Dissimilarity to class medoids as features for 3D point cloud classification

Sylvain Chabanet, Valentin Chazelle, Philippe Thomas, Hind Bril El-Haouzi

Dissimilarity to class medoids as features for 3D point cloud classification*

Sylvain Chabanet[1][0000−0002−3706−293X], Valentin Chazelle, Philippe Thomas[1][0000−0001−9426−3570], and Hind Bril El-Haouzi[1][0000−0003−4746−5342]

Université de Lorraine, CNRS, CRAN, F-88000 Epinal, France {sylvain.chabanet, hind.el-haouzi,valentin.chazelle, philippe.thomas}@univ-lorraine.fr

Abstract. Several sawmill simulators exist in the forest-product industry. They are able to simulate the sawing of a log to generate the set of lumbers that would be obtained by transforming a log at a sawmill. In particular, such simulators are able to use a 3D scan of the exterior shape of the logs as input for the simulation. However, it was observed that they can be computationally intensive. Therefore, several authors have proposed to use Artificial Intelligence metamodel, which, in general, can make predictions extremely fast once trained. Such models can approximate the results of a simulator using a vector of descriptive features representing a log, or, alternatively, the full 3D log scans. This paper proposes to use dissimilarity to representative log scans as features to train a Machine Learning classifier. The concept of class Medoids as representative elements of a class will be presented, and a Simlarity Discrimant Analysis was chosen as a good candidate ML classier. This classifier will be compared with two others models studied by the authors.

Keywords: Sawmill simulation · Artificial Intelligence · Iterative Closest Point dissimilarity · Medoids · Similarity Discriminant Analysis

1 Introduction

The integration of 3D point cloud based tools in decision support systems has been gaining attention in the past decades with the development of reliable acquisition devices like terrestrial or airborne Lidar. For example, [18] reviews different usages of 3D point cloud processing in the construction industry from 2004 to 2018, from construction progress tracking to 3D model reconstruction. Similarly, [13] proposes a method to automatically model complex industrial installations from 3D scan scenes by segmenting and comparing individual elements with a model library. [16] proposes an application to industry 4.0 with the creation of facility digital twins. The point clouds would be acquired from

* The authors gratefully acknowledge the financial support of the ANR-20-THIA-0010-01 Projet LOR-AI (lorraine intellgence artificielle) and région Grand EST. We are also extremely grateful to FPInnovation who gathered and processed the dataset we are working with.

mobile phones and processed on remote servers to be transformed into multiple 2D views of the scene and fed to Artificial intelligence (AI) classification models.

Diverse usages of 3D scans similarly exists in the forest-wood industry and in the related literature. For example, [12] proposes the use of 3D scans generated by a terrestrial Lidar to detect and classify defects on standing tree surfaces. Similarly, scans of wood logs have been used for a long time in the sawmilling industry, which has several simulators as its disposal to process these point clouds. The objective of these simulators is to simulate the sawing of the logs in a non destructive way and generate production data. For example, [11] proposes to use such scans and simulators to optimize the allocation of logs between several harvest sites and sawmills, using the simulated basket of products which would be obtained by transforming sampled logs at each possible sawmill. Since introducing new machinery to sawmills might require heavy investment, it additionally appears of particular interest to provide solutions using existing or low cost laser scanners.

Considering, however, that traditional point cloud processing methods can be extremely time consuming [13], AI based methods have been gaining attention in the literature. For example, multiple works have been published in the past few years proposing new deep learning models trained on huge CAD databases. The interested reader may refer to [6] for a survey on the field. The use of these neural networks to predict log basket of products has been studied recently by [8], with interesting results.

In this paper, we propose to rather use a variant of Naive Bayes classifiers which uses dissimilarity to class medoids as input features. This model is named Similarity Discriminant Analysis (SDA) [3] in the literature. Such a model has, indeed, the advantage of being trainable with relatively few data. Furthermore, training such a classifier doesn't require the extraction of knowledge based features and is, therefore, completely data driven. This methodology was tested on a dataset from the Canadian sawmilling industry.

This paper is structured as follow. Section 2 reviews previous works about the use of ML simulation metamodels in the wood industry. The SDA classifier is briefly explained in section 3. Section 4 presents the experimental setup and simulation results. Section 5 will conclude and gives some perspectives.

2 Previous works on sawmill simulation metamodeling

Breaking a log into lumbers is a divergent process with co-production. Several different products are, indeed, simultaneously obtained from the sawing of one log. Additionally, these lumbers can have various dimensions and grades. For this reason, this process is sometime compared with a disassembly process. Due to the fact that this sawing process can be automatized, with online optimizers, and that logs are heterogeneous in terms of shape and quality, it is difficult to predict in advance the lumber output of a log. The sawmill industry, however, has numeric simulators at its disposal to compute the set of lumbers, called in this paper basket of products, which would be obtained by processing a specific

log at a modeled sawmill. Such simulators can use 3D point clouds obtained by laser scanners. Examples are Optitek [5], Autosaw [17] or Sawsim [7], which, in particular, proposes the assessment of multiple sawmill designs as an example of typical use. Similarly, [19] proposes the use of a sawmill digital simulation to optimize a tactical production plan taking into consideration the acceptation of orders with unusual products. Indeed, to respond positively to such an order would have an impact on the whole lumber mix produced at the sawmill.

These simulators can, however, be computationally intensive. For example, depending on the simulation setting and log scan, computing the resulting basket of products for one log can take from a few seconds to 3 hours and more using Optitek. Considering that fact, [10] proposed to approximate these simulators with AI metamodels. In particular, several Machine Learning (ML) classifiers are trained on results from past simulations. These models include k Nearest Neighbors (kNN) and Random Forest. The input features used as input of these classifiers are know-how features describing each log, like, for example, their length and volume. A further work, [9], considers the problem of logs allocation to sawmills. By using machine learning metamodels, this study demonstrates that it is possible to increase significantly the value of the objective function being optimized. While this objective function doesn't represent the actual benefit obtained at a real sawmill, their numeric experiments are promising. Considering, however, that contrary to sawmill simulators those classifiers only used six features describing the logs, [15] proposed to use a kNN based on a point cloud dissimilarity. A drawback of this method is that computing this dissimilarity involves the Iterative Closest Point (ICP) algorithm [2], which can be relatively computationally intensive, especially since multiple ICP are needed by the kNN to yield each prediction. Each new log has, indeed, to be compared with all known logs in an example database. [4] later proposed to reduce the number of ICP needed to yield a prediction by implementing a set of rules to filter out unnecessary comparisons. While this approach reduce in average by more than half the number of ICP comparisons needed for a prediction, several hundreds are still required. In this paper, another approach is considered, which is to use a dissimilarity as feature scheme [14]: a few representative logs are selected, and a new log is represented by its vector of dissimilarity to these features.

3 Similarity discriminant analysis

Following is a description of the SDA, a naive bayes classifier using similarity to medoids as features.

3.1 Medoids

As presented in [14], the use of an euclidean metric is central to numerous standard ML algorithms. Several methods have been proposed to adapt them when the data points are unstructured and cannot be easily considered member of a metric space, but are, instead, only known by comparison among themselves

using a non metric similarity (or dissimilarity) measure. One of these methods considers the use of dissimilarity toward representative elements of the dataset as features. These features can then be used like any standard vector representation. [1] shows that under some theoretical conditions on the similarity function, a binary classifier with bounded error can be learned from a representation of data points as a vector of similarities to a subset of other randomly sampled points. Similarly, [3] proposed to use medoids as representative points.

Class medoids are, indeed, a natural choice for class representatives when the data points composing said class can't be easily averaged to form a class mean. Considers $x_1, x_2, ..., x_n$ the n elements of a class in a database, i.e, in this paper, the n logs sharing a particular basket of products. The class medoid μ is a central element of the class, i.e:

$$\mu = \underset{j \in \{1,2,...,n\}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} d(x_j, x_i)\,, \tag{1}$$

with d the ICP dissimilarity. The medoid is, therefore, the member of the class with minimal average dissimilarity with all the other class members. Contrary to the class mean, it is a real member of the database.

In this paper, the medoids of each p classes, $\mu_1, ..., \mu_p$ are used to represent log scans in the following way. Let x be a new log scan. It is represented by the p dimensional vector $(d(\mu_1, x), d(\mu_2, x), ..., d(\mu_n, x))$, i.e, the vector of dissimilarity between x and each medoid.

The dataset used in this paper contains numerous classes. Some of them are extremely rare, to the point of appearing only once or twice in the whole dataset. therefore, a medoid is considered only if the class it belongs to appears more than once in the dataset used for training the ML classifier. SDA parameters need, indeed, strictly more than one sample per class to be estimated. Therefore, when a class appears too rarely in the dataset, it isn't taken into consideration by the classifier. This also implies, however, that such a class can never be predicted correctly.

3.2 Similarity Discriminant Analysis

SDA is a generative classifier introduced by [3]. It is specifically tailored for cases where data inputs are only known from similarities or dissimilarities comparison among themselves. More particularly, considering a vector $T^x = (t_1^x, ..., t_p^x)$ of dissimilarities to the class medoids, the model aims at modeling the probability $\mathbf{P}(Y = j|T^x)$ for all possible classes $j \in \{1, ..., p\}$. The class predicted for x is then the class which minimize the expected misclassification cost, i.e:

$$\hat{y} = \underset{j \in (1,...,p)}{\arg\min}\, \mathbf{E}_{\mathbf{P}(Y|T^x)}(\mathrm{Cost}(j, Y)), \tag{2}$$

where $\mathrm{Cost}(j, Y))$ represents the cost of predicting the class j for x, while its real class is Y.

[3] authors give an elegant argument based on Bayes theorem and entropy maximization to justify their proposed estimator for this quantity, leading to the following formula for the prediction $\hat{y}$:

$$\hat{y} = \underset{i\in(1,\dots,p)}{\arg\min} \sum_{j\in(1,\dots,p)} \text{Cost}(i,j)\left(\prod_{k=1}^{p} \lambda_{jk} e^{-\lambda_{jk} t_k^x}\right)\mathbf{P}(Y=j)\,, \tag{3}$$

with $\frac{1}{\lambda_{jk}} = \frac{1}{|J|}\sum_{x \text{ with label } j} t_k^x$, $|J|$ the number of elements with label j in the training database. $\frac{1}{\lambda_{jk}}$ is, therefore, the average dissimilarity from the class j to the medoid μ_k. The probabilities $\mathbf{P}(Y=j)$ are, generally, unknown. They are, therefore, inferred from the training set. The Cost function used in this paper is $1 - s^{pre\times pro}$, with $s^{pre\times pro}$ defined in section 4.

This model was implemented using the programming language Python.

4 Experiment

This section present our experiments on a logs 3D scan database. The results from the SDA models are compared with the results from two other models, previously studied in [4].

4.1 The log database

The database used in this paper was provided by the Canadian wood industry. It contains the scans of 1207 logs, and their associated baskets of products, computed by the sawing simulator Optitek. Each scan is a point cloud composed of a succession of ellipsoid which, together, sample the log surface. The original scans had empty sections, i.e missing ellipsoids, leading to poor performances of the ICP algorithm when computing scan dissimilarities. This behavior had been corrected by repeating the ellipsoid immediately preceding an empty section to fill it. The database contains 19 types of lumbers. The basket of products y associated with a log x can, therefore, be represented by a vector of length 19. The i^{th} element of this vector is then the number of lumbers of type i present in x basket of products. It might be noticed that no basket contains more than five different types of lumbers and that, therefore, the vectors y are sparse.

The database contains in total 105 different baskets, each being considered a class in our classification problem and is represented by a number from 1 to 105.

4.2 Evaluation scores

When training ML classifiers, scores have to be introduced to measure and compare their performances. The most commonly used score is, probably, the 0-1 score, s^{01}, defined as:

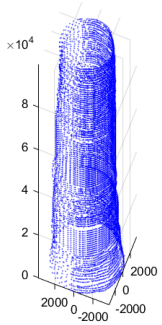

Fig. 1. Example of a log 3D scan

$$s^{01}(y, \hat{y}) = \begin{cases} 1, & \text{if } y = \hat{y} \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

with y the true label of a data point x, and $\hat{y}$ its predicted label. Such a score is then averaged over all the data points in a test dataset to estimate the probability for the classifier to predict the real class of any point x.

However, it might appear desirable for the cost of making a false prediction to vary depending on the true class label y and false prediction $\hat{y}$. The prediction-production score, $s^{pre\times pro}$, was specifically introduced in [10] for the problem of sawmill simulator metamodelling. Let y and $\hat{y}$ be once again the real and predicted baskets of products associated with a log x. Since these vectors are sparse, counting all the $(0, 0)$ real/predicted pairs contained in these vectors would skew the scores optimistically, all such pairs are, therefore, removed. The new length of y and $\hat{y}$ is called l. the prediction score, s^{pre}, is defined as:

$$s^{pre} = \frac{1}{l} \sum_{i=1}^{l} \min(1, \frac{\hat{y}_i}{\max(\epsilon, y_i)}), \tag{5}$$

with ϵ a small value to avoid dividing by 0. Similarly, the prediction score, s^{pro}, is defined as:

$$s^{pro} = \frac{1}{l} \sum_{i=1}^{l} \min(1, \frac{y_i}{\max(\epsilon, \hat{y}_i)}). \tag{6}$$

s^{pre} can be interpreted as the proportion of real basket that was predicted, while s^{pro} is the proportion of the prediction that was effectively produced.

The prediction-production score is then naturally defined as:

$$s^{pre\times pro} = s^{pre} \times s^{pro}. \tag{7}$$

4.3 Results and discussion

For training and testing the SDA, the log database was randomly separated 10 times into a train test of size 724, i.e, 60% of the database, and a test set of size 483, i.e, 40% of the database. The results from the SDA are, furthermore, compared with two other models from [4]. the first one is a classic k nearest neighbors algorithm. The second is a kNN which uses a set of rules to filter out unnecessary ICP comparisons. This model is named r-kNN in this section. The rules used to filter these comparisons are :

- If a log is shorter than the length of the smallest possible lumber, it is attributed an empty basket of product without performing any comparison.
- Since logs come in a few standard length, two logs are compared only if they have the same length.

For both of these models, k was fixed to 25 as in [4].

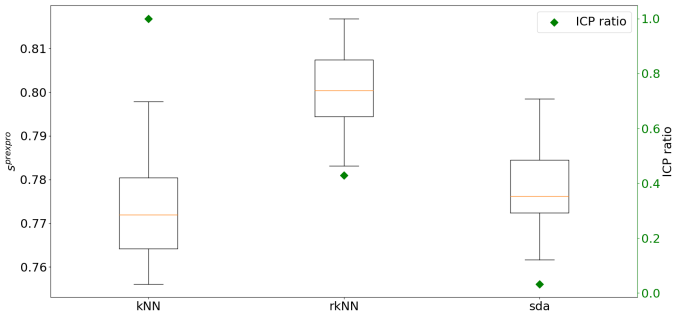

Fig. 2. Boxplots of the average prediction-production scores over the 10 train-test separations for each model. The averaged ratio of the number of ICP needed to yield a prediction over the total size of the train set are presented as well. The scores are in percent

The boxplots of the $s^{pre \times pro}$ scores of each model over the ten train-test are presented figure 2. As can be seen, r-kNN has the highest $s^{pre \times pro}$ scores among the three models, while kNN and SDA are comparable. What strongly counterbalance this lesser prediction performance for SDA, compared with r-kNN, is, however, an important reduction of the number of ICP comparisons needed to yield a prediction. While kNN needs to compare the new log with the whole training database and r-kNN with, in average, 40% of the database, SDA needs only to compare it with 3% of the database, i.e, only with the medoids. Additionally, while the number of comparisons needed would increase linearly in the size of the training database for r-kNN, it would remain constant for SDA as long as no new class appears in the dataset.

The average values of the compared models evaluation scores are further presented in table 1. These evaluation scores are mean s^{pre}, mean s^{pro}, mean $s^{pre \times pro}$ and mean s^{01}, averaged over the ten random separations in train and test sets. The highest scores among both models are set in bold. As previously, r-kNN has slightly higher evaluations scores than the SDA model, except for the production score. However, when comparing the 0-1 scores of r-kNN and SDA using a Mcnemar test, the minimum pvalue of the test among the ten experiments was 6 %. The error rate difference among the models can, therefore, never be considered significant at a 5% confidence level in any of our experiments. The difference in score is, indeed, only around 2% in average for both s^{01} and $s^{pre \times pro}$. To deem this difference acceptable or not would depend on the actual industrial application considered.

Model	s^{pre}	s^{pro}	$s^{pre \times pro}$	s^{01}	ICP ratio
kNN	89.2 ± 0.5	85.5 ± 0.9	77.5 ± 0.8	66.5 ± 0.1	100
r-kNN	$\mathbf{89.3 \pm 0.5}$	88.2 ± 0.6	$\mathbf{80.1 \pm 0.7}$	$\mathbf{69.0 \pm 0.9}$	42 ± 1
SDA	85.2 ± 0.7	$\mathbf{89.8 \pm 0.5}$	77.8 ± 0.7	67.2 ± 0.1	$\mathbf{3.2 \pm 0.2}$

Table 1. Evaluation scores of SDA and r-kNN, averaged over ten random separations of the database into a train and a test set, as well as the averaged ratio of the number of ICP needed to yield a prediction over the total size of the train set. The scores are similarly presented in percent.

5 Conclusion

This paper explores the use of medoids as features to train a well understood ML classifier to the task of predicting the baskets of products of 3D log scans. In particular, this method improves on previous works using kNN classifier by reducing drastically the number of ICP comparisons needed to compare a new log with a known database from 40% to 3% of the size of the training set, with a limited reduction in score.

A second advantage of this method is that it doesn't require complex feature extraction and, therefore, it would be of interest to generalize it to other point cloud applications.

Furthermore, while the SDA was specifically introduced for case with non metric similarities, the representation as a vector of dissimilarities to medoids could be used with other off the shelf ML classifiers. In particular, further works will consider using this representation with Random Forest classifiers and Multi Layer Perceptrons.

References

1. Balcan, M.F., Blum, A., Srebro, N.: Improved guarantees for learning via similarity functions (2008)
2. Besl, P.J., McKay, N.D.: A method for registration of 3-d shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **14**(2), 239–256 (1992)
3. Cazzanti, L.: Similarity Discriminant Analysis (01 2009)
4. Chabanet, S., Thomas, P., Bril El-Haouzi, H., Morin, M., Gaudreault, J.: A knn approach based on icp metrics for3d scans matching: an application to thesawing process. In: 17th IFAC Symposium on Information Control Problems in Manufacturing (2021)
5. Goulet, P.: Optitek: User's manual (2006)
6. Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M.: Deep learning for 3d point clouds: A survey. IEEE transactions on pattern analysis and machine intelligence (2020)
7. HALCO software systems ltd. : The sawsim sawmill simulation tool. https://www.halcosoftware.com/software-1-sawsim, last accessed on June, 2021
8. Martineau, V., Morin, M., Gaudreault, J., Thomas, P., Bril El-Haouzi, H.: Neural network architectures and feature extraction for lumber production prediction. In: The 34th Canadian Conference on Artificial Intelligence. Springer (2021)
9. Morin, M., Gaudreault, J., Brotherton, E., Paradis, F., Rolland, A., Wery, J., Laviolette, F.: Machine learning-based models of sawmills for better wood allocation planning. International Journal of Production Economics **222**, 107508 (2020)
10. Morin, M., Paradis, F., Rolland, A., Wery, J., Laviolette, F., Laviolette, F.: Machine learning-based metamodels for sawing simulation. In: 2015 Winter Simulation Conference (WSC). pp. 2160–2171. IEEE (2015)
11. Morneau-Pereira, M., Arabi, M., Gaudreault, J., Nourelfath, M., Ouhimmou, M.: An optimization and simulation framework for integrated tactical planning of wood harvesting operations, wood allocation and lumber production. In: MOSIM 2014, 10eme Conférence Francophone de Modélisation, Optimisation et Simulation (2014)
12. Nguyen, V.T., et al.: Estimation de la qualité de bois ronds et d'arbres sur pied par Lidar terrestre. Ph.D. thesis, Paris, AgroParisTech (2018)
13. Pang, G., Qiu, R., Huang, J., You, S., Neumann, U.: Automatic 3d industrial point cloud modeling and recognition. In: 2015 14th IAPR international conference on machine vision applications (MVA). pp. 22–25. IEEE (2015)
14. Schleif, F.M., Tino, P.: Indefinite proximity learning: A review. Neural Computation **27**(10), 2039–2096 (2015)
15. Selma, C., El Haouzi, H.B., Thomas, P., Gaudreault, J., Morin, M.: An iterative closest point method for measuring the level of similarity of 3d log scans in wood industry. In: Service Orientation in Holonic and Multi-Agent Manufacturing, pp. 433–444. Springer (2018)
16. Stojanovic, V., Trapp, M., Richter, R., Hagedorn, B., Döllner, J.: Towards the generation of digital twins for facility management based on 3d point clouds. In: Proceeding of the 34th Annual ARCOM Conference. vol. 2018, pp. 270–279 (2018)
17. Todoroki, C., et al.: Autosaw system for sawing simulation. New Zealand Journal of Forestry Science **20**(3), 332–348 (1990)
18. Wang, Q., Kim, M.K.: Applications of 3d point cloud data in the construction industry: A fifteen-year review from 2004 to 2018. Advanced Engineering Informatics **39**, 306–319 (2019)

19. Wery, J., Gaudreault, J., Thomas, A., Marier, P.: Simulation-optimisation based framework for sales and operations planning taking into account new products opportunities in a co-production context. Computers in industry **94**, 41–51 (2018)