with different modalities requires to convert the data into similar representations. In this case, encoder-decoder networks are typically used [39, 40]. They are based on the concept of fully convolutional networks and differ from traditional CNNs in that the fully connected layers are not included in the network architecture, but instead, the input and output of the network are of the same size [33].

The fully convolutional neural network consists of two main parts (see Fig. 7 [40]):

1. Encoder. The input image is fed to the encoder where it is downsampled and passed to a decoder as a feature map.
2. Decoder. The output image is received from the decoder where the feature map is upsampled back to the original size and translated into the required representation.

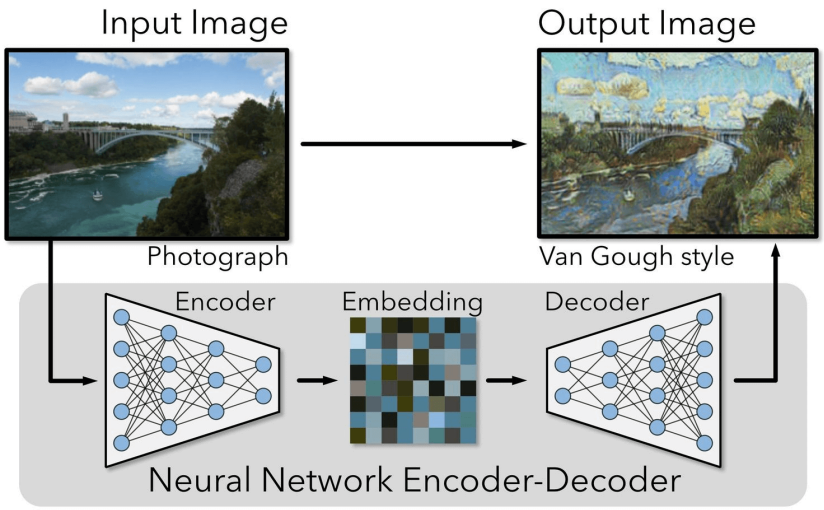

Figure 7. Encoder-decoder network used to transform a photograph into a drawing of the Vincent van Gogh style. [40]

Nowadays, the fully convolutional networks are broadly used in a variety of applications [33]. The most recent examples of such network architectures are U-NET [41] and SegNet [39].

3.4 Interpretability and explainability

In order to move towards the integration of intelligent systems into various fields of life, transparent and reliable models have to be built. This goal can be achieved through interpretation and explanation of why intelligent systems do what they do. Nowadays, theory, frameworks, and tools are being actively developed for Explainable AI (XAI). Particularly, there has been a significant interest in developing explainable convolutional neural networks as an important part of deep learning [42].

There is a problem of neural network interpretability and explainability despite the fact that they allow to produce accurate models from data and extract relevant patterns. There are ongoing debates on the definition of these two concepts in terms of AI, and therefore improper use is possible. In the scope of this work, the following definitions are used according to [4]:

- interpretability is the ability to provide meaningful information to a human in understandable terms;
- explainability is an interface between a decision-maker and a human which denotes any action or procedure taken by a model with the intent of clarifying or detailing its internal functions.

In [43], the impact of model interpretability and explainability on some areas is explained. For example, doctors would be able to review a model prediction and provide feedback to enhance the model; engineers would be able to trace reasons why an autonomous vehicle behaved in a certain way and took a certain action; systems for object detection would provide a list of features used to classify an object. It raises the question of developing methods to both interpret and explain machine learning models. In order to reduce the scope of this work, this question is discussed in terms of CNNs only.

The architecture of CNNs includes complex internal relations that are extremely difficult to explain. However, in comparison to other deep learning models, it is still easier to explain CNNs than other models because people are able to understand visual data. There are currently two common ways to understand the learning process of CNNs [43]:

1. Map the output into the input space to highlight parts of the input which were discriminative for the output.

2. Analyze the visual information contained inside the network (how the inner layers see the world in general).

3.5 Visual interpretability

In order to make CNNs reliable and overcome failing without providing any plausible explanation, various methods have been proposed over the past few years as an attempt to explain their decisions [4]. These methods consider the model inputs (images) as visually interpretable by humans, and thus locate specific image regions the model "looked at" to assign a class label to the image. This approach allows to understand how CNNs work, what they learn and why they work in a given manner.

Gradient visualization and class activation maps (CAMs) are the examples of the most widely adopted methods to explain CNNs. Gradient visualization is one of the earliest efforts to understand CNNs and includes several methods based on the idea of the gradient backpropagation such as described in [44, 45]. Nowadays, class activation maps are more preferable in comparison to gradient visualization [46].

Class activation mapping is a technique used to understand what parts of the image led a convolutional neural network to make a specific decision. This method identifies discriminative regions by linearly weighted combination of activation maps of the last convolutional layer before the global pooling layer [47]. It produces a heatmap over the input image indicating the importantance of each location with respect to an object class (see examples in Fig. 8). Having a heatmap over the image helps to verify the decision process of CNNs more easily.

It is possible to compute heatmaps for different neural network layers to understand which parts of the image are being activated in each layer since earlier layers extract primitive features [48]. However, the class prediction mostly depends on the last layer activations which is why it makes sense to visualize heatmaps for the last convolutional layer only.

Most recent visualization methods are CAM [47], Grad-CAM [44], Grad-CAM++ [45], Smooth Grad-CAM++ [6] and Score-CAM [46]. CAM is able to identify the importance of different parts of the image only if the model is designed with a global average pooling layer. Grad-CAM is a generalization of CAM which is applicable to any CNN-based architectures and does not require re-training. Two other methods, Grad-CAM++ and Smooth Grad-CAM++, represent further development and improvements of Grad-

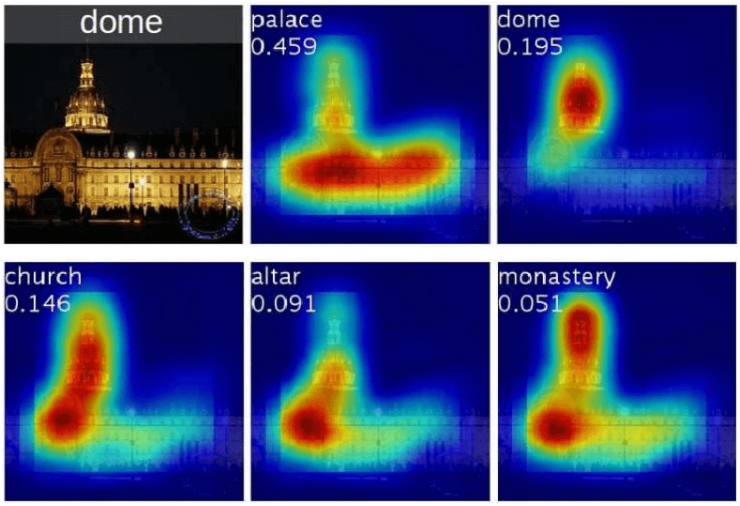

Figure 8. The highlighted regions vary across predicted classes on the class activation maps generated from the top 5 predicted categories for the given image with the ground truth as dome. [47]

CAM with better localization of objects. Grad-CAM and its variations backpropagate the gradient of a target class to the input to highlight specific image regions influencing the prediction. Unlike gradient-based approaches, Score-CAM gets rid of the dependence on gradients by obtaining the weight of each activation map through its forward passing score on a target class, and the final result is obtained by a linear combination of weights and activation maps. The comparison of Grad-CAM, Grad-CAM++ and Score-CAM is shown in Fig. 9 [46] for two different images.

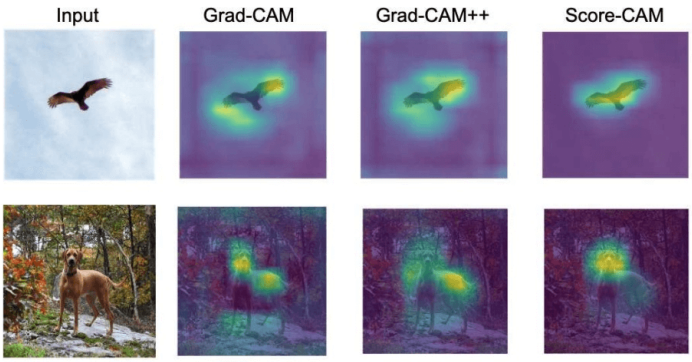

Figure 9. Comparison of different methods. Score-CAM focuses better on relevant objects than Grad-CAM and Grad-CAM++. [46]

4 ACTIVATION MAPS AND KNOT DETECTION

The activation maps allow to visualize the features learned by the encoder-decoder networks used in timber matching. These features can be further utilized to detect interesting regions such as knots in a weakly supervised manner.

4.1 Timber tracing with encoder-decoder networks

In [3], an automatic method for board identification (board-to-log matching) was proposed to enable tracing of materials in a sawmill environment. This method relies on the existing sawmill sensors and convolutional multimodal encoder-decoder networks to match boards to logs using the features learned by the neural network. The proposed method utilizes grayscale images of boards and laser scans of the corresponding log surfaces as point clouds.

The steps of matching boards to logs are as follows (see Fig. 10) [3]:

1. Generate heightmaps for laser scans of log surfaces using the point clouds by converting Cartesian coordinates to log-centric coordinates.
2. Translate the images of boards and heightmaps into "barcode" images (matchable representations) using multimodal encoder-decoder networks.
3. Apply cross-correlation using the "barcodes" to find the best match for each board, i.e., the log which the board was sawn from.

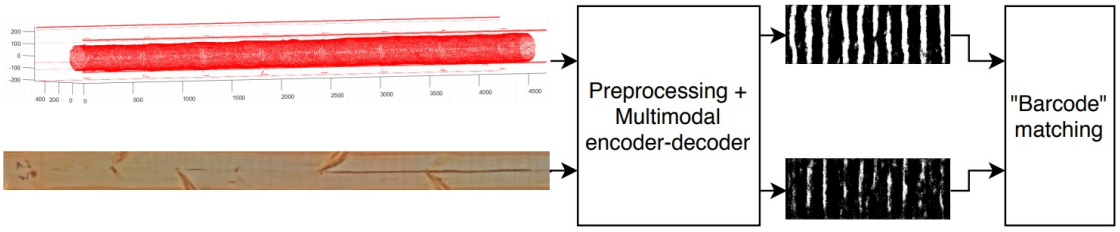

Figure 10. Process of matching boards to logs based on the log surface scans and board images. The multimodal encoder-decoder networks translate the data from both modalities into "barcodes" to match boards to logs by applying cross-correlation. [3]

The neural network is trained using "barcode" images which represent starting and end points of knot clusters obtained from the discrete X-ray tomography [3]. The neural network is able to generate similar "barcodes" using both the images of boards and the heightmaps. Laser scanning and RGB imaging are required only for production deployment once the network is trained with the discrete X-ray tomography data.

As an attempt to verify the decisions of the proposed multimodal encoder-decoder networks, this study mainly suggests solutions for the following tasks:

- visualization of what the neural network learns in the training process;
- detection of knot locations using the data obtained from the visualization.

These solutions include data preprocessing, computing activation maps and displaying them over the images of boards or laser scans of log surfaces to visualize and analyze how the neural network makes its decisions. The activation maps are further utilized to detect knots from both modalities using various techniques of filtering and segmentation.

4.2 Visualization of neural network decisions

The main task of the multimodal encoder-decoder networks studied in this work is to match boards to the corresponding logs using the log laser scans and the board images. The previous study [3] assumes that it is possible due to the neural network learning the knot locations from both modalities. In order to verify this assumption, the neural network should have the strongest activations in the regions where knots are present. The activations can be shown on the images the neural network operates with.

4.2.1 Data preprocessing

The goal of data preprocessing is to transform the data into the format the neural network can operate with. Ideally, the steps of the data preprocessing must be the same in both visualization and training. In this case, the following categories of images must be preprocessed before visualization:

- RGB images of boards (see the example in Fig. 11);

- grayscale images of log heightmaps generated by converting Cartesian coordinates to log-centric coordinates [14] (see the example in Fig. 12).

Figure 11. Example of the board image.

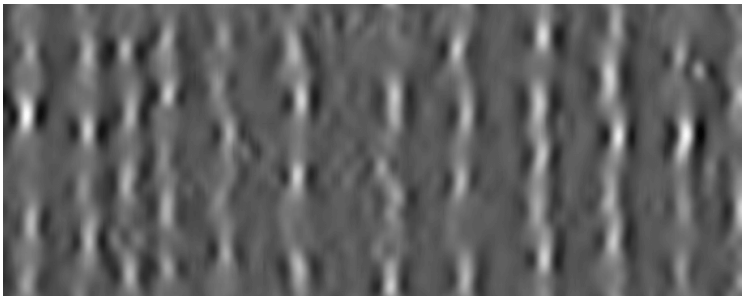

Figure 12. Example of the heightmap image.

The proposed method includes the following basic operations:

1. Convert images to grayscale.
2. Normalize images using the min-max scaling in the range from 0 to 255.
3. Rescale images to the lower resolution previously used in training.

The preprocessed images can be passed into the trained neural network to compute activation maps and analyze what regions have the highest impact on decision making.

4.2.2 Computing activation maps

The proposed method relies on Gradient-weighted Class Activation Mapping (Grad-CAM) which was originally introduced in [44] to debug deep learning models visually and properly understand what regions on the image are the most important in decision making. Grad-CAM utilizes the gradients of any target concept in the final convolutional layer to highlight the important regions.

The idea of this approach can be described for CNNs as follows:

1. Find the final convolutional layer in the neural network.
2. Construct the gradient model by supplying:
   (a) the inputs to the pre-trained model;
   (b) the output of the final layer in the neural network;
   (c) the output of the softmax activations from the model.
3. Compute the gradients towards the required class using automatic differentiation. In the case of encoder-decoder networks, the required class is the model output.
4. Compute the guided gradients to eliminate elements acting in a negative way towards the decision by zeroing out the negative gradients or gradients associated with a negative value of the convolutional filter.
5. Compute the average of the gradient values as weights to establish the importance of each convolutional filter in the decision.
6. Sum up all mathematically weighted maps into a final heatmap.

If an activation map has been lightened up during the forward pass and its gradients are large, the activated region has a large impact on the decision.

4.2.3 Overlaying activation maps on images

As the final step of the visualization, the heatmap should be displayed over the original image to evaluate the result visually. This process requires to get the dimensions of the original image (see examples in Figs. 13(a) and 14(a)), i.e., width and height in pixels, rescale the grayscale heatmap to the higher resolution which matches the resolution of the original image (see examples in Figs. 13(b) and 14(b)), colorize the heatmap by applying a pseudo/false-color to it (see examples in Figs. 13(c) and 14(c)) and transparently overlay the heatmap on the original image. The examples of the final visualization after applying these operations are shown for both images of boards and heightmaps in Figs. 13(d) and 14(d).

In order to apply a pseudo/false-color to a grayscale heatmap, it is required to use one of the perceptually uniform sequential colormaps (see examples of colormaps in Fig. 15). For example, with the VIRIDIS colormap, darker input grayscale values result in a dark purple RGB color while lighter input grayscale values result in a light yellow RGB color.

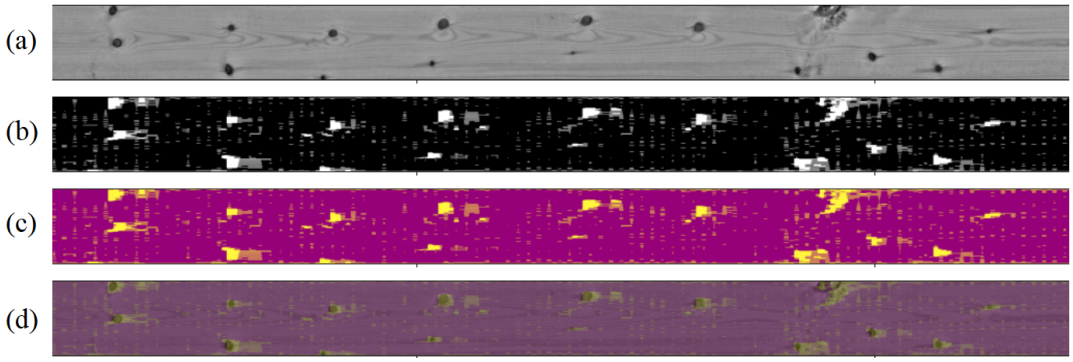

Figure 13. Displaying the heatmap over the board image: (a) Original image; (b) Grayscale heatmap; (c) Colorized heatmap; (d) Final result.

4.3 Weakly supervised knot detection from board images and laser scans

According to the examples of the visualization, knots clearly stand out and can be detected as regions or contours using various image processing techniques such as thresholding, filtering, and edge detection. More specifically, the algorithm consists of the following operations:

1. Specify the range where activations should be taken into account.
   (a) Choose the upper limit. This step requires to find the maximum value of activations on the heatmap. In most cases, the maximum value corresponds to the true knot locations.
   (b) Choose the lower limit. This step requires to specify a certain value of activations on the heatmap as the minimum value that can be taken into account. In most cases, this value should be slightly lower than the upper limit to exclude areas with the lowest impact on decision making.
2. Apply a mask by removing all activations outside the specified range. If the algorithm removes (saves) too many activations, the lower limit should be reduced (increased).
3. Smooth the mask by applying a median filter. The size of the filter should be chosen empirically to remove some extra noise and improve the shape of regions.
4. Filter the mask using the closing morphological transformation of the elliptical shape. The size and the number of iterations are chosen empirically. The elliptical shape is chosen due to knots usually have such a shape.

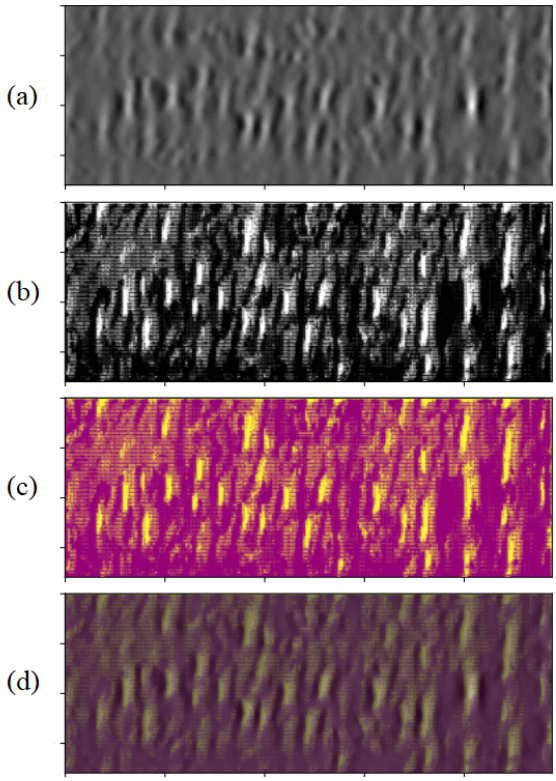

Figure 14. Displaying the heatmap over the heightmap: (a) Original image; (b) Grayscale heatmap; (c) Colorized heatmap; (d) Final result.

Figure 15. Perceptually uniform sequential colormaps. [49]

5. Transform the filtered mask into a binary image by treating non-zero pixels as ones.
6. Find contours in a binary image using an edge detection algorithm.
7. Iterate over the found contours and remove too small contours using the bitwise AND. A contour is considered too small if its area is less than the specified value which can be chosen empirically.

The described algorithm can be applied to both images of boards and heightmaps. However, specific parameters may be different in each case since they are usually chosen experimentally. The overall process and some of its steps are shown for images of boards in Fig. 16 and laser scans of logs in Fig. 17.

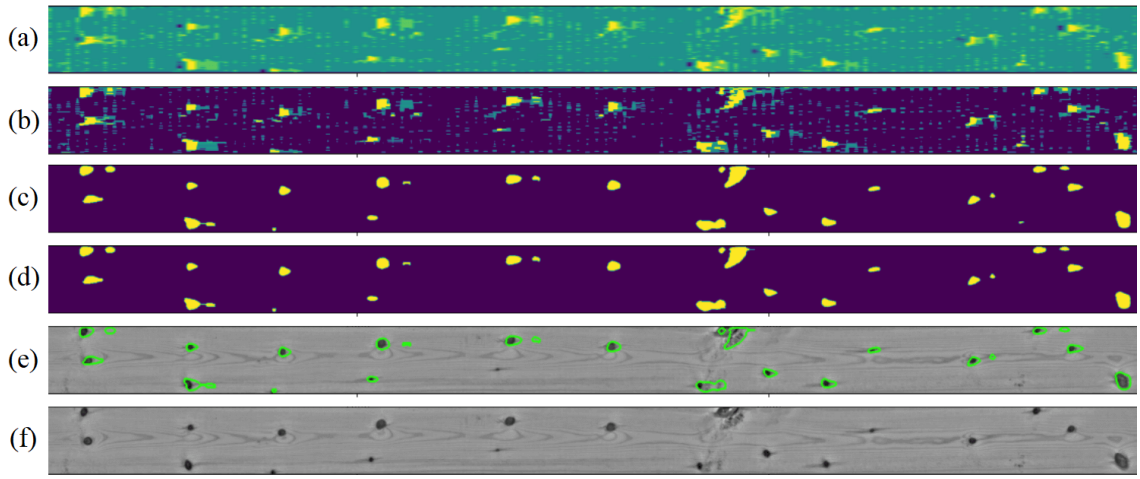

Figure 16. Main steps of the knot detection on the images of boards: (a) Rescaling the heatmap; (b) Removing weak activations; (c) Smoothing and filtering; (d) Excluding too small areas; (e) Edge detection; (f) Original image for comparison.

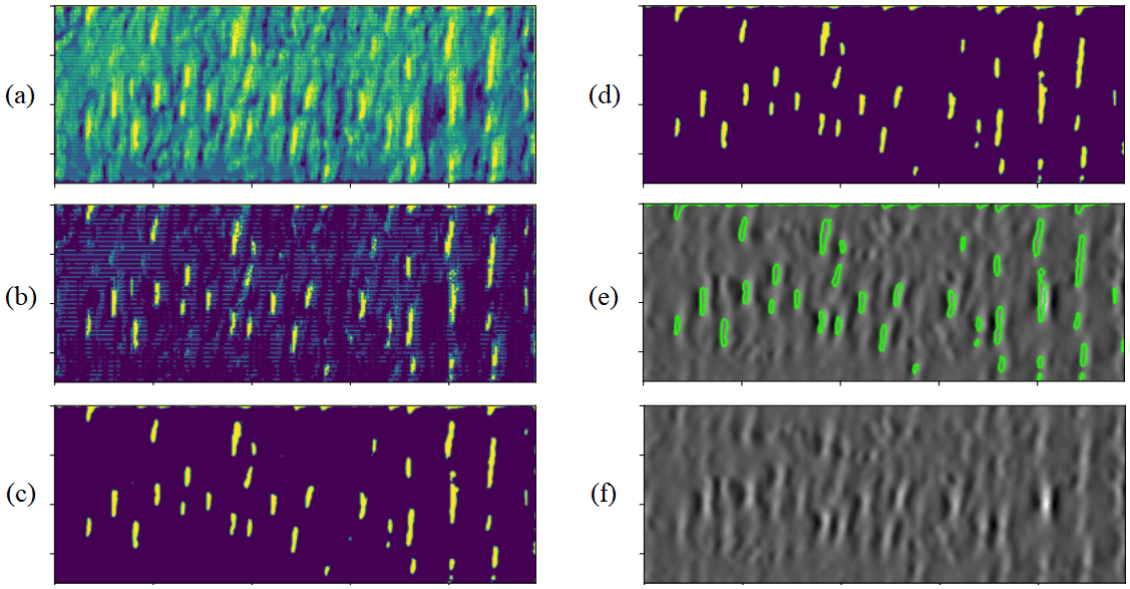

Figure 17. Main steps of the knot detection on the heightmaps: (a) Rescaling the heatmap; (b) Removing weak activations; (c) Smoothing and filtering; (d) Excluding too small areas; (e) Edge detection; (f) Original image for comparison.

5 EXPERIMENTS

5.1 Data

The data for experiments was obtained using 100 debarked logs and 270 boards sawn from those logs [14]. In the process of the data preparation, each log was scanned multiple times and the log heightmaps were generated based on the point clouds after each scan. Then, logs were sawn into boards and each board lower and upper sides were imaged with an RGB camera system. Moreover, all logs were marked manually to make it possible to track them through the whole sawmilling process and provide the ground truth for the board-to-log matching. The proposed method operates on the following data:

- 435 images of sawn timber boards;
- 300 images of generated log heightmaps.

This data was collected in the real sawmill environment. All the logs were automatically graded using an existing grading system in a sawmill where grades A, B and C are low quality while grades D and E are high quality. A semi-automatic knot segmentation was applied to all the board images to get the ground truth which can be utilized to evaluate the performance of the proposed method of weakly supervised knot detection for the board images. Moreover, possible knot locations were identified [14] on the log heightmaps using the assumption about bright areas corresponding to knots since there is no information about the true knot locations on the heightmaps. However, the real number of knots, their real shapes and positions may differ from the assumption significantly. The collected data examples are shown in Figs. 18 and 19.

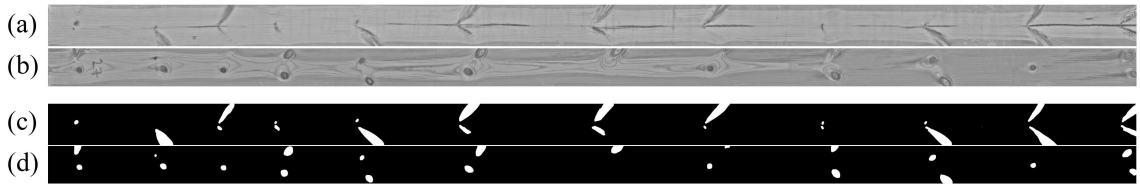

Figure 18. Collected data examples: (a) Lower side of the board; (b) Upper side of the board image; (c) Ground truth for the lower side of the board; (d) Ground truth for the upper side of the board.

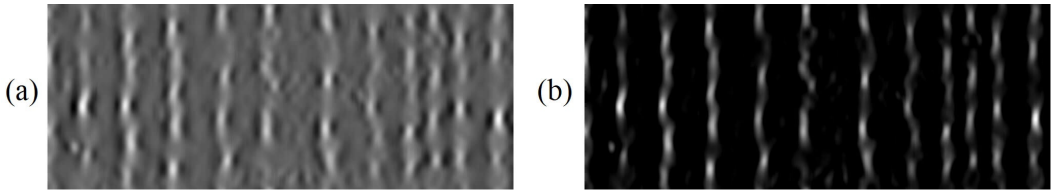

Figure 19. Collected data examples: (a) Log heightmap; (b) Possible knot locations on the heightmap.

According to the given examples, defects clearly stand out on the images of boards in the collected data. However, the heightmaps do not allow to identify knot locations easily, and only the knot clusters can be recognized visually as vertical bright lines. Consequently, only the ground truth for the board images can be used to evaluate the proposed method performance.

5.2 Implementation

The experiments were performed using Python 3.8.8, OpenCV 4.5.1, and TensorFlow 2.3.1 on the server with the following computational resources: NVIDIA GeForce GTX 1080 Ti/TITAN RTX GPU, Intel Xeon CPU E5-2680 and 256 GB RAM.

The multimodal encoder-decoder networks were initialized using the pre-trained model proposed in [3]. The neural network was not modified in any way to perform the experiments.

Grad-CAM was implemented from scratch to make it compatible with the given Python and TensorFlow versions. Moreover, the original implementation of Grad-CAM proposed in [44] was adjusted to enable it on the pre-trained model.

5.3 Description of experiments

This study suggests two experiments to evaluate the proposed method performance:

1. segment knots on the computed heatmaps for the images of boards and verify knot locations using the segmentation ground truth;

2. segment knots on the computed heatmaps for the heightmaps in order to detect possible knot locations.

Both experiments are based on the same method of weakly supervised knot detection introduced in Section 4.3. However, different parameters are needed to adjust the method for each modality. Table 1 shows the parameter values used in both experiments. The values were selected empirically based on the experiments.

Table 1. Parameters used in knot detection from images of boards and laser scans of log surfaces.

Method Parameters	Boards Segmentation	Heightmaps Segmentation
Activations: upper limit	100%	100%
Activations: lower limit	75%	90%
Median filter size	15	9
Morph. transform. size	3×3	5×5
Morph. transform. iterations	1	2
Contour detection	Extreme outer contours	Extreme outer contours

The purpose of these experiments is to estimate the ability of the multimodal encoder-decoder networks to learn true knot locations in the training process. In this case, the performance can be evaluated on individual samples and groups of samples.

5.4 Evaluation criteria

The proposed method of weakly supervised knot detection can be considered as a binary classifier. In this case, the individual knot position can be either true or false meaning the individual model output is right or wrong [50]. Consequently, there are the following possible outcomes:

- True Positive (TP): the detected knot location is correct and corresponds to the true knot location.
- False Positive (FP): the detected knot location is incorrect and does not correspond to any true knot location.
- False Negative (FN): the true knot location was not detected.

With these possible outcomes, the following metrics are computed to evaluate the proposed method performance in the experiments: precision, recall, and F1-score. These metrics serve as a measure to assess how well the method performs in the task of object detection.

Precision is the fraction of relevant instances among all retrieved instances:

$$Precision = \frac{TP}{TP + FP}. \tag{4}$$

Recall is the fraction of retrieved instances among all relevant instances:

$$Recall = \frac{TP}{TP + FN}. \tag{5}$$

Both precision and recall range from 0 to 1. If precision is high and recall is low, most found objects are correct, but most ground truth objects are missed, i.e., there are many false negatives. However, if precision is low and recall is high, most ground truth objects are detected properly, but most found objects are incorrect, i.e., there are many false positives. Finally, if both precision and recall are high, the detector found most ground truth objects correctly.

F1-score leverages both precision and recall metrics to summarize them into a single metric as follows:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}. \tag{6}$$

Moreover, Intersection over Union (IoU) is usually used as an evaluation metric in the tasks of object segmentation and measures similarity between finite sample sets. This coefficient is defined as the size of the intersection divided by the size of the union of two sample sets as:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{7}$$

where A is the ground truth and B is the model prediction.

5.5 Results

The proposed method provides the visualization of the features learned by the neural network and allows to detect specific knot locations using the learned features only. The detected knots are further used to evaluate the method performance.

5.5.1 Board images

The examples of board images, corresponding visualizations, detected knots, and true knot locations are shown in Figs. 20 and 21. According to the visual explanation of the neural network decisions, it tends to learn true knot locations on the images of boards as important features. However, it also learns the timber texture in addition to knots in such cases as shown in Fig. 22.

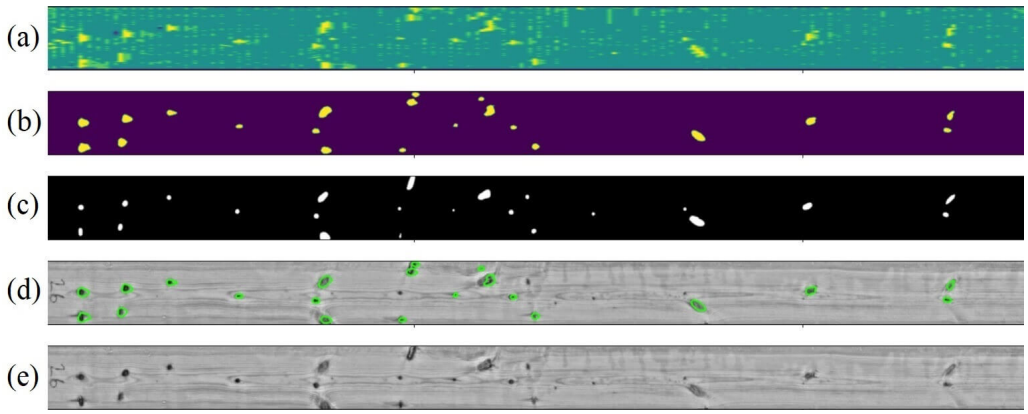

Figure 20. The learned features mostly match true knot locations (precision: 0.94, recall: 0.86, F1-score: 0.90, IoU: 0.29): (a) Computed heatmap; (b) Filtered heatmap; (c) True knot locations; (d) Detected knot locations; (e) Original image.

The model demonstrates promising performance with high precision, recall, and F1-score on the individual samples. The metrics prove the assumption about the model learning true knot locations to match boards to logs. However, IoU is relatively low for all individual samples because of an obvious imbalance in the ratio between the board texture and knots.

Moreover, the method was evaluated for boards in different groups. According to Table 2, precision varies from 44% to 75%, recall varies from 85% to 91%, F1-score varies from 54% to 77%, and IoU varies from 21% to 29% in the selected groups of samples. The

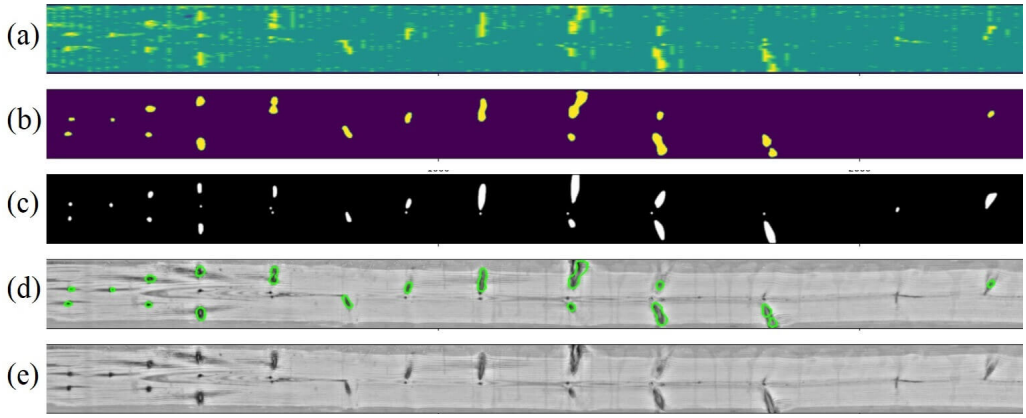

Figure 21. The learned features mostly match true knot locations (precision: 0.97, recall: 0.76, F1-score: 0.85, IoU: 0.36): (a) Computed heatmap; (b) Filtered heatmap; (c) True knot locations; (d) Detected knot locations; (e) Original image.

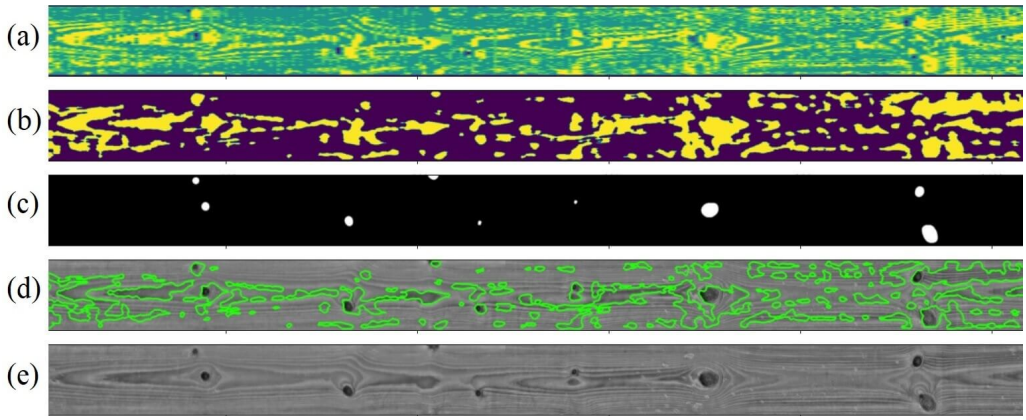

Figure 22. The model learns the timber texture in addition to knots in some cases (precision: 0.09, recall: 1.00, F1-score: 0.16, IoU: 0.04): (a) Computed heatmap; (b) Filtered heatmap; (c) True knot locations; (d) Detected knot locations; (e) Original image.

model learns true knot locations better on low quality boards than on high quality boards. In addition to it, the model learns areas without any knots more often on high quality boards than on low quality boards.

5.5.2 Laser scans of log surface

The examples of heightmaps, corresponding visualizations, detected defects, and possible knot locations are shown in Figs. 23 and 24. According to the visualization, the model tends to learn possible knot locations on the generated heightmaps as important features. In addition to it, the model learns knot clusters instead of specific knots. However, it does not provide any meaningful visual explanation in such cases as shown in Fig. 25.

Table 2. Weakly supervised knot detection performance on the images of boards.

Groups of samples	Samples	Precision	Recall	F1-score	IoU
Grade A (low quality)	136	0.75	0.86	0.77	0.26
Grade B (low quality)	60	0.69	0.87	0.74	0.29
Grade C (low quality)	40	0.57	0.87	0.65	0.25
Grade D (high quality)	160	0.64	0.87	0.69	0.21
Grade E (high quality)	39	0.44	0.91	0.54	0.21
Low quality boards	236	0.71	0.86	0.74	0.26
High quality boards	199	0.59	0.85	0.65	0.21
All boards	435	0.66	0.87	0.70	0.24

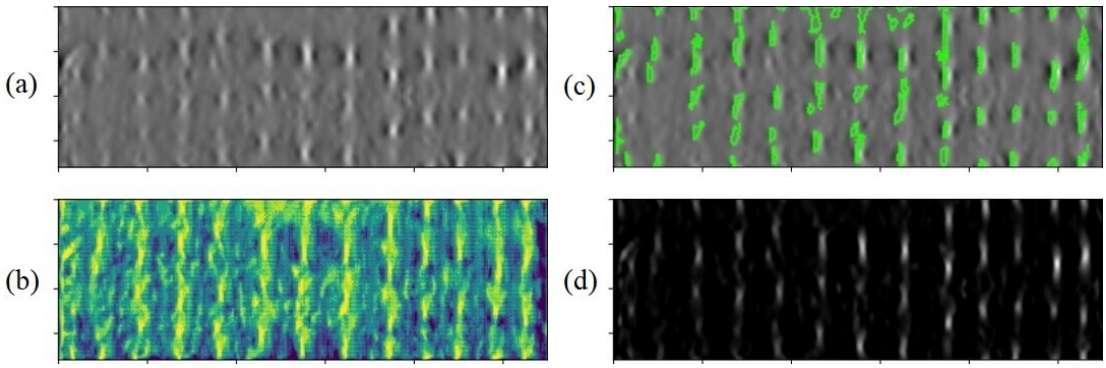

Figure 23. The learned features mostly match possible locations of individual knots: (a) Heightmap; (b) Heatmap; (c) Detected defects; (d) Possible knot locations.

As stated before, true knot locations are unknown on the generated heightmaps at the moment which is why the prepared images of possible knot locations cannot be used to evaluate the proposed method performance and get reliable results. At this point, the method performance for laser scans of log surfaces should be evaluated as part of future work when the ground truth becomes available.

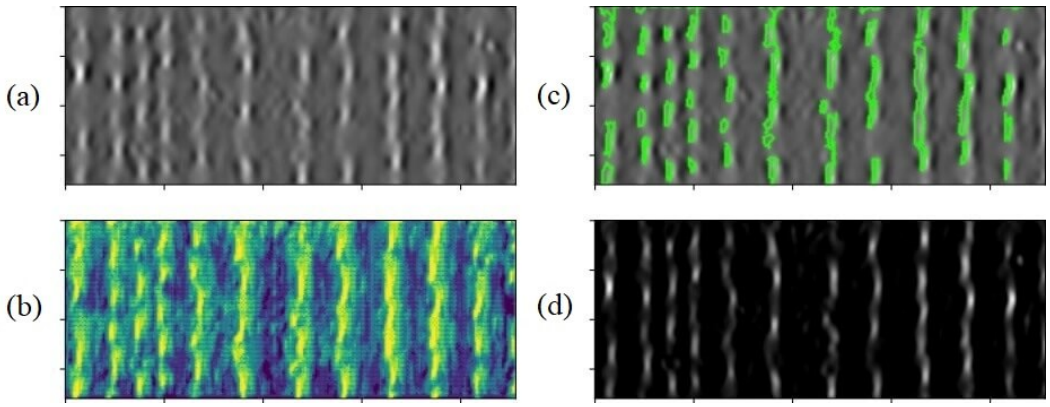

Figure 24. The learned features mostly match possible locations of individual knots and knot clusters: (a) Heightmap; (b) Heatmap; (c) Detected defects; (d) Possible knot locations.

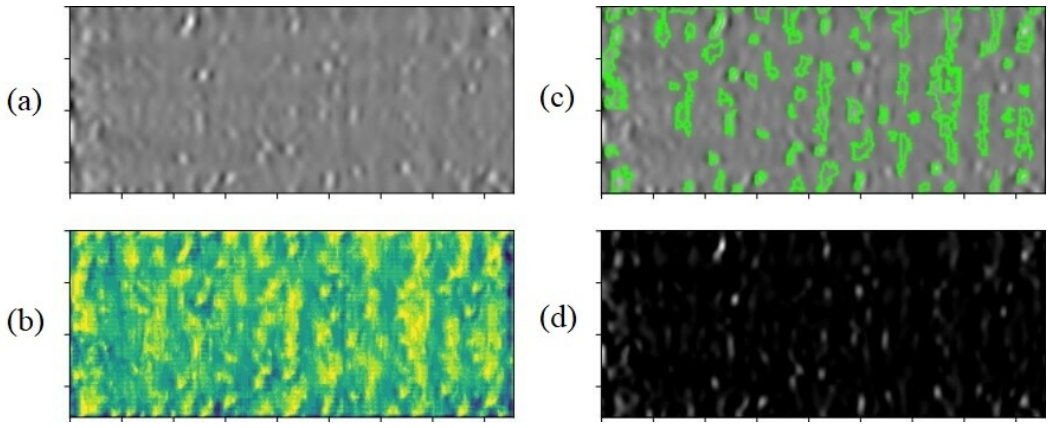

Figure 25. The model does not indicate any concentration on specific knot locations: (a) Heightmap; (b) Heatmap; (c) Detected defects; (d) Possible knot locations.

6 DISCUSSION

6.1 Current study

Timber matching is an important part of the sawmilling process as it provides better product quality prediction, efficient process control, and process optimization for the industry. The recent approach developed to trace timber in a sawmill greatly relies on machine vision. It is based on the architecture of convolutional neural networks and appears to be more effective in terms of accuracy and operational costs than such methods as RFID-tags and marking logs with paint. However, such a solution needed to be verified to become a reliable industry standard.

In this study, the multimodal decoder-encoder network was examined and its decisions in timber matching were verified through visualization. The basic idea of this approach is to reveal what parts of the input image are the most important for the neural network when it makes a specific decision. Nowadays, this approach belongs to the concept of Explainable AI where various methods have been developed recently to explain and interpret machine learning models. In particular, it offers tools such as Grad-CAM to make convolutional neural networks more transparent. With Grad-CAM implemented, this thesis suggests methods for visualization and knot detection where the most important steps are data preprocessing, computation of activations maps, and object detection based on the features learned by the neural network only.

The experimental part of this work describes the results and evaluates the proposed method performance using the true knot locations on the images of boards. According to the results, the weakly supervised knot detection produces satisfactory results and confirms the assumption about the neural network learning knot locations to match boards to logs when applied to the images of boards. For example, the values of precision, recall, and F1-score achieve 94%, 86%, and 90% respectively on a single board image. In this case, most of the found knots are correct. However, it should be noted that the model learns more areas without any defects on high quality boards than on low quality boards. For example, precision is 75%, recall is 86% and F1-score is 77% for one of the low quality groups of boards while precision is 44%, recall is 91% and F1-score is 54% for one of the high quality groups of boards. As a result, the weakly supervised knot detection produces more false positives for high quality boards, but it detects more true positives for low quality boards.

It was also shown that the encoder-decoder network learns areas inside knot clusters on the log heightmaps. However, it is not possible to evaluate the weakly supervised knot detection in this case since there is no information about true knot locations on the heightmaps at the moment.

6.2 Future work

The research has shown promising results and there are still several possibilities for future work. First of all, it would be possible to estimate the method performance and get reliable results for laser scans of log surfaces if more information about true knot locations was obtained as the ground truth. Secondly, the proposed method can be further utilized to improve the accuracy of board-to-log matching. For instance, changes in the detected knot location, shape, and size can indicate whether modifying neural network layers and altering input data have a positive impact on the neural network decisions. Thirdly, more methods such as Score-CAM can be utilized to compute activation maps and compare their performance to Grad-CAM. Finally, further examination of the neural network can include visualization of hidden layers to better understand its internals.

7 CONCLUSION

This study focuses on the problem of interpretability and explainability of multimodal timber matching networks used in a sawmill environment. The multimodal encoder-decoder network developed in previous research outperforms analyzed traditional methods while acting as a black box. As an attempt to explain the neural network decisions, activation maps were computed to show what regions on the input images have an impact on decision making. According to the visualization, the neural network tends to learn true knot locations and true knot clusters as features used for timber matching.

Moreover, this study suggests a weakly supervised method to detect knots on the images of boards using the features learned by the neural network only. According to the experiments, the highest F1-score is about 90% for individual samples and it varies from 54% to 77% for the chosen groups of samples. The model demonstrates better precision, recall, F1-score, and IoU on low quality boards than on high quality boards. Besides, this study suggests a similar method to detect knots on the heightmaps generated with the laser scans of log surfaces. However, currently, it is infeasible to evaluate this method performance since true knot locations are unknown.

Finally, the concepts and methods described in this thesis can be used in future works to improve the accuracy of timber matching and understand the neural network internals better.

REFERENCES

[1] P. D. Dykstra, G. Kuru, R. Taylor, R. Nussbaum, B. W. Magrath, and J Story. *Technologies for Wood Tracking. Verifying and Monitoring the Chain of Custody and Legal Compliance in the Timber Industry*. Discussion paper, Word Bank, Washington, DC, 2002.

[2] R. A. Kozak and T. C. Maness. A system for continuous process improvement in wood products manufacturing. *Holz als Roh- und Werkstoff*, 61(2):95–102, April 2003.

[3] Fedor Zolotarev, Tuomas Eerola, Lasse Lensu, Heikki Kälviäinen, Heikki Haario, Jere Heikkinen, and Tomi Kauppi. Timber tracing with multimodal encoder-decoder networks. In *Computer Analysis of Images and Patterns, Springer Lecture Notes in Computer Science, LNCS Vol. 11679, International Conference on Computer Analysis of Images and Patterns (CAIP)*, pages 342–353. Springer, Springer International Publishing, August 2019.

[4] Alejandro Barredo Arrieta, Natalia Diaz Rodriguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado González, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, V. Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, December 2019.

[5] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *International Conference on Engineering and Technology (ICET)*. IEEE International Conference on Engineering and Technology (ICET), August 2017.

[6] Daniel Omeiza, S. Speakman, C. Cintas, and K. Weldemariam. SmoothGrad-CAM++: An enhanced inference level visualization technique for deep convolutional neural network models. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 111–119, 2020.

[7] DigiSaw - Leap of Digitalisation for Sawmill Industry. http://www2.it.lut.fi/project/digisaw/index.shtml, 2020. [Online; accessed December, 12, 2020].

[8] Guide to sawmills. https://www.yorksaw.com/guide-to-sawmills/, 2021. [Online; accessed January, 25, 2021].

[9] Swedish wood: From log to plank. https://www.swedishwood.com/wood-facts/choosing-wood/from-log-to-plank/, 2011. [Online; accessed January, 25, 2021].

[10] Sawmill debarkers. https://sahateollisuuskirja.fi/en/sahatavaran-valmistus/tukkien-kuorinta/kuorimakoneet/, 2021. [Online; accessed January, 25, 2021].

[11] P. Cavalin, L. S. Oliveira, A. L. Koerich, and A. S. Britto. Wood defect detection using grayscale images and an optimized feature set. In *Institute of Electrical and Electronics Engineers (IEEE) - 32nd Annual Conference*, November 2006.

[12] Swedish wood: Wood grades. https://www.swedishwood.com/wood-facts/about-wood/wood-grades/, 2011. [Online; accessed January, 25, 2021].

[13] Swedish wood: Wood and moisture. https://www.swedishwood.com/wood-facts/about-wood/wood-and-moisture/, 2011. [Online; accessed January, 25, 2021].

[14] Fedor Zolotarev, Tuomas Eerola, Lasse Lensu, Heikki Kälviäinen, Tapio Helin, Heikki Haario, Tomi Kauppi, and Jere Heikkinen. Modelling internal knot distribution using external log features. *Computers and Electronics in Agriculture*, 179:105795, December 2020.

[15] Irene Yu-Hua Gu, Henrik Andersson, and Raul Vicen. Wood defect classification based on image analysis and support vector machines. *Wood Science and Technology*, 44(4):693–704, November 2009.

[16] Sorin Chiorescu and Anders Grönlund. The fingerprint method: Using over-bark and under-bark log measurement data generated by three-dimensional log scanners in combination with radiofrequency identification tags to achieve traceability in the log yard at the sawmill. *Scandinavian Journal of Forest Research*, 19(4):374–383, 2004.

[17] Tobias Pahlberg, Olle Hagman, and Matthew Thurley. Recognition of boards using wood fingerprints based on a fusion of feature detection methods. *Computers and Electronics in Agriculture*, 111:164–173, 2015.

[18] Asif Rahman, Siril Yella, and Mark Dougherty. Simulation and optimization techniques for sawmill yard operations—a literature review. *Journal of Intelligent Learning Systems and Applications*, 06(01):21–34, 2014.

[19] R. Hietaniemi, S. Varjo, and J. Hannuksela. A machine vision based lumber tracing system. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 98–103, 2013.

[20] Ting He, Y. Liu, C. Xu, Xiao lin Zhou, Zhongkang Hu, and J. Fan. A fully convolutional neural network for wood defect location and identification. *IEEE Access*, 7:123453–123462, 2019.

[21] E. Johansson, T. Pahlberg, and O Hagman. Fast visual recognition of scots pine boards using template matching. *Computers and Electronics in Agriculture*, 118:85–91, 2015.

[22] L. Thomas and R.E Thomas. A graphical automated detection system to locate hardwood log surface defects using high-resolution three-dimensional laser scan data. In *17th Central Hardwood Forest Conference*, 2010.

[23] Nikolay Rudakov, Tuomas Eerola, Lasse Lensu, Heikki Kälviäinen, and Heikki Haario. Detection of mechanical damages in sawn timber using convolutional neural networks. In *Pattern Recognition, Springer Lecture Notes in Computer Science, LNCS Vol. 11269, German Conference on Pattern Recognition (GCPR)*, pages 115–126. Springer International Publishing, 2019.

[24] Gonzalo A. Ruz, Pablo A. Estévez, and Pablo A. Ramírez. Automated visual inspection system for wood defect classification using computational intelligence techniques. *International Journal of Systems Science*, 40(2):163–172, February 2009.

[25] Yizhuo Zhang, Chao Xu, Chao Li, Huiling Yu, and Jun Cao. Wood defect detection method with PCA feature fusion and compressed sensing. *Journal of Forestry Research*, 26(3):745–751, April 2015.

[26] Schad K.C., Schmoldt D.L., and Ross R.J. Nondestructive methods for detecting defects in softwood logs. *U.S. Department of Agriculture Research Paper. Forest Service, Forest Products Laboratory*, 1996.

[27] F. Longuetaud, F. Mothe, B. Kerautret, A. Krähenbühl, L. Hory, J.M. Leban, and I. Debled-Rennesson. Automatic knot detection and measurements from x-ray CT images of wood: A review and validation of an improved algorithm on softwood samples. *Computers and Electronics in Agriculture*, 85:77–89, July 2012.

[28] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The Handbook of Brain Theory and Neural Networks*, 3361(10), 1995.

[29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[30] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, January 2012.

[32] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, November 2013.

[33] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440. IEEE, 2015.

[34] Convolutional neural network tutorial. https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network, 2021. [Online; accessed May, 25, 2021].

[35] Introduction to convolutional neural networks. https://dev.to/pmgysel/introduction-to-convolutional-neural-networks-4358, 2020. [Online; accessed January, 25, 2021].

[36] Convolution, padding, stride, and pooling in cnn. https://medium.com/analytics-vidhya/convolution-padding-stride-and-pooling-in-cnn-13dc1f3ada26, June 2020. [Online; accessed January, 31, 2021].

[37] Training a convolutional neural network. https://victorzhou.com/blog/intro-to-cnns-part-2/, 2021. [Online; accessed May, 15, 2021].

[38] Understanding different loss functions for neural networks. https://towardsdatascience.com/understanding-different-loss-functions-for-neural-networks-dd1ed0274718, 2019. [Online; accessed May, 28, 2021].

[39] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, December 2017.

[40] For AI, translation is about more than language. http://cachestocaches.com/2018/9/ai-translation-more-language/, 2018. [Online; accessed January, 25, 2021].

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2015.

[42] Jindong Gu, Yinchong Yang, and Volker Tresp. Understanding individual decisions of CNNs via contrastive backpropagation. In *Asian Conference on Computer Vision (ACCV)*, pages 119–134. Springer International Publishing, 2019.

[43] Luca Longo, Randy Goebel, Freddy Lecue, Peter Kieseberg, and Andreas Holzinger. Explainable artificial intelligence: Concepts, applications, research challenges and visions. 2020.

[44] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, October 2019.

[45] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *Winter Conference on Applications of Computer Vision (WACV)*. IEEE, March 2018.

[46] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2020.

[47] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2016.

[48] Towards data science: Visual interpretability for convolutional neural networks. https://towardsdatascience.com/visual-interpretability-for-convolutional-neural-networks-2453856210ce, 2018. [Online; accessed January, 25, 2021].

[49] Choosing colormaps in matplotlib. https://matplotlib.org/3.1.0/tutorials/colors/colormaps.html, 2021. [Online; accessed April, 1, 2021].

[50] Evaluating deep learning models: The confusion matrix, accuracy, precision, and recall. https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/, 2021. [Online; accessed May, 1, 2021].