Erasmus University Rotterdam

Erasmus School of Economics

Master thesis Econometrics and Management Science

Selection of sustainable packaging materials using a Bayesian Neural Network and a Multi-Objective Genetic Algorithm

Author:

Anouk Marissa Montfoort

(565509)

Supervisors:

dr. A. Naghi

B. van Poederooijen

Second assessor:

E. O'Neill

Date thesis version 2: July 27, 2022

Abstract:

Unilever's Sustainable Living Plan (Unilever, 2021) proposes several sustainability goals regarding packaging. The design of sustainable packaging already starts at the material selection stage. Material selection is a global concern because of raw material depletion. It is crucial to find which packaging characteristics can accurately represent a sustainability measure such that Unilever gets insights into how to improve their packaging portfolio. This research extends the paper of Zhou et al. (2009) by using Unilever data, a Neural Network (NN) and a Bayesian Neural Network (BNN) in combination with two evolutionary algorithms, NSGA-II and NSGA-III. Eventually, this study finds that BNN improves the prediction error compared to NN in exchange for computational time, while NSGA-III does not improve NSGA-II. Based on the multi-objective optimization with respect to all materials (aluminium, glass, HDPE, LDPE, paper, virgin PET, virgin PP, recycled PET, recycled PP and steel) using BNN in combination with NSGA-II, the most optimal environmental footprint and packaging costs equal 0.0026 points and 0.96 euros per gram of packaging. In this case aluminium should be prioritized the most and LDPE should be prioritized the least. If only plastics (HDPE, LDPE, virgin PET, virgin PP, recycled PET and recycled PP) are included, then the optimized values equal 0.0011 points and 0.82 euros per gram of packaging. Then, RPET should be prioritized the most.

Keywords: **Machine Learning**, **NN**, **BNN**, **MOGA**, **NSGA-II**, **NSGA-III**, **Packaging**, **Unilever**.

Contents

1 Introduction

Unilever's Sustainable Living Plan (Unilever, 2021) proposes goals to tackle environmental degradation. This research focuses on the goals regarding packaging. Unilever strives to be the first major consumer goods company with an absolute plastic reduction across its packaging portfolio, it has the ambition to halve the use of virgin plastic, which is newly non-recycled plastic made of raw materials, and to use at least 25% of post-consumer recycled plastic by 2025 (Unilever, 2020).

This research bases on the simultaneous multi-objective optimization process of Zhou, Yin and Hu (2009), where the environmental impact is modelled using a Life Cycle Assessment (LCA) model.[1] The design of sustainable packaging starts at the material selection stage. Material selection is a global concern because of raw material depletion. It is crucial to find packaging characteristics that can accurately represent a sustainability measure such that Unilever gets insights into how to improve their packaging portfolio and to accomplish their packaging manifesto. Using a multi-objective optimization, it is possible to incorporate multiple goals in the packaging design procedure. Since multiple objectives are often conflicting, relations and trade-offs between objectives become clear. To find the most optimal packaging characteristics, Zhou et al. (2009) makes use of a Artificial Neural Network, i.e. a Neural Network (NN), in combination with a Multi-Objective Genetic Algorithm (MOGA) introduced by Murata and Ishibuchi (1995). This study extends the paper of Zhou et al. (2009) by using NN as well as a Bayesian Neural Network (BNN) in combination with an extended version of the general MOGA.

A Genetic Algorithm (GA), which is part of evolutionary algorithms, is a search algorithm based on natural selection by Charles Darwin's survival of the fittest and is introduced by Holland in 1975 (Konak, Coit, & Smith, 2006; Holland, 1992). Genetically strong species are more likely to pass genes to future generations via reproduction. GA imitates this reproduction process by using a so-called fitness score. Nowadays GAs are mostly used in combination with multi-objective optimization problems. In 1995 Murata and Ishibuchi (1995) introduced a MOGA that searches for a Pareto optimal solution of multi-objective optimization problems based on fitness scores. Instead of using MOGAs, a more simple solution to solve the multi-objective optimization is to convert it to a single-objective optimization by introducing weights, i.e. the objective becomes a weighted objective of the multiple objectives. Zhang and Cui (2019) show that in this case results are unstable and it is difficult to find the optimal solution. MOGAs are able to solve multi-objective optimization problems well, since each individual in the population corresponds to a single solution, it searches over different regions of the solution space and it does not require to prioritize, scale or weight objective functions (Konak et al., 2006). To select packaging materials that minimize environmental degradation and costs the most, this research uses two versions of the Non-dominated Sorting Genetic Algorithm (NSGA), namely NSGA-II, and its extended version NSGA-III (Deb, Pratap, Agarwal & Meyarivan, 2002 ; Konak et al., 2006; Deb & Jain, 2013*a*; Deb & Jain, 2013*b*). Based on performance and speed of convergence, NSGA-II is one of the

[1] Details regarding the abbrevations and symbols can be found in Appendix A.

best algorithms to deal with multi-objective optimization (Afshari & Tesfamariam, 2019). NSGA-III was proposed to improve the performance regarding multiple objective problems because NSGA-III selects diverse and well-distributed nondominated solutions (Yannibelli, Pacini, Monge, Mateos & Rodriguez, 2020). Ishibuchi, Imada, Setoguchi & Nojima (2016) state that NSGA-III does not always outperform NSGA-II but that it depends on the number of objectives and the optimization problem itself. However, Ciro, Dugardin, Yalaoui and Kelly (2016) state that for small multi-objective problems both algorithms perform similarly, while for large multi-objective problems NSGA-III performs better.

The first approach of developing a NN was in 1943, when neuro-physiologist McCulloch and mathematician Pitts modelled a NN using electrical circuits (Foote, 2021). The ideas of Pitts and McCulloch were extended by psychologist Hebb that noticed that firing an impulse multiple times changes the strength of that impulse (Foote, 2021). A NN mimics the human brain; similarly to the human brain a NN consists of neurons connected through edges. A human brain learns by altering the strength of the synaptic connection between neurons when it is exposed to the same impulse for multiple times. A NN works in a similar manner and learns the training data by adjusting weights between neurons until the output of the NN approximates the desired target value (Soleimani, Shoushtari, Mirza & Salahi, 2013). The NN learns the relationship between the inputs and the outputs and its performance depends on the generalization ability; it should be able to accurately predict the output of the unseen test data (Somkuwar, Khaira & Somkuwar, 2010). If the NN fits the training data set too well and is not able to explain the test data, overfitting occurs (Nielsen, 2015). NNs are useful in this research because NNs analyse complex relations while detailed information on the structure is not required (Penm, Chaar & Moles, 2013). NNs are flexible and able to solve non-linear problems that cannot be solved by classical mathematical modeling methods (Azari, Garshasbi, Amini, Rashed-Ali & Mohammadi, 2016). The universal approximation theorem of Hornik, Stinchcombe and White (1989) states that NNs with at least one hidden layer are able to approximate any continuous function to any desired degree of precision.[2] Although, compared to classical mathematical models deep learning models are less interpretable: it is a black-box model and therefore difficult to explain why the model obtains certain relations between inputs and outputs.

Not only Zhou et al. (2009) but in general literature with respect to MOGAs focuses on NN instead of BNN. A BNN is a stochastic artificial NN trained using Bayesian inference (Jospin, Buntine, Boussaid, Laga & Bennamoun, 2020). BNN trains a robust model because it finds a distribution of weights instead of a single point estimate, implying that uncertainty of estimated weights is included in the model. Thus, the advantage of using BNN over NN is that due to the probability distribution of weights, uncertainty regarding predictions can be expressed. A problem with standard NNs is the uncertainty of predictions (Wimarshana, Ryu & Choi, 2014). Confidence in predictions in data-rich spaces is higher than in non-data-rich spaces; standard NNs cannot express these confidence limits.

[2]The proof of the universal approximation theorem can be found in Hornik et al. (1989).

Jospin et al. (2020) mentions that for BNN uncertainty is in line with the observed errors, so that there is less often over- or underconfidence. Moreover, BNNs distinguish between epistemic and aleatoric uncertainty, such that they are highly efficient and can learn from small datasets without overfitting (Jospin et al., 2020).[3] Environmental and packaging data is scarce and standard NNs could overfit, while BNNs correctly estimate the parameters regarding the available data with often high uncertainty if data is scarce (Qinghui, Creager, Duvenaud & Bettencourt, n.d.). Also, Bayesian deep learning methods give well-calibrated predictions on out-of-distribution data (Izmailov, Vikram, Hoffman & Wilson, 2021). Although, BNNs receive criticism on choosing the prior a priori, Jospin et al. (2020) states that priors are soft constraints and comparable to regularization and data augmentation. The disadvantage of using BNN over NN is that in general BNNs are more complex than NNs and more training epochs are needed for convergence (Jospin et al., 2020).

This study has three research objectives that are in line with the aspirations of Unilever:

1. The overall goal of this paper is to get a clear methodology to benchmark packaging of Unilever's products with respect to sustainability and costs, so that Unilever gets clear insights and a road map towards their packaging manifesto.

2. To select packaging materials that minimize environmental degradation and costs the most, a general prediction model based on NN and BNN with a Back-Propagation (BP) algorithm is established. More specifically, the environmental footprint is predicted based on the material fractions in a package, emissions with respect to a package and recyclability rate of a package. In this study material fractions of aluminium, glass, HDPE (High Density Polyetheen), LDPE (Low Density Polyetheen), paper, virgin PET (Polyethyleentereftalaat), virgin PP (Polypropyleen), recycled PET (RPET), recycled PP (RPP) and steel are considered. The method with respect to the networks is emphasized in Sections 3.3.1 and 3.3.2.

3. Based on Unilever's packaging manifesto and savings ambition, NSGA-II and NSGA-III algorithms minimize the environmental footprint predicted by the network and the life-cycle packaging costs with respect to above mentioned ten materials. The optimization framework is demonstrated based on a regular NN as well as a BNN in combination with NSGA-II and NSGA-III. The method with respect to NSGA algorithms is emphasized in Sections 3.3.4 and 3.3.5.

The remainder of this research is structured as follows: Section 2 summarizes important developments regarding the methods and algorithms, Section 3 describes the methodology, Section 4 summarizes the used data, Section 5 describes the main findings and Section 6 summarizes insights and conclusions drawn from the results. Appendices A-J give additional information regarding the method, data, R code and results.

[3] Epistemic uncertainty is uncertainty caused by lack of data, while aleatoric uncertainty is uncertainty caused by the noise in the process (Jospin et al., 2020).

2 Literature

Using NNs together with evolutionary algorithms is popular in studies having a science or physics topic, where often laboratory-based experiments are time consuming and/or expensive. Sometimes it is even impossible to take into account multiple objectives. Instead of doing laboratory-based experiments, it became more popular to study multiple objectives using deep learning. This is also the case for product design. A wide range of industries focuses more and more on sustainable and environmental-friendly interventions in combination with deep learning methods and optimization techniques, among others in the construction industry, in waste management and with respect to product design. However, literature focusing on packaging and using NNs in combination with MOGAs is scarce. Since literature both focusing on packaging and the proposed methods is scarce, the remainder of this Section summarizes developments regarding the proposed methods and algorithms, where the subject of these studies is sustainability or the environment.

Azari et al. (2016), Asadi, Da Silva, Antunes and Dias (2012) and Si et al. (2019) use NNs and MOGAs to analyze the optimal construction and design of buildings mainly focusing on minimizing energy use and costs. Results regarding single-objective optimization shows that NN is able to predict the behavior of the data well, since training and test mean squared error values are fairly low. To see the underlying trade-offs, Asadi et al. (2014) split up the optimization task in three cases: single-objective optimization, double-objective optimizations and triple-objective optimizations. Si et al. (2019) optimize four objective functions with respect to minimizing the building energy and maximizing the indoor thermal comfort using a NN and four evolutionary algorithms, namely NSGA-II, Multi-Objective Particle Swarm Optimization (MOPSO), Multi-Objective Simulated Annealing (MOSA) and Evolution Strategy (ES). Taking all performance measures into consideration, they concluded that NSGA-II performed best, followed by MOPSO. ES and MOSA were in this case the worst performers. Both Asadi et al. (2014) and Si et al. (2019) address to the problem of GA in building construction and design studies: models for construction and design of buildings are usually complex and therefore high computational times are required to get accurate results. Therefore, Magnier and Haghighat (2010) state that using a NN together with a GA is an efficient solution to encounter this problem.

Dantas, Leite and de Jesus Nagahama (2013) try to predict the compressive strength of concrete, where the sustainable intervention is that the concrete is made of demolition waste. They optimize the compressive strength using a NN. The NN shows high prediction accuracy equal to a R^2 of 0.928 and 0.971 for training and testing, respectively. Mahjoubi, Barhemat, Guo, Meng and Bao (2021) extend the paper of Dantas et al. (2013) by maximizing the compressive strength but also four other objectives. Mahjoubi et al. (2021) use two evolutionary algorithms, namely NSGA-III and Unified Non-dominated Sorting Genetic Algorithm (UNSGA) III. Eventually, they found that in this problem setting UNSGA-III performs slightly better in the multi-objective optimization than NSGA-III. The machine learning prediction model in this study is not a NN but they use a Support Vector Machine together with adaptive boosting followed by a extreme gradient boosting algorithm.

Ali Abdoli, Falah Nezhad, Salehi Sede and Behboudian (2012) and Azarmi, Oladipo, Vaziri and Alipour (2018) focus on waste managment using machine learning techniques. Both Abdoli et al. (2011) and Azarmi et al. (2018) compare a traditional approach, a multivariate regression model, with a machine learning approach, a NN, to predict waste generation. It was found that the NN was better able to model trends and fluctuations of solid waste generation than the multivariate regression model. However, they do not use a MOGA.

Regarding packaging there is less literature. Stoica, Antohi, Zlati and Stoica (2020) do not use machine learning techniques but an econometric efficiency model to study the impact of replacing plastic packaging by biodegradable biopolymers. They only study the financial impact and do not optimize multiple objectives. Up to my knowledge, the Bayesian method in combination with a MOGA of Zhang and Cui (2019) and the method of Zhou et al. (2009) based on a NN with a MOGA are closest to the method proposed in this paper but both do not make use of a BNN. Zhou et al. (2009) focuses on material selection of sustainable soda packaging by doing a multi-objective optimization (minimizing the weight, minimizing the life-cycle costs and minimizing the environmental impact) using a MOGA and a NN trained with the BP algorithm. Zhang and Cui (2019) use a MOGA together with a Bayesian approach to select green suppliers but do not use a BNN. Similarly as Zhou et al. (2009), Somkuwar et al. (2010) focus on the selection of materials for designing a product using a NN. However, they do not focus on packaging but on the design of coil springs and ceramic valves for taps. Bezazi, Pierce and Worden (2007) use a BNN to predict the fatigue life prediction of sandwhich composite materials and Siripatrawan and Jantawat (2008) use a NN to predict the shelf-life of a packaged snack but both do not use a multi-objective optimization framework and/or GAs. Bezazi et al. (2007) use both a NN and BNN and concludes that the Bayesian approach predicted the experimental data better than the frequentist approach.

Earlier research uses NNs and BNNs or NNs in combination with evolutionary algorithms, but up to my knowledge there exists no research combining a BNN together with a MOGA framework in such a problem setting. This paper demonstrates a framework being a combination of NN or BNN together with NSGA-II or NSGA-III, where the environmental footprint and packaging costs are optimized to select the most optimal materials.

3 Methodology

Based on Zhou et al. (2009) and Azari et al. (2016), a two-step research method is conducted. First, the multiple objectives are stated. Thereafter, the optimization algorithm is conducted. Figure 1 gives a complete overview of the method; Sections 3.1-3.3.5 give a detailed explanation of the distinct implementations of this method.

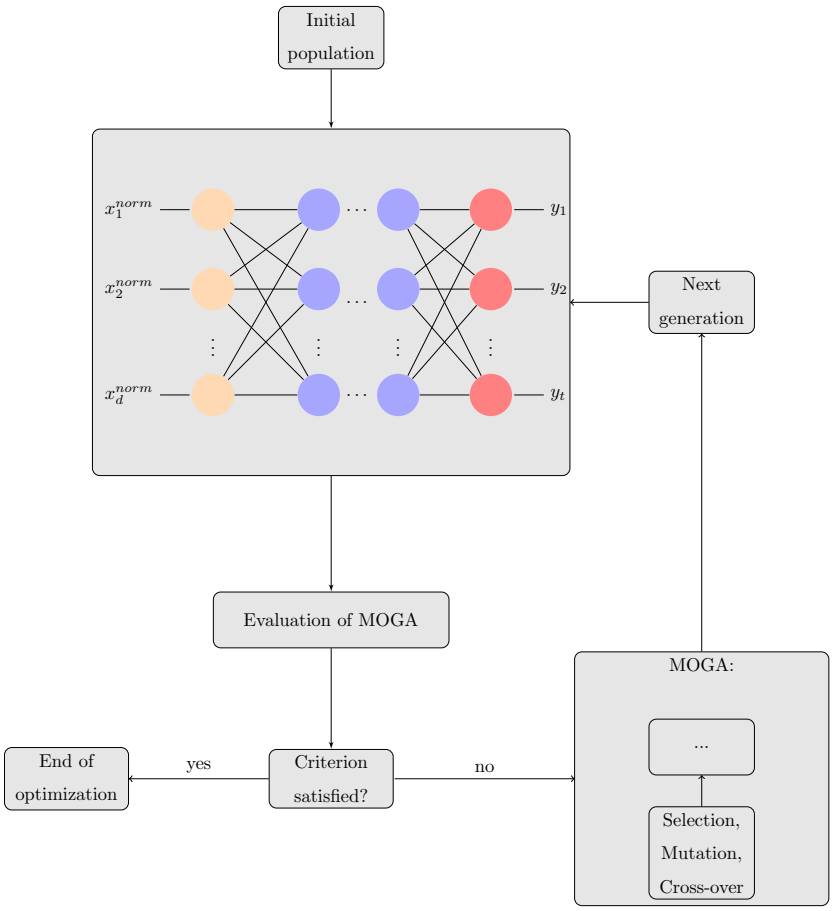


Figure 1: Flowchart describing the complete method.

3.1 Data pre-processing

Before training the NN, input data is normalized, such that magnitude and scale of the input data do not influence the NN training (Azari et al., 2016; Zhou et al., 2009). Also, normalization helps the optimizer to converge faster (Levy, 2016). Following Si et al. (2019) the minimum-maximum normalization method transforms all features onto the interval $[0, 1]$:

$$x_k^{norm} = \frac{x_k - min(x_k)}{max(x_k) - min(x_k)}, \tag{1}$$

where x_k^{norm} is the kth normalized input, x_k is the kth (raw) input, $min(x_k)$ is the minimum value of the kth input and $max(x_k)$ is the maximum value of the kth input.

3.2 Multi-objective optimization problems

Based on Unilever's environmental and savings ambition, this study minimizes the environmental impact and economic impact. Sections 3.2.1 and 3.2.2 summarize the multi-objective optimization shown in (2). Since it is not realistic to put all materials in one package and approximately 70% of the packaging data used in this study consists of plastics, a request from Unilever was to exclude some materials and create a specific material set that only contains plastics; this multi-objective optimization is shown in (3). This means that in (2) the optimization is conducted with respect to all ten materials, while in (3) the optimization is conducted with respect to only the six plastics.

$$\min_{\mathbf{m}\in\Omega} \begin{cases} z_{LCA}(\mathbf{m}) \\ z_{cost}(\mathbf{m}) \end{cases} \qquad (2)$$

$$\text{s.t.} \quad 0 \leq m_k \leq 100 \qquad \forall\, k = 1, \ldots, M$$

$$\sum_{k=1}^{M} m_k = 100$$

$$\min_{\mathbf{m}\in\Omega} \begin{cases} z_{LCA}(\mathbf{m}) \\ z_{cost}(\mathbf{m}) \end{cases} \qquad (3)$$

$$\text{s.t.} \quad 0 \leq m_k \leq 100 \qquad \forall\, k = 3, \ldots, 9$$

$$\sum_{k=1}^{9} m_k = 100$$

$$\lambda_k \in \{0, 1\} \qquad \forall\, k = 3, \ldots, 9$$

$$\lambda_i m_j = 0 \qquad \forall\, i = 3, \ldots, 9 \text{ and } j = 1, 2, 4, 5, 6, 7, 8, 10$$

$$m_8 + m_9 \geq 15$$

In (2) and (3) z_{LCA} equals the environmental footprint measured in points per gram package, z_{costs} equals the life-cycle costs measured in euros per gram package, m_k is the proportion of material k of the total package in percentage, M the total number of materials equal to 10 and λ_k equals a dummy variable indicating if material k is included or excluded in the problem. The vector of all material fractions $\mathbf{m} = [m_1, \ldots, m_M]$ includes the fraction of aluminium (m_1), the fraction of glass (m_2), the fraction of HDPE (m_3), the fraction of LDPE (m_4), the fraction of paper (m_5), the fraction of PET (m_6), the fraction of PP (m_7), the fraction of RPET (m_8), the fraction of RPP (m_9) and the fraction of steel (m_{10}) of the total packaging weight in percentages. Many countries in the European Union put an additional tax payment on packages that are not made of a specific percentage of recycled materials. To make it more beneficial for tax payments, in (3) the restriction is added that recycled PET (m_8) and recycled PP (m_9) should be at least equal to 15% of the total packaging composition. However, this tax reduction is not implemented in the computation of the packaging costs.

Following Asadi et al. (2012) three scenarios of optimizations are conducted (see Table 1). The first scenario involves only single-objective optimizations and the second scenario involves bi-objective optimizations. In this way, trade-offs between the two objective functions can be observed. It is expected that when a package is more (less) sustainable, z_{LCA} will be lower (higher) and z_{cost} will be higher (lower).

Table 1: Optimization scenarios including single- and bi-objective optimizations.

	Scenario I		Scenario II
	a	*b*	
Environmental impact	×		×
Economic impact		×	×

3.2.1 Environmental impact

To evaluate the environmental impact of packaging, the Life Cycle Assessment (LCA) method is used. The LCA model specifically takes into account the whole life-cycle of packaging because all phases could have an detrimental impact on the environment. The setup of the LCA framework and principles are constructed by the International Organization for Standardization (ISO, 2006). In short a life-cycle includes production phases, process phases, using phases, waste disposal phases and recycling phases. First, LCA requires to make an inventory analysis of extractions and emissions, i.e. define the environmental inputs (extraction of raw materials) and environmental outputs (waste and emission of pollutants). Thereafter, these inputs and outputs are translated to impact categories called Life Cycle Impact Assessment, e.g. global warming and ozone depletion. Finally, these impact categories are via weights combined into a single environmental indicator. To minimize the (detrimental) environmental impact of packaging, the total environmental footprint z_{LCA} expressed in points per gram of packaging is minimized:

$$\min_{\mathbf{m}\in\Omega} \quad z_{LCA}(\mathbf{m}) \tag{4}$$

The relation between the inputs and the environmental footprint z_{LCA} is predicted by a NN, and is further discussed in Section 3.3.1.

3.2.2 Economic impact

Since high packaging costs are not beneficial for both Unilever and the consumer, this objective aims to minimize the life-cycle costs of packaging. Life-cycle costs of packaging include purchase costs of raw materials, process costs, transportation costs, warehousing costs and recycling costs. To minimize the life-cycle costs z_{costs} expressed in euros per gram of packaging, the following measure will be minimized:

$$\min_{\mathbf{m}\in\Omega} \quad z_{costs}(\mathbf{m}) = \sum_{k=1}^{M} m_k z_{k,costs}, \tag{5}$$

where m_k is the material fraction of material k measured in percentages of the total weight of a package and $z_{k,costs}$ are the life-cycle costs with respect to material k expressed in euros per gram. Thus, the reader should be aware that the environmental footprint z_{LCA} is predicted by a NN and optimized, while for optimization of the life-cycle packaging costs z_{costs} the relation is assumed to be known and not predicted by a NN.

3.3 Optimization Algorithm

The optimization algorithm constitutes of two steps. The first step includes the NN, the second step includes the MOGA. In the first step both the NN and the BNN are considered, in the second step both NSGA-II and NSGA-III are considered

3.3.1 Neural Network

This Section discusses the NN in general form based on Hastie, Tibshirani and Friedman (2009), Nielsen (2015) and Taddy (2019). A standard multilayer NN, shown in Figure 2, consists of an input layer of d normalized inputs, L hidden layers and an output layer of t outputs, and in this study is trained using Stochastic Gradient Descent (SGD) and Back-Propagation (BP), where for each pair of normalized inputs and outputs (x_k^{norm},y_k) the algorithm first produces the output y_k by passing information from the input layer through the L hidden layers to the output layer. Then, the error between the predicted value and the actual value is propagated back from the output layer through the L hidden layers to the input layer to update the weights. In this way, the NN learns and improves itself. It is a bi-directional information flow. The training procedure stops when the output is close enough to the desired value. In this study the input and output layer constitute of $d = 13$ inputs and $t = 1$ output and are discussed later in this Section.

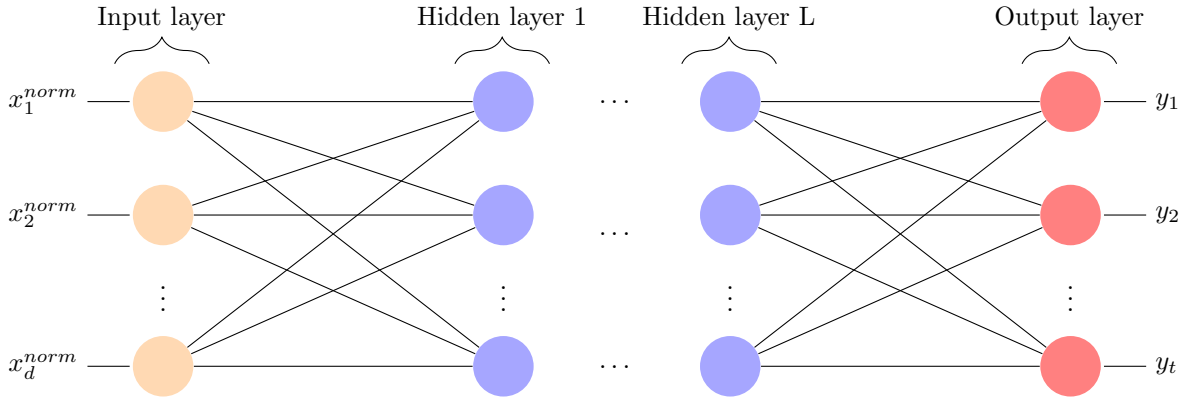


Figure 2: A multilayer NN having an input layer of d inputs, L hidden layers and an output layer of t outputs.

Inputs of neurons, which can be raw data or earlier processed data, are transformed by activation functions, which are non-linear such that they are able to explain complex relations, and returned as output of those neuron (Taddy, 2019). This paper will use Rectified Linear Unit (ReLu) (Liu, 2017) and the sigmoid (Han & Moraga, 1995) activation functions:

$$\delta_1(u) = max(0, u) \tag{6}$$

$$\delta_2(u) = \frac{1}{1 + \exp(-u)} \tag{7}$$

Compared to ReLu, sigmoid and hyperbolic tangent are less attractive because they saturate, meaning that large values go to 1 for hyperbolic tangent and small values go to -1 or 0 for sigmoid, and show only sensitivity for changes around their midpoints, which is 0.5 for sigmoid and 0 for hyperbolic tangent (Goodfellow, Bengio & Courville, 2017). Also, ReLu is faster compared to sigmoid and hyperbolic tangent (Nwankpa, Ijomah, Gachagan & Marshall, 2018). ReLu does not suffer from the

vanishing gradient problem that describes the problem of the NN weights that barely change compared to the previous iteration due to very small partial derivative updates. Sigmoid and hyperbolic tangent do suffer from this problem and the NN training can be completely stopped due to this. However, ReLu overfits more easily compared to sigmoid (Nwankpa et al., 2018).

Altogether, at each node k information will be transformed via a weighted sum over all inputs plus a bias term according to

$$y_k = \delta_a\Big(\sum_{i=1}^{d} w_{ki} x_i^{norm} + b_k\Big), \tag{8}$$

where y_k equals the output of node k, δ_a is activation function a ($a = 1, 2$), d the total number of input nodes preceding to the output y_k, w_{ki} is the weight for the normalized input x_i^{norm} at node k and b_k equals the bias at node k. The output of node k, y_k, can either be the input of a node in the next hidden layer or it can be a node in the output layer. In each epoch, the NN updates the weights, starting in the first epoch with initial weights.

To prevent overfitting the normalized data set will be pre-processed by randomly dividing into three subsets: 60% for the training data set to adjust the weights of the NN, 20% for the validation data set to evaluate the performance based on different hyperparameters, and 20% for the test data set to obtain the performance of the NN on new unseen (independent) data. Since the algorithm might be sensitive to the train, validation and test split, this three-way holdout method is repeated ten times to prevent for sampling bias. The performance measure is averaged over these ten 3-way holdout samples. After splitting the data into training, validation and test sets, the structure of the NN is determined using validation data, i.e. determining the number of layers and the number of neurons in each layer (hyperparameters). The number of neurons in the input (output) layer equals the number of input (output) variables. Due to the bias-variance trade-off the selection of the optimal number of neurons in the hidden layers of the NN is critical. A high bias corresponds to a low accuracy between the model and the training data (underfitting), whereas a high variance corresponds to a too complex model and a low generalization ability (overfitting). With underfitting both the training and test error are high, while with overfitting only the test error is high. A high generalization ability is achieved when both the training and test error are low. Following Esfandiari, Ghoreyshi and Jahanshahi (2017), the number of neurons in the hidden layer is determined via trial and error. During training the NN maximizes the likelihood $p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})$:

$$\theta^\star = \underset{\theta}{\operatorname{argmax}} \;\; \log\big[p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})\big], \tag{9}$$

where $\boldsymbol{\theta}$ is the parameter vector, $\mathcal{D} = (\mathbf{x}, \mathbf{y})$ is the training data and $\mathcal{M}$ is the model.

During the NN training the loss in every epoch is evaluated using the mean absolute error (MAE) and the root mean squared error (RMSE):

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \mid \hat{y}_i - y_i \mid \tag{10}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2} \tag{11}$$

where N equals the total number of data points, $\hat{y}_i$ equals the predicted value, y_i equals the actual value and $\bar{y}$ equals the mean of the actual values. MAE is expected to be more robust to outliers than RMSE; however, if model errors follow a Gaussian Normal distribution, then RMSE would be more suitable to use (Chai & Draxler, 2014). The performance of the NN or the cost function, which is the average of the losses of all training instances, is also evaluated using measures (10) and (11). If the above training performance measures converge over the epochs, the training is ended. Early stopping, which is a method that prevents the network from continuing training while the performance is not improving, is used to minimize the number of epochs during the stage to evaluate the performance based on different hyperparameters (Nielsen, 2015).

The NN constructed in this research predicts the total environmental footprint z_{LCA} in points per gram, where the material fractions of the packaging $\mathbf{m} = [m_1, \ldots, m_M]$ measured in percentages, the technical recyclability rate measured in percentages, the process energy GHG (Green House Gases) emission measured in carbon-dioxide equivalent and the life-cycle GHG emission measured in carbon-dioxide equivalent constitute the input layer of the network. The total environmental footprint is predicted and together with the life-cycle packaging costs z_{costs} minimized with respect to the material fractions $\mathbf{m}$ of the package. Presumably, the relation between the environmental footprint and the inputs is complex and therefore it is convenient to use a NN. For training and evaluation of the NN, the `torch` library (Falbel & Luraschi, 2021) is used to construct the NN from scratch.

3.3.2 Bayesian Neural Network

This research extends the paper of Zhou et al. (2009) by using a NN as well as a BNN, which is discussed in this Section. The NN discussed in Section 3.3.1 considers a frequentist approach. Its task is to maximize the likelihood $p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})$ to train the network. A Bayesian approach does not get a parameter point estimate but a complete probability distribution over parameters, the posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$. The posterior distribution describes beliefs about the value of each parameter. A prior distribution $p(\boldsymbol{\theta})$ describes the initial belief of the parameters before observing the data, based on theory or empirical results. Since no clear theory exists regarding the distribution of the environmental footprint with respect to the inputs, the default option, a Gaussian Normal distribution, is chosen. However, results could be sensitive to the prior choice and other distributions could give better results (Silvestro & Andermann, 2020; Fortuin, Garriga-Alonso, Wenzel, Rätsch, Turner, van der Wilk & Aitchison, 2021). The aim of a BNN, shown in Figure 3, is to find posterior distributions for all weights and biases.

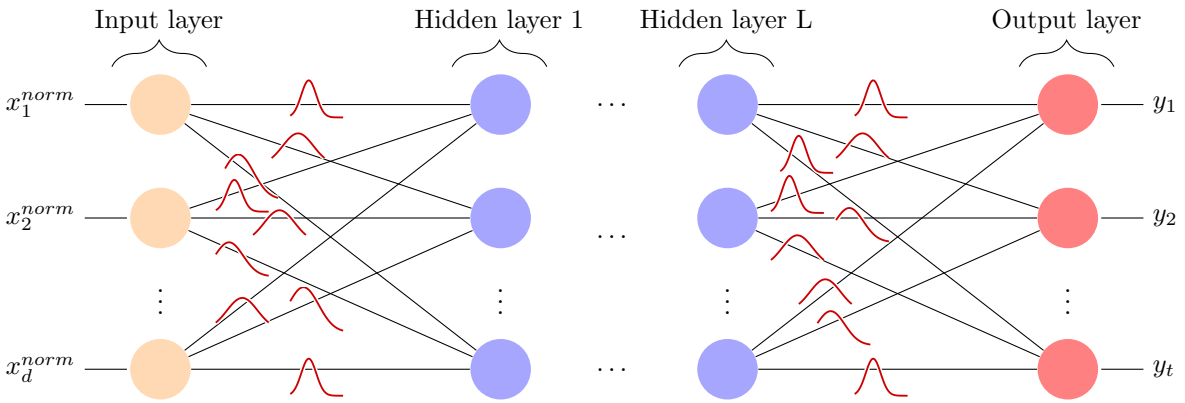


Figure 3: A BNN having an input layer of d inputs, L hidden layers and an output layer of t outputs.

Using Bayes' rule (Bayes, 1763) the posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$ over parameters $\boldsymbol{\theta}$, which constitute all weights $\mathbf{w}$ and biases $\mathbf{b}$, of the model $\mathcal{M}$ after observing the training data $\mathcal{D}$ can be written as:

$$p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M}) \propto p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta}), \tag{12}$$

In the BNN for each node k, again information will be transformed according to (8) but now the weights follow a distribution, which in this paper is assumed to be a Gaussian Normal distribution:

$$w_{ki} \sim \mathcal{N}(\mu_{ki}, \sigma_{ki}^2) \tag{13}$$

Then, based on observed training data $\mathcal{D}$ including observations y_i with $i = 1, ..., N$, predictions on new unobserved test data y^{n+1} are obtained using:

$$p(y^{n+1} \mid \mathcal{D}, \mathcal{M}) = \int p(y^{n+1} | \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta} | \mathcal{D}) d\boldsymbol{\theta} \tag{14}$$

Analytically, the integral in (14) cannot be evaluated, therefore the numerical method Variational Inference (VI) is used to solve the integral (Graves, 2011). A more common numerical method is a Markov chain Monte Carlo sampling algorithm; this study does not considers this method because it is usually slower than VI (Kochurov & Wiecki, 2017). VI methods solve the integral using an optimization technique with a distance measure called the Kullback Leibler (KL) divergence (Kullback & Leibler, 1951), where the posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$ is approximated using a variational distribution $q(\boldsymbol{\theta} \mid \boldsymbol{\eta})$, with $\boldsymbol{\eta} = (\boldsymbol{\mu}, \boldsymbol{\sigma})$ the distribution parameters of a Gaussian distribution, that is as close as possible to the true posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$. Both the real posterior distribution and the variational distribution are unknown but using the latter can solve the optimization problem using numerical methods.

VI methods determine the optimal value of $\boldsymbol{\theta}$ by minimizing the KL divergence between the variational distribution $q(\boldsymbol{\theta} \mid \eta)$ and the posterior distribution $p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$[4]:

$$
\begin{aligned}
&KL\big[q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \mid\mid p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})\big] \\
&= \int q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \log\left(\frac{q(\boldsymbol{\theta} \mid \boldsymbol{\eta})}{p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})}\right) d\boldsymbol{\eta} \\
&\dots \\
&= VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) + \log(p(\mathcal{D} \mid \mathcal{M}))
\end{aligned}
\tag{15}
$$

Minimizing the KL divergence in (15) without the last term, since it does not depend on $\boldsymbol{\theta}$, gives the minimization problem of the variational free energy $VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$ (Graves, 2011):

$$
\boldsymbol{\eta}^{\star} = \underset{\boldsymbol{\eta} \in \Omega}{\operatorname{argmin}} \; VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) \tag{16}
$$

The variational free energy $VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$ in (16) can be approximated by drawing θ_i from the variational distribution $q(\boldsymbol{\theta} \mid \boldsymbol{\eta})$:

$$
VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) \approx \frac{1}{N} \sum_{i=1}^{N} \Big[\log(q(\theta_i \mid \boldsymbol{\eta})) - \log(p(\mathcal{D} \mid \theta_i, \mathcal{M})) - log(p(\theta_i \mid \mathcal{M})) \Big] \tag{17}
$$

Assume that both the variational posterior distribution $q(\boldsymbol{\theta} \mid \boldsymbol{\eta})$ and the prior $p(\boldsymbol{\theta} \mid \boldsymbol{\eta})$ follow a Gaussian distribution, according to

$$
q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \sim \mathcal{N}(\boldsymbol{\theta} \mid \boldsymbol{\mu}, \boldsymbol{\sigma} I) \text{ and} \tag{18}
$$

$$
p(\boldsymbol{\theta} \mid \mathcal{M}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{19}
$$

The BNN training is based on the Stochastic Variational Inference (SVI), which is SGD for VI (Hoffman, Blei, Wang & Paisley, 2013). The problem is that BP training stops in the hidden nodes of the BNN due to stochasticity. However, Bayes-by-Backprop (BBB) is a solution to implement this in practice. The remainder of this Section is based on Blundell, Cornebise, Kavukcuoglu & Wierstra (2015) and Jospin et al. (2020).

The idea of BBB is to use a reparametrization trick to ensure that BP works.[5] The Gaussian variational distribution in (18) can be reparameterized to: $\boldsymbol{\theta} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$, where $\odot$ is element-wise multiplication and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, I)$. Similarly to Blundell et al. (2015) the standard deviation is parameterized according to $\boldsymbol{\sigma} = \log(1 + \exp(\boldsymbol{\rho}))$, so that it cannot become negative. Then, the non-variational noise $\boldsymbol{\epsilon}$ is sampled independently from $\boldsymbol{\mu}$ and $\log(1 + \exp(\boldsymbol{\rho}))$.[6] The reparameterization can then be used to evaluate the VI approximation expressed in (17). BP is conducted by calculating the gradients of $VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$ with respect to the variational parameters $\boldsymbol{\eta}$. Finally, the variational parameters are updated using learning rate α. These steps are conducted for a predefined total number of epochs, TE. Algorithm (1) gives a summary of the BBB algorithm that learns the values of the variational parameters $\boldsymbol{\eta}$.[7] Similar to NN,

[4] For convenience the variational free energy, $VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$, is written without $\mathcal{D}$ and $\mathcal{M}$. It could also be written as $VFE(\boldsymbol{\theta}, \boldsymbol{\eta} \mid \mathcal{D}, \mathcal{M})$. For details about the VI method and KL divergence see Appendix B.

[5] For the proof of the reparameterization trick see Kingma & Welling (2013).

[6] If all $\sigma_j = 0$, then $\theta_j = \mu_j$ and it reduces to a normal BP algorithm in a deterministic NN.

[7] This algorithm is specifically for the choice of this paper's priors; see Jospin et al. (2020) for the general case.

the performance of the BNN is evaluated according to (10) and (11), and for training and evaluation of the BNN, the the `torch` library (Falbel & Luraschi, 2021) is used to construct the BNN from scratch.

Algorithm 1 Bayes-by-Backprop algorithm.

Initialize: $\eta = \eta_0$
for $i = 0$ `to` TE **do**
  `Draw` $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  `Reparameterization:` $\boldsymbol{\theta} = \boldsymbol{\mu} + \log(1 + \exp(\boldsymbol{\rho})) \odot \boldsymbol{\epsilon}$
  $VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) = \log(q(\boldsymbol{\theta} \mid \eta)) - \log(p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})) - log(p(\boldsymbol{\theta} \mid \mathcal{M}))$
  `Back-propagation:` $\nabla_\eta VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$
  `Update:` $\eta = \eta - \alpha \nabla_\eta VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$
end for

3.3.3 Single-objective Genetic Algorithm

This Section summarizes the algorithm of a general single-objective GA based on (Konak et al, 2006). As mentioned before GA is an evolutionary algorithm and an iterative procedure where, similarly to natural evaluation, a population of chromosomes goes through a number of stages called a generation. Generation g consists of a selection stage, a mutation stage, a cross-over stage and an elitism stage. A population P_g is randomly initialized and with every generation the population becomes fitter and fitter.

A GA uses a population of possible solutions, i.e. chromosomes. A chromosome, which is a solution vector, is made of (discrete) genes. A chromosome is a string of characteristics; each gene corresponds to a characteristic of the chromosome. Therefore, population in generation g, P_g, is defined as C chromosomes of the population in generation g. Due to limited computational power, in this study the number of chromosomes C is equal to 12. Each chromosome equals a package consisting of material fractions $\mathbf{m} = [m_1, \ldots, m_M]$. Thus, the genes of the chromosomes are equal to the material fractions $\mathbf{m}$ measured in percentages. In the multi-objective optimization in (2) there are ten genes, while in the multi-objective optimization in (3) there are six genes. In order to imitate natural selection, a fitness function determines how good a chromosome is. A chromosome with a high fitness score FI is most likely to be selected for reproduction.

Generation $g = 0$ starts with a random generation of C chromosomes for population g, P_g. Also, the cross-over probability P^c, the mutation probability P^m and the maximum number of generations G are initially determined. Due to limited computational power, in this study the number of generations G is equal to 10. Based on Yang, Chien & Ting (2015) in this study the P^c and P^m are set to 0.9 and 0.03, respectively. In the selection stage, chromosomes of population P_g will be selected based on their fitness score FI. Thereafter, to create offsprings Q_g the cross-over and mutation operators are used.[8] In the (single-point) cross-over stage, two chromosomes from population P_g are selected based on their level of FI. The higher FI, the more likely they are selected. Then, offsprings Q_g are created by interchanging a specific part of the chromosomes. If no interchange takes place, the offsprings are

[8] Details regarding the cross-over and mutation operator can be found in Appendix C.

similar to the parents. Not all pairs of chromosome undergo the cross-over operation; it is based on the cross-over probability P^c. In the mutation stage, one gene in a chromosome at a specific locus of each off-spring is changed by some probability P^m. This brings back genetic diversity into the population and prevents the algorithm to get stuck in local. Then, the elitism stage guarantees that C chromosomes from Q_g with high fitness scores FI are selected and carried over to the next generation $g+1$, so that the genetic quality is improved compared to generation g. The population becomes fitter and fitter during GA, eventually if the stopping criterion is satisfied, the population converges, the search is terminated and the current population is returned. Algorithm 2 summarizes the GA algorithm.

Algorithm 2 Single-objective Genetic Algorithm.

function FITNESS(P_g)
    computeFitness(P_g)
    return FI_g
end function

function SELECTION(P_g)
    $Parent_1$ = P_g[max(Fitness(P_g))]
    $Parent_2$ = P_g[max(Fitness($P_i - Parent_1$))]
    return $[Parent_1, Parent_2]$
end function

function CROSSOVER($[Parent_1, Parent_2]$,P^c)
    Q_g = computeCrossOver($[Parent_1, Parent_2]$,P^c)
    return Q_g
end function

function MUTATION(Q_g,P^m)
    Q_g = computeMutation(Q_g,P^m)
    return Q_g
end function

Initialize: $G = 10$, $C = 12$, $P^c = 0.9$, $P^m = 0.03$
P_0 = rand(C)
FI_0 = Fitness(P_0)
while not termination condition **do**
    for $g = 0$ to G **do**
        Q_g = Mutation(CrossOver(Selection(P_g),P^c),P^m)
        maxQ_g = max(Fitness(Q_g))

        if maxQ_g > min(Fitness(P_g)) **then**
            swap(maxQ_g,min(Fitness(P_g))
        end if
        $g \leftarrow g + 1$
    end for
end while
return P_{g+1}

3.3.4 NSGA-II

This Section gives an overview of NSGA-II, which uses crowding distance (CD) to obtain the fittest population (Deb et al., 2002 ;Konak et al., 2006). Chromosomes with a higher CD are considered as better.[9] Similarly as GA in Section 3.3.3, NSGA-II starts with a random population P_g in generation $g = 0$ and initializes parameters. Offspring Q_i are created by applying the selection, cross-over and mutation operator. If the stopping criterion is satisfied, the algorithm already stops and returns population P_g for generation $g = 0$. If not, the algorithm continuous by making a combined population $R_g = P_g \cup Q_g$. Thus, R_g is twice the size of P_g. The fast non-dominated sorting algorithm[10] is applied to identify all non-dominated fronts $F_1, \ldots, F_R$ in R_g. Then, for all F_j with $j = 1, \ldots, R$ the following steps are conducted:

1. Calculate the CD of all chromosomes in F_j.

2. Create the population of the next generation P_{g+1} by including chromosomes of the highest ranked fronts until the size of P_{g+1} equals C (Case 1). Then, if a front contains too many chromosomes and exceeds size C, the CD measure is used (Case 2) because all chromosomes in that front have the same performance.
   Case 1: If $\mid P_{g+1} \mid + \mid F_j \mid \leq C$, then $P_{g+1} = P_{g+1} \cup F_j$
   Case 2: If $\mid P_{g+1} \mid + \mid F_j \mid > C$, then add the least crowded $C - \mid P_{g+1} \mid$ chromosomes from F_j to P_{g+1}.

Thereafter, again the selection, cross-over and mutation operator are applied to create offspring Q_{g+1} of size C. If the stopping criterion is satisfied, the search is terminated and the current population is returned. Algorithm 3 summarizes the NSGA-II algorithm. The `mco` library (Mersmann, Trautmann, Steuer, Bischl & Deb, 2020) is used to solve the multi-objective optimization using NSGA-II.

Algorithm 3 NSGA-II.

function FNSA(R_g)
    $[F_1, \ldots, F_R]$ `= computeFronts(`R_g`)`
    return $[F_1, \ldots, F_R]$
end function

function CrowdingDistance(F_j)
    CD_j `= computeCrowdingDistance(`F_j`)`
    if $\mid P_{g+1} \mid + \mid F_j \mid \leq C$ **then**
        $P_{g+1} = P_{g+1} \cup F_j$
    else
        $P_{g+1} = P_{g+1} \cup F_j[1 : C - \mid P_{g+1} \mid]$
    end if
    return CD_j
end function
Initialize: $G = 10$, $C = 12$, $P^c = 0.9$, $P^m = 0.03$
P_0 `= rand(`C`)`
Q_0 `= Mutation(CrossOver(Selection(`P_0`),`P^c`),`P^m`)`

[9] Details about the CD can be found in Appendix D.
[10] The fast non-dominated sorting algorithm verifies pairwise if a chromosome dominates other chromosomes. Then, the first most highest ranked front contains all chromosomes that are dominated by no other chromosomes. Chromosomes do get a penalty term of 1 when they are dominated by other chromosomes, such that they are classified into a least ranked front.

17: **while** `not termination condition` **do**
18: **for** $g = 0$ `to` G **do**
19: $R_g \leftarrow P_g \cup Q_g$
20: $[F_1, \ldots, F_R]$ `= FNSA(`R_g`)`
21: **for** $j = 1$ `to` R **do**
22: $CD_j =$ `CrowdingDistance(`F_j`)`
23: $j \leftarrow j + 1$
24: **end for**
25: $Q_{g+1} =$ `Mutation(CrossOver(Selection(`P_{g+1}`,`P^c`),`P^m`)`
26: **end for**
27: $g \leftarrow g + 1$
28: **end while**
29: `return` P_{g+1}

Note:

1. $\mid P_{g+1} \mid$ and $\mid F_j \mid$ equal the number of chromosomes in population $g+1$ and in front j, respectively.
2. In above algorithm functions `Selection`, `Mutation` and `CrossOver` are similar to the functions in Algorithm 2.

3.3.5 NSGA-III

This Section summarizes NSGA-III, which is the extended version of NSGA-II in Section 3.3.4 and replaces the computationally expensive CD operator with another approach to select chromosomes for the population in the next generation (Deb & Jain, 2013*a*; Deb & Jain, 2013*b*). Deb & Jain (2013*a*) determine reference points on a hyperplane, so that diversity in chromosomes among the population remains. The problem with NSGA-II is that when the number of objectives increases, proportionally there are more non-dominated than dominated solutions in a random set of objective vectors (Deb & Jain, 2013*a*; Deb & Jain, 2013*b*). They use the method of Das and Dennis (1998) to determine the reference points on the hyperplane. Reference points on a normalized hyperplane are defined such that they are equally inclined to all axes of the objectives and intercept with one of the axes of the objectives.

Algorithm 4 gives the pseudocode of NSGA-III. Similarly to NSGA-II, NSGA-III starts with a random population, a merged parent and offspring population and the fast non-dominated sorting algorithm to determine the fronts. They both use the same crossover and mutation operators, `CrossOver` and `Mutation` respectively. Then, the selection of fronts to be included in the new population takes place until the size of the new population equals or exceeds C. Now NSGA-II behaves different than NSGA-III. For NSGA-II the chromosomes in the last front F_L that can be partially included is based on the CD operator, while in NSGA-III this is based on the reference points on the normalized hyperplane.

After normalization of each objective, for NSGA-III the remaining chromosomes from front F_L that will be selected to form a new survival population S is based on the chromosomes in F_L that will maximize the diversity of the population the most. To ensure diverse and well-distributed solutions, chromosomes of F_L that are associated with each of the reference points are selected, so that diversity of the survival population S is maintained (Yannibelli et al., 2020). Every chromosome in S is associated with a reference point. That is why for each reference point on the hyperplane a reference line is defined by connecting the reference point with the origin. Thereafter, the perpendicular distance of each chromosome in S from each reference line is computed. Then, the reference line corresponding

to its reference point that is closest to the chromosome in the normalized objective space is associated with the chromosome (Deb & Jain, 2013*a*; Deb & Jain, 2013*b*).

It might be the case that reference points are not associated with any chromosomes or do have multiple associations. In the `Niching` operation the number of chromosomes associated with each reference point is counted and based on this count inclusion or exclusion for the population of the next generation is considered (Deb & Jain, 2013*a*; Deb & Jain, 2013*b*). For details regarding functions `Normalize`, `Associate`, `Niching` see Appendix E. The `MaOEA` library (Irawan, 2020) is used to solve the multi-objective optimization using NSGA-III. Details regarding the complete R code can be found in Appendix G or in `HTTPS://COLAB.RESEARCH.GOOGLE.COM/DRIVE/1HWC7E1NPLGFKW7PJGPKWLL7YBZZRUNPC?USP=SHARING`.

Associated to this study, the NSGA algorithms optimize the multi-objective optimization problem discussed in Section 3.2 with respect to the material fractions. It should be emphasised that the NSGA algorithms solely optimize with respect to the material fractions, while the NN or BNN predicts the environmental footprint based on multiple inputs (material fractions, emissions and technical recyclability rate).

Algorithm 4 NSGA-III.

Initialize: $G = 10$, $C = 12$, $P^c = 0.9$, $P^m = 0.03$, H, Z^s
P_0 = `rand`(C)
Q_0 = `Mutation(CrossOver(Selection(`P_0`),`P^c`),`P^m`)`
while `not termination condition` **do**
  for $g = 0$ `to` G **do**
    $S = \emptyset$
    $R_g \leftarrow P_g \cup Q_g$
    $[F_1, \ldots, F_R]$ = `FNSA(`R_g`)`
    for $j = 1$ `to` R **do**
      `Define` F_L = `last front to be included`
      if $\mid S \mid + \mid F_j \mid < C$ **then**
        $S \leftarrow S \cup F_j$
      else if $\mid S \mid + \mid F_j \mid = C$ **then**
        $P_{g+1} \leftarrow S \cup F_L$
      else
        `Normalize objective space & create` Z_r`:`
        $z_k^{norm}, S^{norm}, F_L^{norm}, Z_r^{norm}, \cdot \leftarrow$ `Normalize(`$z_k, S, F_L, Z_r, \cdot$`)`
        `Associate each chromosome of` S^{norm} `with reference point` Z_r^{norm}`:`
        for $k = 1$ `to` $\mid S \mid$ **do**
          $[\pi_s, \Delta(s)]$ = `Associate(`S_k^{norm}, Z_r^{norm}`)`
        end for
        `Remaining chromosomes from` F_L `to fill up` S`:`
        $S \leftarrow S \cup$ `Niching(`$F_L^{norm}, C - \mid S \mid, \cdot$`)`
      end if
      $j \leftarrow j + 1$
    end for
    $P_{g+1} \leftarrow S$
    Q_{g+1} = `Mutation(CrossOver(Selection(`P_{g+1}`),`P^c`),`P^m`)`
  end for
  $g \leftarrow g + 1$
end while

32: `return` P_{g+1}

Note:

1. In above algorithm functions `Selection`, `Mutation`, `CrossOver` and `FNSA` are similar to the functions in Algorithms 2 and 3.
2. To keep it concise, some details regarding `Normalize`, `Associate` and `Niching` are left out, see Appendix E for further details.
3. In above algorithm, Z^s are H structured reference points, Z^r is the reference set, z_k is objective function k, $\pi(s)$ is the closest reference point and Δ is the distance between s and $\pi(s)$.

4 Data

To train, validate and test the proposed method, the Packaging portfolio (Unilever, 2022*a*), Food database (Unilever, 2022*b*), Idemat data set (Delft University of Technology, 2022), average process GHG emission data (Containers & Good, 2016) and average life-cycle GHG emission data (Kissinger, Sussmann, Moore & Rees, 2013) are used.[11] The Food database provides information of 17564 products, where each product has one of 364 packages retrieved from the Packaging Portfolio. However, due to missing data only 4225 observations remain. For each product, data of the material fractions in percentages, the technical recyclability rate in percentages and the life-cycle costs in euros per gram of packaging are extracted from the Packaging Portfolio and the Food database. To be able to evaluate the LCA model and to be able to predict this, data regarding life-cycle assessment based on the method of the ISO (2006) is needed. The Idemat data set provides the environmental footprints of the materials in points per kilogram. This is extracted and transformed to the environmental footprint of every package measured in points per gram of packaging. Table 2 shows the units of measurements of all variables. Since neural networks perform better if more data points are used, a simulation data set of 20.000 observations is created that is based on the data of Unilever. Also, some materials are underrepresented in the original Unilever data set, e.g. RPET and RPP are included in only 8 and 28 of the packages in the original Unilever data set. For the simulated data a wider range of values is covered. In the remainder of this paper, the Unilever data set is referred to as the original data set of 4225 observations and the simulation data set is referred to as the simulated data set of 20.000 observations. Appendix F gives detailed information about both data sets.

Table 2: Units of data variables.

Data	Unit
Environmental footprint	Points per g of packaging
Life-cycle GHG emissions	CO2E in g
Life-cycle costs	€ per g of packaging
Materials	% of total packaging weight
Process energy GHG emissions	CO2E in g
Technical recyclability rate	%

Note: CO2E is carbon-dioxide equivalent, which makes it possible to compare different GHG.

[11] (Containers & Good, 2016) also provide data regarding the transportation energy per material. This is not taken into account, since this is too sensitive regarding the assumptions made. However, the life-cycle emission data of (Kissinger et al., 2013) takes into account the complete (average) life-cycle emissions per material, so also the transportation emissions. Therefore, in this study the life-cycle emissions is rather a proxy than an exact measure with respect to every package, since these average transportation emissions are not exactly equal to the transportation emissions of Unilever's products. Also, (Kissinger et al., 2013) do not distinguish between virgin and recycled materials.

5 Results

This Section describes the results of the combined algorithm of an evolutionary algorithm, NSGA-II or NSGA-III, and a neural network, NN or BNN.

5.1 Grid search hyperparameters and training of the networks

After normalization of the inputs, the network is trained on several hyperparameter settings using the train and validation data sets. The grid search is limited to search within three hidden layers and ten neurons per hidden layer. On top of that the total number of hidden neurons in the network is restricted to be smaller than or equal to 25. Adding more hidden layers and/or having more neurons in the network did not improve the performance substantially and overfitting is prevented. Via trial-and-error a learning rate of 0.0005 is chosen for the actual network training using the train and test data. A learning rate of 0.0005 performed best and learning rates bigger than 0.0009 do not learn the patterns of the networks well. Table 3 shows the performance measures of the network training, which is the average performance over ten 3-way holdout samples.

Table 3: Generalization performance over ten 3-way holdout samples using 10.000 epochs; the best performing networks are indicated in bold.

	(MAE train, MAE test)	(RMSE train, RMSE test)
Unilever + All + NN + ReLu	(**0.0489**, **0.0482**)	(**0.0673**, **0.0694**)
Unilever + All + NN + Sigmoid	(0.1396, 0.1394)	(0.1463, 0.1462)
Unilever + All + BNN + ReLu	(0.0514, 0.0521)	(0.0689, 0.0684)
Unilever + All + BNN + Sigmoid	(0.1394, 0.1394)	(0.1463, 0.1462)
Unilever + Plastics + NN + ReLu	(0.0535, 0.0551)	(0.0664, 0.0709)
Unilever + Plastics + NN + Sigmoid	(0.2326, 0.2333)	(0.1519, 0.1529)
Unilever + Plastics + BNN + ReLu	(**0.0535**, **0.0550**)	(**0.0647**, **0.0668**)
Unilever + Plastics + BNN + Sigmoid	(0.1494, 0.1503)	(0.1603, 0.1614)
Simulation + All + NN + ReLu	(0.1528, 0.1528)	(0.0808, 0.0808)
Simulation + All + NN + Sigmoid	(0.0971, 0.0972)	(0.1023, 0.1025)
Simulation + All + BNN + ReLu	(**0.0211**, **0.0211**)	(**0.0256**, **0.0257**)
Simulation + All + BNN + Sigmoid	(0.0971, 0.0973)	(0.1024, 0.1025)
Simulation + Plastics + NN + ReLu	(0.0367, 0.0367)	(0.0349, 0.0349)
Simulation + Plastics + NN + Sigmoid	(0.1100, 0.1100)	(0.1105, 0.1105)
Simulation + Plastics + BNN + ReLu	(**0.0072**, **0.0073**))	(**0.0087**, **0.0091**)
Simulation + Plastics + BNN + Sigmoid	(0.1100, 0.1101)	(0.1104, 0.1105)

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors.

In general training a BNN on one 3-way holdout sample takes more time than training a NN. As expected, for all simulation cases the test error is slightly higher than the train error. Due to the small number of observations, the original Unilever data shows sensitivity to the 3-way holdout method and suffers arbitrarily from sampling bias. The test performance is arbitrarily smaller than the train performance in some cases, which means that it is a coincidence that it fits the sample well, and in some cases it is vice versa as would be expected. As can be seen in Figures 4 and 5, sigmoid activation needs more epochs to converge, in general around 30.000 to converge completely, while for ReLu only 10.000 epochs are needed to converge completely and convergence already starts around 500 epochs. Therefore, in the complete algorithm ReLu activation has been chosen to train the networks for both the NN and the BNN.

As highlighted in Table 3, in the remainder of this Section results are shown for ReLu activation, the simulation data set and MAE loss. Using the train and validation data sets, the most optimal hyperparameters for training networks inside the NSGA-II and NSGA-III algorithms for the highlighted cases are shown in Table 4. In Appendix H results for RMSE loss can be found.

Table 4: Most optimal hyperparameters.

	Number of hidden layers	Neurons in hidden layers
Simulation + All + NN + ReLu	2	(2,4)
Simulation + All + BNN + ReLu	2	(2,4)
Simulation + Plastics + NN + ReLu	3	(1,4,2)
Simulation + Plastics + BNN + ReLu	3	(1,2,3)

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors, MAE loss, simulation data, ReLu activation.

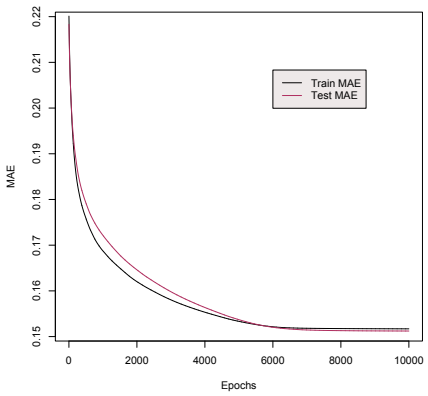
For the BNN different normal distributions are considered. Table 5 gives an overview of the generalization performance on ten 3-way holdout samples and shows that using a Gaussian Normal distribution different than the standard normal distribution does not make a major difference in generalization performance for both including all materials and including only plastics. Therefore, NSGA algorithms in combination with a BNN use standard normally distributed priors.

Table 5: Normal distributions considered for priors in the BNN.

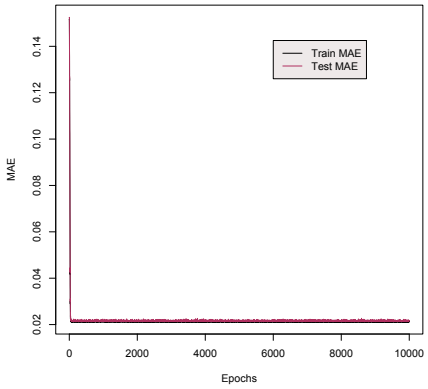
Distributions	(MAE train, MAE validation)	
	All	*Plastics*
$\mathcal{N}(\mu = 0, \sigma = 1)$	(0.0211, 0.0211)	(0.0072, 0.0073)
$\mathcal{N}(\mu = 0.5, \sigma = 0.5)$	(0.0215, 0.0212)	(0.0072, 0.0073)
$\mathcal{N}(\mu = 0.5, \sigma = 1)$	(0.0218, 0.0216)	(0.0072, 0.0074)
$\mathcal{N}(\mu = 0, \sigma = 2)$	(0.0214, 0.0213)	(0.0073, 0.0074)
$\mathcal{N}(\mu = 0.5, \sigma = 2)$	(0.0215, 0.0214)	(0.0073, 0.0074)
$\mathcal{N}(\mu = 5, \sigma = 0.5)$	(0.0216, 0.0215)	(0.0073, 0.0074)

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors, MAE loss, simulation data, ReLu activation.

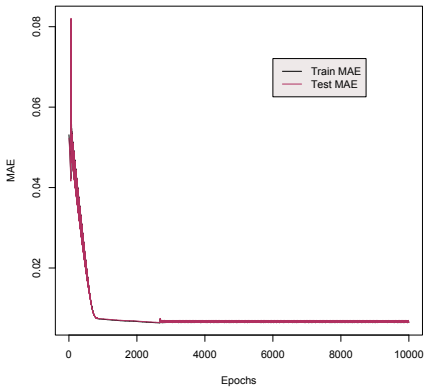
Figure 4: MAE loss using simulation data, training for 10.000 epochs and in case of the BNN then priors follow a standard Normal distribution.



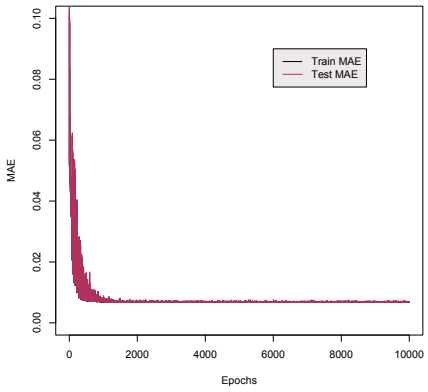
(a) All - NN - ReLu



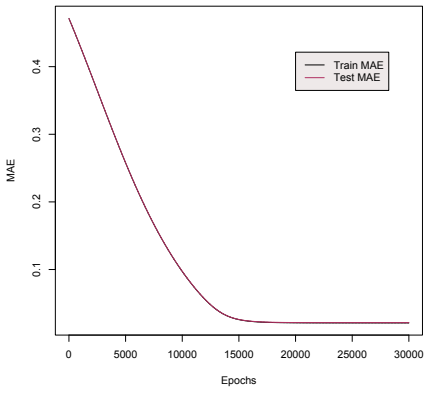
(b) All - BNN - ReLu



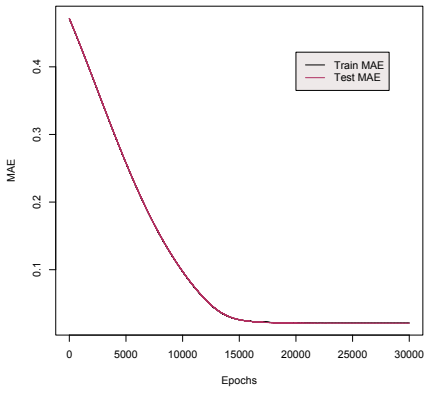
(c) Plastics - NN - ReLu



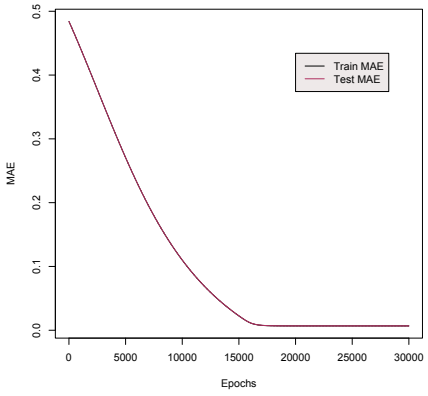
(d) Plastics - BNN - ReLu



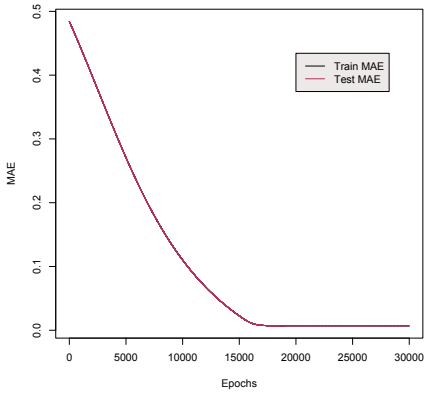
(e) All - NN - Sigmoid



(f) All - BNN - Sigmoid

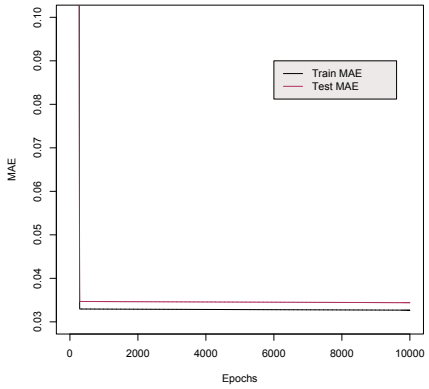


(g) Plastics - NN - Sigmoid

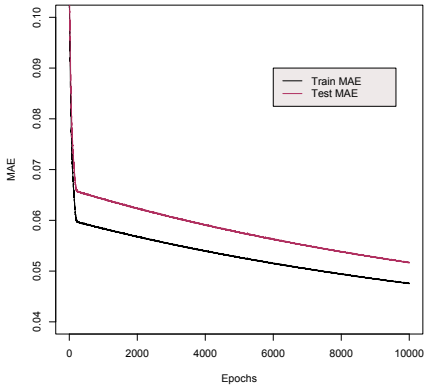


(h) Plastics - BNN - Sigmoid

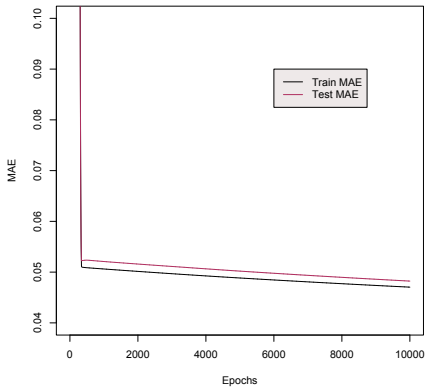
Figure 5: MAE loss using Unilever data, training for 10.000 epochs and in case of the BNN then priors follow a standard Normal distribution.



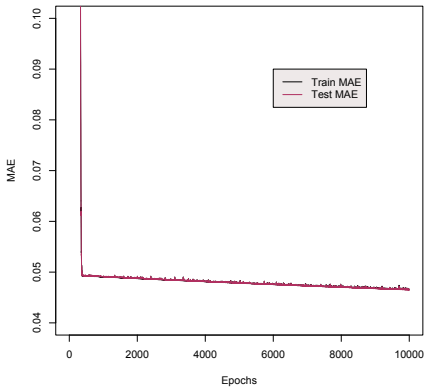
(a) All - NN - ReLu



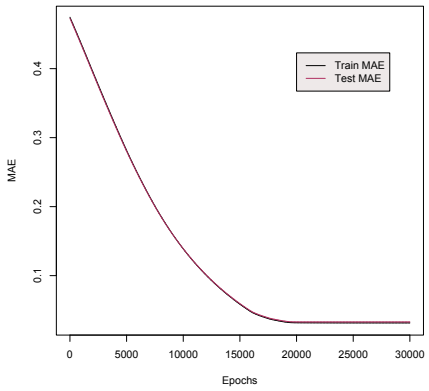
(b) All - BNN - ReLu



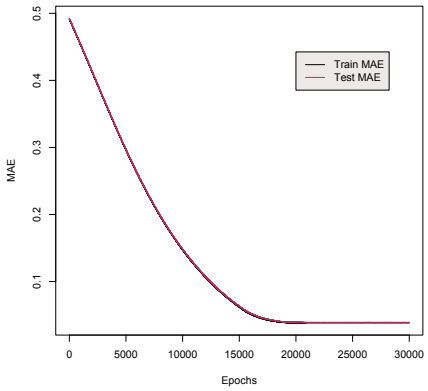
(c) Plastics - NN - ReLu



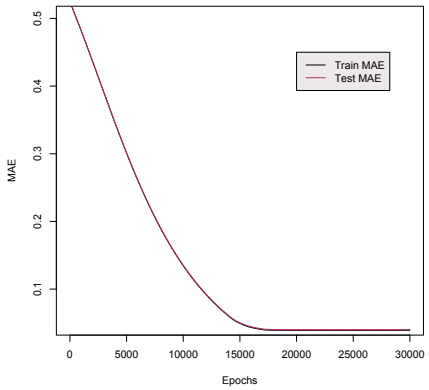
(d) Plastics - BNN - ReLu



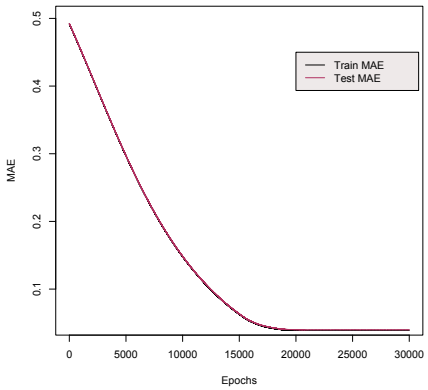
(e) All - NN - Sigmoid



(f) All - BNN - Sigmoid



(g) Plastics - NN - Sigmoid



(h) Plastics - BNN - Sigmoid

5.2 NSGA-II

Conducting the NSGA algorithms for all 3-way holdout samples is too computationally intensive, therefore it is trained for the sample attaining the lowest loss. This potentially could give a slight optimistic bias but Table 6 shows that comparing the generalization performance averaged over ten 3-way holdout samples is in line with the generalization performance of one sample. However, 3-way holdout samples using the original Unilever data often suffer from sampling bias and show big differences among each other. Due to the small number of observations, there is arbitrariness in the sampling.

Table 6: Comparison of performance on ten 3-way holdout samples and one sample.

	(MAE train, MAE test) on ten 3-way holdout samples	(MAE train, MAE test) on one sample
Simulation + All + NN + ReLu	(0.1529, 0.1529)	(0.1517, 0.1512)
Simulation + All + BNN + ReLu	(0.0211, 0.0211)	(0.0212, 0.0216)
Simulation + Plastics + NN + ReLu	(0.0367, 0.0367)	(0.0067, 0.0069)
Simulation + Plastics + BNN + ReLu	(0.0072, 0.0073)	(0.0068, 0.0068)

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors, simulation data, ReLu activation.

As expected, the computational cost for the NSGA-II and NSGA-III algorithms is high. Especially, NSGA-III in combination with a BNN is computationally intensive. Due to limited computation power, NSGA-II and NSGA-III algorithms are executed for 10 generations and a population of 12 chromosomes. Because networks start to converge around 500 epochs using ReLu activation, the network trained inside the NSGA algorithms is trained for 500 epochs. Tables 7 and 8 show all Pareto optimal outcomes for NSGA-II in combination with a NN and a BNN, respectively. In exchange for higher computational times, MAE values are lower when using a BNN than a NN.[12] Moreover, when using a BNN the expected trade-off between the environmental impact and the cost impact, as explained in Section 3.2, is confirmed.[13] If including all materials and using a BNN, then from an environmental perspective there is no clear preference for one of the materials, while in combination with a NN the algorithm tells that paper (31.4%) and LDPE (20.4%) should be prioritized. Note that in fact, the single-objective optimization that minimizes the environmental footprint (scenario Ia) is not a single-objective optimization because also the performance measures (train MAE and test MAE) are minimised. If including all materials and using a BNN [NN], then from a cost perspective RPET (26.8%) and steel (23.1%) [RPET (18.7%) and steel (22.1%)] should be prioritized. If including all materials, using a BNN and taking both the environment and costs into account, then there is no clear preference for one of the materials. However, LDPE (4.8%) and paper (5.6%) should be least prioritized. If a NN is used, then aluminium (27.8%) and RPET (19.2%) should be prioritized.

If including only plastics, in all three scenarios the constraint that the sum of RPET and RPP should be at least 15% is satisfied.[14] If including only plastics and using a BNN [NN], then from an environmental perspective PET (7.2%) and HDPE (10.4%) should be least prioritized [there is no clear preference for

[12] Appendix I shows that on the Unilever data BNN does not show an improvement compared to NN.
[13] As can be seen in Appendix I, this expected trade-off is not always confirmed when using the original Unilever data.
[14] When using the original Unilever data and including only plastics, then sometimes the algorithm had difficulties with satisfying the constraint of the sum of RPET and RPP being greater or equal to 15%.

one of the materials]. If including only plastics and using a BNN [NN], then from a cost perspective RPET (36.7%) [RPET (48.9%)] should be prioritized. If including only plastics, using a BNN [NN] and taking both the environment and costs into account, then HDPE (31.3%) [RPET (26.6%)] should be prioritized.

Table 7: Optimization results using NSGA-II with a NN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.02660	0.00187	-	-	0.03015	0.00193
	Costs (€/g)	-	-	0.54	0.25	0.31	1.74
	Aluminium (%)	3.9	-	8.2	-	**27.8**	-
	Glass (%)	6.8	-	11.5	-	2.7	-
	HDPE (%)	**2.0**	15.1	15.6	21.6	4.0	18.8
	LDPE (%)	**20.4**	14.5	3.6	**1.3**	**0.8**	14.6
	Paper (%)	**31.4**	-	5.5	-	5.1	-
	PET (%)	6.4	19.0	8.2	3.4	11.4	14.9
	PP (%)	7.6	19.6	**1.3**	5.9	7.4	13.0
	RPET (%)	7.0	19.4	**18.7**	**48.9**	19.2	26.6
	RPP (%)	3.8	12.5	5.3	18.9	16.1	12.1
	Steel (%)	10.7	-	22.1	-	5.4	-
	Time (sec.)	< 1000	< 2000	< 5	< 5	< 1000	< 2000
	MAE train	0.1549	0.0293	-	-	0.1571	0.0271
	MAE test	0.1558	0.0172	-	-	0.1585	0.0185

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

Table 8: Optimization results using NSGA-II with a BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.00259	0.00104	-	-	0.00262	0.00107
	Costs (€/g)	-	-	0.18	0.38	0.96	0.82
	Aluminium (%)	8.9	-	16.7	-	14.7	-
	Glass (%)	9.1	-	8.4	-	9.3	-
	HDPE (%)	8.3	10.4	11.9	**21.5**	9.0	**31.3**
	LDPE (%)	10.3	18.6	**0.9**	7.7	4.8	6.2
	Paper (%)	11.4	-	**0.2**	-	5.6	-
	PET (%)	8.4	7.2	3.4	5.7	8.7	11.5
	PP (%)	14.1	23.1	3.9	8.3	6.5	8.4
	RPET (%)	6.9	20.4	**26.8**	**36.7**	14.0	22.1
	RPP (%)	10.8	20.4	4.6	**20.2**	12.5	20.5
	Steel (%)	11.9	-	23.1	-	14.9	-
	Time (sec.)	< 3500	< 6000	< 5	< 5	< 3500	< 6000
	MAE train	0.0237	0.0077	-	-	0.0271	0.0073
	MAE test	0.0240	0.0077	-	-	0.0274	0.0074

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors, simulation data, ReLu activation.

5.3 NSGA-III

Tables 18 and 19 show all Pareto optimal outcomes for NSGA-III in combination with a NN and a BNN, respectively. Compared to Section 5.2, the expected trade-off between the environmental impact and the cost impact as explained in Section 3.2 is not always confirmed when using a BNN. It is confirmed for all cases but not when only costs are minimized and all materials are included. Again, performance measures (MAE train and MAE loss) are lower when using a BNN than when using a NN in all cases.

If including all materials and taking into account the environment impact, the costs impact or both impacts and using a NN or BNN does not show a clear preference or prioritization for one of the materials.

If including only plastics, in all three scenarios the constraint that the sum of RPET and RPP should be at least 15% is satisfied. If including only plastics and using a BNN [NN], then from an environmental perspective PP (23.2%), PET (20.7%) and LDPE (20.7%) [PP (17.5%) and RPP (17.4%)] should be prioritized. If including only plastics and using a BNN [NN], then from a cost perspective HDPE (26.0%) and PP (19.3%) should be prioritized [there is no clear prioritization but LDPE (5.2%) is less prioritized than others]. If including only plastics, using a BNN or NN and taking both the environment and costs into account, then there is no clear preference or prioritization for one of the materials. However, when using NN LDPE (7.3%) is less prioritized than other materials.

Table 9: Optimization results using NSGA-III with a NN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.06162	0.00203	-	-	0.05417	0.00200
	Costs (€/g)	-	-	0.94	0.95	0.79	1.34
	Aluminium (%)	9.3	-	10.5	-	8.0	-
	Glass (%)	10.3	-	14.7	-	13.1	-
	HDPE (%)	9.5	21.2	9.5	18.0	11.4	18.9
	LDPE (%)	13.1	12.5	5.3	**5.2**	5.8	**7.3**
	Paper (%)	9.9	-	4.8	-	6.4	-
	PET (%)	8.7	15.9	12.8	19.2	10.6	20.0
	PP (%)	7.5	17.5	9.5	19.5	7.3	17.1
	RPET (%)	11.0	15.6	10.3	18.3	15.0	19.2
	RPP (%)	10.4	17.4	10.4	19.9	10.5	17.6
	Steel (%)	10.3	-	12.2	-	11.7	-
	Time (sec.)	< 2000	< 3500	< 5	< 5	< 2000	< 3500
	MAE train	0.1682	0.0263	-	-	0.1661	0.0268
	MAE test	0.1710	0.0203	-	-	0.1687	0.0200

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

Table 10: Optimization results using NSGA-III with a BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.00261	0.00105	-	-	0.00304	0.00106
	Costs (€/g)	-		1.04	1.23	0.76	1.62
	Aluminium (%)	10.3	-	14.2	-	15.7	-
	Glass (%)	**5.6**	-	9.9	-	10.9	-
	HDPE (%)	13.4	16.5	11.7	**26.0**	6.4	17.2
	LDPE (%)	10.7	20.7	6.8	13.2	4.9	16.1
	Paper (%)	8.5	-	8.7	-	4.1	-
	PET (%)	10.3	20.7	8.0	14.8	8.8	17.7
	PP (%)	12.5	23.2	11.6	19.3	10.6	21.1
	RPET (%)	10.3	12.7	6.9	15.2	8.2	14.0
	RPP (%)	8.6	**6.3**	12.6	11.5	16.0	14.0
	Steel (%)	9.8	-	9.7	-	14.4	-
	Time (sec.)	< 6500	> 9000	< 5	< 5	< 6500	> 9000
	MAE train	0.0284	0.0079	-	-	0.0248	0.0076
	MAE test	0.0288	0.0079	-	-	0.0256	0.0076

Settings: learning rate = 0.0005, epochs = 10.000, BNNs are trained using standard normally distributed priors, simulation data, ReLu activation.

5.4 Limitations

This research has important limitations, which could cause potential uncertainties in the results. To compare the hypothetical material fractions predicted by the networks with actual outcomes, results of the NSGA algorithms combined with NNs or BNNs could be compared to results of the NSGA algorithms without neural networks. The latter case could be seen as the potential underlying data generating process (DGP). It should be emphasised that in this study this relation is not the true underlying DGP and thus a limitation of this study is that material fractions could not be compared to true optimal material fractions. The prediction of the NN inside the NSGA algorithms is based on the material fractions, the emissions and the technical recyclability rate, while for the potential DGP it is solely based on the material fractions and the environmental footprint for every material. This relation is presumably more complex and requires more advanced data. However, for research purposes it might be interesting to make this comparison. Results with respect to this potential DGP can be found in Appendix J.

Due to computational power, this study is limited in the number of generations and chromosomes in the NSGA algorithms. These hyperparameters should preferably be increased. Limitations related to the used method are relying solely on neural networks instead of using multiple machine learning methods and using only Gaussian Normally distributed priors in the BNN. Limitations related to the data are the little amount of observations of the original Unilever data set, assuming fixed costs that do not change with seasonality or other factors, emission and environmental footprint data that is not from Unilever but an external database and missing data with respect to the process energy GHG emissions of RPP and RPET. Unilever should extend its data set with more advanced numerical data of better quality.

Confounding problems could arise due to the selection or exclusion of environmental variables and/or differences in natural circumstances or protocol differences when measuring environmental variables. Then, the data is manipulated by another source that changes as well, e.g. temporal differences, seasonal differences, location differences. Conditioning on these covariates conditional unconfoundedness can be assumed (Deng, 2021). Also, randomization in generating samples could account for differences in natural circumstances and ensures independence among samples (Wiens & Parker, 1995). In general for causal inference randomization is important and is in this study incorporated in several ways, e.g. sufficiently large and randomized simulated data set and random initialization in the neural network.

6 Conclusion

As mentioned in Section 1, this study has three research objectives, which now can be answered:

1. By creating this methodology to benchmark packaging of Unilever's products regarding sustainability, Unilever is one step closer to achieving their sustainability packaging manifesto. This research provides Unilever a primary multi-objective framework to build further on by extending it with higher quality data and more objectives to determine an explicit strategy to navigate towards and achieve their Sustainable Living Plan (Unilever, 2021). To get a more realistic situation objectives regarding legalisation and laws could be added, since some materials have stricter rules than others, e.g. plastics have stricter laws than paper. To make the connection with food, objectives

maximizing shelf-life could be added because some (sustainable) materials have a declining effect on shelf-life. The interpretation of results should be taken with caution because the original data set of Unilever counts little observations.

2. The prediction model based on NN and BNN in this study found that the generalization performance is lower using a BNN than a NN. If all materials are included and the network is trained with ReLu activation for 10.000 epochs, the MAE (RMSE) on the test set equals 0.153 (0.081) and 0.021 (0.026) for NN and BNN, respectively. If only plastics are included and the network is trained with ReLu activation for 10.000 epochs, the MAE on the test set equals 0.037 (0.035) and 0.007 (0.009) for NN and BNN, respectively.

3. This study finds that the multi-objective optimization with respect to all materials (only plastics) using a NN in combination with NSGA-II gives a training MAE and test MAE of 0.157 and 0.159 (0.027 and 0.019), respectively, and in combination with NSGA-III gives a training MAE and test MAE of 0.166 and 0.169 (0.027 and 0.020), respectively. A multi-objective optimization with respect to all materials (only plastics) using a BNN, with standard normally distributed priors, in combination with NSGA-II gives a train MAE and test MAE of 0.027 and 0.027 (0.007 and 0.007), respectively, and in combination with NSGA-III gives a train MAE and test MAE of 0.025 and 0.026 (0.008 and 0.008), respectively. Overall, findings of this study show that BNN improves the prediction error compared to NN in exchange for computational time, while NSGA-III does not improve it compared to NSGA-II. Based on the multi-objective optimization with respect to all materials using a BNN in combination with NSGA-II, the most optimal environmental footprint and packaging costs equal 0.0026 points per gram of packaging and 0.96 euros per gram of packaging. In this case aluminium should be prioritized the most and LDPE should be prioritized the least. If only plastics are included, then the optimized values equal 0.0011 points per gram of packaging and 0.82 euros per gram of packaging. Then, RPET should be prioritized the most. Among the other plastics, there is no clear preference.

To extend literature future research could focus more on improving NSGA algorithms in combination with BNNs because it is found that it is beneficial for the accuracy of the network. Computational times of NSGA algorithms are high and therefore this could be an entrance for future research, e.g. by incorporating a tool such as preliminary dropout of individuals in the population and/or early stopping. To improve the overall algorithm and method, it might be interesting to use Adam optimizer to overcome the intensive computational burden, to consider Unified NSGA-III of Seada and Deb (2014) and/or to extend the method of Blundell et al. (2015) by trying a wider set of priors instead of only Gaussian Normal priors. Going beyond the scope of this research, another option would be to use other machine learning methods, e.g. random forest, in combination with NSGA.

References

[1] Ali Abdoli, M., Falah Nezhad, M., Salehi Sede, R., & Behboudian, S. (2012). *Longterm forecasting of solid waste generation by the artificial neural networks.* Environmental Progress Sustainable Energy, 31(4), 628-636.

[2] Afshari, H., Hare, W., & Tesfamariam, S. (2019). *Constrained multi-objective optimization algorithms: Review and comparison with application in reinforced concrete structures.* Applied Soft Computing, 83, 105631.

[3] Asadi, E., Da Silva, M. G., Antunes, C. H., & Dias, L. (2012). *Multi-objective optimization for building retrofit strategies: A model and an application.* Energy and Buildings, 44, 81-87.

[4] Azari, R., Garshasbi, S., Amini, P., Rashed-Ali, H., & Mohammadi, Y. (2016). *Multi-objective optimization of building envelope design for life cycle environmental performance.* Energy and Buildings, 126, 524-534.

[5] Azarmi, S. L., Oladipo, A. A., Vaziri, R., & Alipour, H. (2018). *Comparative modelling and artificial neural network inspired prediction of waste generation rates of hospitality industry: the case of North Cyprus.* Sustainability, 10(9), 2965.

[6] Bayes, T. (1763). *An essay towards solving a problem in the doctrine of chances.* By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S. Philosophical transactions of the Royal Society of London, (53), 370-418.

[7] Bezazi, A., Pierce, S. G., & Worden, K. (2007). *Fatigue life prediction of sandwich composite materials under flexural tests using a Bayesian trained artificial neural network.* International Journal of Fatigue, 29(4), 738-747.

[8] Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015). *Weight uncertainty in neural network.* In International conference on machine learning (pp. 1613-1622). PMLR.

[9] Chai, T. & Draxler, R. (2014). *Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature.* Geoscientific model development, 7(3), 1247-1250.

[10] Ciro, G. C., Dugardin, F., Yalaoui, F. & Kelly, R. (2016). *A NSGA-II and NSGA-III comparison for solving an open shop scheduling problem with resource constraints.* IFAC-PapersOnLine, 49(12), 1272-1277.

[11] Containers, P. & Good, N. D. (2016). *Documentation for greenhouse gas emission and energy factors used in the waste reduction model (WARM).*

[12] Dantas, A. T. A., Leite, M. B., & de Jesus Nagahama, K. (2013). *Prediction of compressive strength of concrete containing construction and demolition waste using artificial neural networks.* Construction and Building Materials, 38, 717-722.

[13] Das, I. & Dennis, J. E. (1998). *Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems.* SIAM journal on optimization, 8(3), 631-657.

[14] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). *A fast and elitist multiobjective genetic algorithm: NSGA-II.* IEEE transactions on evolutionary computation, 6(2), 182-197.

[15] Deb, K. & Jain, H. (2013). *An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints.* IEEE transactions on evolutionary computation, 18(4), 577-601.

[16] Jain, H. & Deb, K. (2013). *An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach.* IEEE Transactions on evolutionary computation, 18(4), 602-622.

[17] Delft University of Technology (2022). Idemat [Data file]. Retrieved February 2022, from: *https : //www.ecocostsvalue.com/data/*

[18] Deng, A. (2021). *Causal Inference and Its Applications in Online Industry.* Retrieved July 2022, from: *https : //alexdeng.github.io/causal/rcm.html*

[19] Esfandiari, K., Ghoreyshi, A.A. & Jahanshahi M. (2017). *Using artificial neural network and ideal adsorbed solution theory for predicting the CO2/CH4 selectivities of metal- organic frameworks: a comparative study.* Ind. Eng. Chem. Res. 56 14610–14622

[20] Falbel, D. & Luraschi, J. (2021). *Package "torch": Tensors and Neural Networks with'GPU'Acceleration.* Retrieved May 2022, from: *https : //cran.r − project.org/web/packages/torch/torch.pdf*

[21] Foote K. D. (2021). *A Brief History of Neural Networks.* Retrieved February 2022, from: *https : //www.dataversity.net/a − brief − history − of − neural − networks/*

[22] Fortuin, V., Garriga-Alonso, A., Wenzel, F., Rätsch, G., Turner, R., van der Wilk, M. & Aitchison, L. (2021). *Bayesian neural network priors revisited.* arXiv preprint arXiv:2102.06571.

[23] Goodfellow, I., Bengio, Y., & Courville, A. (2017). *Deep learning (adaptive computation and machine learning series).* Cambridge Massachusetts, 321-359.

[24] Graves, A. (2011). *Practical variational inference for neural networks.* Advances in neural information processing systems, 24.

[25] Han, J. & Moraga, C. (1995). *The influence of the sigmoid function parameters on the speed of back-propagation learning.* In International workshop on artificial neural networks (pp. 195-201). Springer, Berlin, Heidelberg.

[26] Hoffman, M. D., Blei, D. M., Wang, C., & Paisley, J. (2013). *Stochastic variational inference.* Journal of Machine Learning Research.

[27] Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* MIT press.

[28] Hastie, T., Tibshirani, R. & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2, pp. 1-758). New York: springer.

[29] Hornik, K., Stinchcombe, M., & White, H. (1989). *Multilayer feedforward networks are universal approximators.* Neural networks, 2(5), 359-366.

[30] Irawan, D. (2020). *Package "MaOEA": Many Objective Evolutionary Algorithm.* Retrieved May 2022, from: $https://cran.r-project.org/web/packages/MaOEA/MaOEA.pdf$

Ishibuchi Ishibuchi, H., Imada, R., Setoguchi, Y. & Nojima, Y. (2016). *Performance comparison of NSGA-II and NSGA-III on various many-objective test problems.* In 2016 IEEE Congress on Evolutionary Computation (CEC) (pp. 3045-3052). IEEE.

[31] International Organization for Standardization (2006). *Environmental management: life cycle assessment*; Principles and Framework. ISO.

[32] Izmailov, P., Vikram, S., Hoffman, M. D., & Wilson, A. G. G. (2021). *What are Bayesian neural network posteriors really like?.* In International Conference on Machine Learning (pp. 4629-4640). PMLR.

[33] Jospin, L. V., Buntine, W., Boussaid, F., Laga, H., & Bennamoun, M. (2020). *Hands-on Bayesian neural networks–a tutorial for deep learning users.* arXiv preprint arXiv:2007.06823.

[34] Kingma, D. P., & Welling, M. (2013). *Auto-encoding variational bayes.* arXiv preprint arXiv:1312.6114.

[35] Kissinger, M., Sussmann, C., Moore, J. & Rees, W. (2013). *Accounting for greenhouse gas emissions of materials at the urban scale-relating existing process life cycle assessment studies to urban material and waste composition.*

[36] Kochurov, M. & Wiecki, T. (2017). *Variational Inference: Bayesian Neural Networks.* Retrieved May 2022, from: $https://docs.pymc.io/en/v3/pymc-examples/examples/variational_inference/bayesian_neural_network_advi.html$

[37] Konak, A., Coit, D. W., & Smith, A. E. (2006). *Multi-objective optimization using genetic algorithms: A tutorial.* Reliability engineering system safety, 91(9), 992-1007.

[38] Kullback, S. & Leibler, R. A. (1951). *On information and sufficiency.* The annals of mathematical statistics, 22(1), 79-86.

[39] Liu, D. (2017). *A Practical Guide to ReLU.* Medium. Retrieved March 2022, from: $https://medium.com/@danqing/a-practical-guide-to-relu-b83ca804f1f7$

[40] Magnier, L., & Haghighat, F. (2010). *Multiobjective optimization of building design using TRNSYS simulations, genetic algorithm, and Artificial Neural Network.* Building and Environment, 45(3), 739-746.

[41] Mahjoubi, S., Barhemat, R., Guo, P., Meng, W., & Bao, Y. (2021). *Prediction and multi-objective optimization of mechanical, economical, and environmental properties for strain-hardening cementitious composites (SHCC) based on automated machine learning and metaheuristic algorithms.* Journal of Cleaner Production, 329, 129665.

[42] Mersmann, O., Trautmann, H., Steuer, D., Bischl, B. & Deb, K. (2020). *Package "mco": multiple criteria optimization algorithms and related functions.* Retrieved May 2022, from: $https://cran.r-project.org/web/packages/mco/mco.pdf$

[43] Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 25). San Francisco, CA, USA: Determination press.

[44] Levy, K. Y. (2016). *The power of normalization: Faster evasion of saddle points.* arXiv preprint arXiv:1611.04831.

[45] Murata, T., & Ishibuchi, H. (1995). *MOGA: multi-objective genetic algorithms.* In IEEE international conference on evolutionary computation (Vol. 1, pp. 289-294). Piscataway, NJ, USA: IEEE.

[46] Nielsen, M. A. (2015). *Neural networks and deep learning (Vol. 25).* San Francisco, CA, USA: Determination press.

[47] Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2018). *Activation functions: Comparison of trends in practice and research for deep learning.* arXiv preprint arXiv:1811.03378.

[48] Penm, J., Chaar, B., Moles, R., & Penm, J. (2013). *Predicting ASX Health Care Stock Index Movements After the Recent Financial Crisis Using Patterned Neural Networks.* In Rethinking Valuation and Pricing Models (pp. 599-610). Academic Press.

[49] Qinghui Yu, J., Creager, E., Duvenaud, D. & Bettencourt, J. (n.d.). *Bayesian Neural Networks.* Retrieved May 2022, from: $https://www.cs.toronto.edu/\ duvenaud/distill_bayes_net/public/$

[50] Seada, H. & Deb, K. (2014). *U-NSGA-III: A unified evolutionary algorithm for single, multiple, and many-objective optimization.* COIN report, 2014022.

[51] Si, B., Wang, J., Yao, X., Shi, X., Jin, X., & Zhou, X. (2019). *Multi-objective optimization design of a complex building based on an artificial neural network and performance evaluation of algorithms.* Advanced Engineering Informatics, 40, 93-109.

[52] Silvestro, D. & Andermann, T. (2020). *Prior choice affects ability of Bayesian neural networks to identify unknowns.* arXiv preprint arXiv:2005.04987.

[53] Siripatrawan, U. & Jantawat, P. (2008). *A novel method for shelf life prediction of a packaged moisture sensitive snack using multilayer perceptron neural network.* Expert Systems with Applications, 34(2), 1562-1567.

[54] Soleimani, R., Shoushtari, N. A., Mirza, B., & Salahi, A. (2013). *Experimental investigation, modeling and optimization of membrane separation using artificial neural network and multi-objective optimization using genetic algorithm.* Chemical engineering research and design, 91(5), 883-903.

[55] Somkuwar, A. K., Khaira, H. K., & Somkuwar, V. (2010). *Materials selection for product design using artificial neural network technique.* Journal of Engineering, Science and Management Education, 51-54.

[56] Stoica, M., Antohi, V. M., Zlati, M. L., & Stoica, D. (2020). *The financial impact of replacing plastic packaging by biodegradable biopolymers - a smart solution for the food industry.* Journal of Cleaner Production, 277, 124013.

[57] Taddy, M. (2019). *The technological elements of artificial intelligence (p. 61).* Chicago, IL: University of Chicago Press.

[58] Unilever (2020). *Unilever makes progress on its sustainable packaging goals.* Retrieved February, 2022 from: $https://www.unilever.com/news/press-and-media/press-releases/2020/unilever-makes-progress-on-its-sustainable-packaging-goals/$

[59] Unilever (2021). *Unilever Sustainable Living Plan 2010 to 2020.* Retrieved February, 2022 from: $https://assets.unilever.com/files/92ui5egz/production/287881e6e4572af1bc2a1d3c97e3b4abd4e57ea1.pdf/USLP-summary-of-10-years-progress.pdf$

[60] Unilever (2022). Packaging portfolio. [Data file]. Wageningen: Unilever.

[61] Unilever (2022). Food database. [Data file]. Wageningen: Unilever.

[62] Wiens, J. A. & Parker, K. R. (1995). *Analyzing the effects of accidental environmental impacts: approaches and assumptions.* Ecological Applications, 5(4), 1069-1083.

[63] Wimarshana, B., Ryu, J., & Choi, H. J. (2014). *Neural network based material models with Bayesian framework for integrated materials and product design.* International journal of precision engineering and manufacturing, 15(1), 75-81.

[64] Yang, X. S., Chien, S. F. & Ting, T. O. (2015). *Bio-inspired computation in telecommunications.* Morgan Kaufmann.

[65] Yannibelli, V., Pacini, E., Monge, D., Mateos, C., & Rodriguez, G. (2020). *A comparative analysis of NSGA-II and NSGA-III for autoscaling parameter sweep experiments in the cloud.* Scientific Programming, 2020.

[66] Zhang, J., Huang, Y., Wang, Y., & Ma, G. (2020). *Multi-objective optimization of concrete mixture proportions using machine learning and metaheuristic algorithms.* Construction and Building Materials, 253, 119208.

[67] Zhang, H., & Cui, Y. (2019). *A model combining a Bayesian network with a modified genetic algorithm for green supplier selection.* Simulation, 95(12), 1165-1183.

[68] Zhou, C. C., Yin, G. F., & Hu, X. B. (2009). *Multi-objective optimization of material selection for sustainable products: artificial neural networks and genetic algorithm approach.* Materials Design, 30(4), 1209-1215.

A Nomenclature

Table 11: Nomenclature including abbreviations and symbols.

Variables	Definition
Abbreviations:	
BBB	Bayes-By-Backprop
BNN	Bayesian Neural Network
BP	Back-Propagation
CD	Crowding Distance
DGP	Data Generating Process
ES	Evolution Strategy
GA	Genetic Algorithm
GHG	Green House Gases
HDPE	High Density Polyetheen
ISO	International Organization for Standardization
KL	Kullback Leibler
LDPE	Low Density Polyetheen
MAE	Mean absolute error
MOGA	Multi-Objective Genetic Algorithm
MOPSO	Multi-Objective Particular Swarm Optimization
MOSA	Multi-Objective Simulated Annealing
LCA	Life Cycle Assessment
NN	Neural Network
NSGA	Non-dominated Sorting Genetic Algorithm
PET	Polyethyleentereftalaat (virgin)
PP	Polypropyleen (virgin)
ReLu	Rectified Linear Unit
RMSE	Root mean squared error
RPET	Recycled PET
RPP	Recycled PP
SGD	Stochastic Gradient Descent
SVI	Stochastic Variational Inference
UNSGA	Unified Non-dominated Sorting Genetic Algorithm
VI	Variational Inference
Symbols:	
z_{LCA}	Environmental footprint in points/gram
z_{cost}	Life-cycle costs in euros/gram
m_k	Proportion of material k of the total package in percentages
m_1	Proportion of aluminium of the total package in percentages

m_2	Proportion of glass of the total package in percentages
m_3	Proportion of HDPE of the total package in percentages
m_4	Proportion of LDPE of the total package in percentages
m_5	Proportion of paper of the total package in percentages
m_6	Proportion of virgin PET of the total package in percentages
m_7	Proportion of virgin PP of the total package in percentages
m_8	Proportion of recycled PET of the total package in percentages
m_9	Proportion of recycled PP of the total package in percentages
m_{10}	Proportion of steel of the total package in percentages
M	Total number of materials of the package equal to 10
λ_k	Dummy variable that indicates if material k is included or excluded in the problem.
$z_{k,LCA}$	Total environmental footprint of material k in points per kilogram
$z_{k,costs}$	Life-cycle costs for material k of the package
x_k	Raw value of input k of the NN
x_k^{norm}	Normalized value of output k of the NN
y_k	Actual value of output k of the NN
$\hat{y}_k$	Predicted value of output k of the NN
$\bar{y}$	Mean of actual values $y_1, \ldots, y_N$
L	Number of hidden layers in the NN
δ_a	Activation function a
w_{ki}	Weight for the normalized input x_i^{norm} at node k
d	Total number of inputs
t	Total number of outputs
N	Total number of data points
b_k	Bias at node k
$\boldsymbol{\theta}$	Parameter vector
$\mathcal{D}$	Training data set, $\mathcal{D} = (\mathbf{x}, \mathbf{y})$
$\mathcal{M}$	Model
$\boldsymbol{\mu}$	Vector of mean values of Gaussian Normal distribution
$\boldsymbol{\sigma}$	Vector of standard deviations of Gaussian Normal distribution
$p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})$	Likelihood
$p(\boldsymbol{\theta})$	Prior distribution
$p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})$	Posterior distribution
$q(\boldsymbol{\theta} \mid \boldsymbol{\eta})$	Variational distribution
$VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$	Variational free energy
$\boldsymbol{\eta}$	Distribution parameters of variational distribution
I	Identity matrix
$\boldsymbol{\epsilon}$	Error term
TE	Total number of epochs
α	Learning rate

∇	Gradient
$\odot$	Element-wise multiplication
g	gth generation
C	Total number of chromosomes in population
g	Generation g
P_g	Population in generation g
FI	Fitness score
Q_g	Offsprings in generation g
P^c	Cross-over probability
P^m	Mutation probability
G	Maximum number of generations
R_g	Merged population of P_g and Q_g
F_j	Front j
R	Maximum number of fronts
F_L	Last front to be included
S	Survival population
H	Total number of reference points
Z^s	H structured reference points
Z_r	Reference set
r	Reference point in reference set Z_r
z_k	Objective function k
s	Chromosome s in survival population S
π_s	Closest reference point s
Δ	Distance between s and $\pi(s)$
l	Reference line

B VI and KL divergence

This Appendix provides detailed information regarding the VI methods and KL divergence. The complete derivation of how to write out the KL divergence given in (15) in Section 3.3.2 is as follows:

$$
\begin{aligned}
&KL\big[q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \mid\mid p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})\big] \\
&= \int q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \log\left(\frac{q(\boldsymbol{\theta} \mid \boldsymbol{\eta})}{p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})}\right) d\boldsymbol{\eta} \\
&= \mathbb{E}\left[\log\left(\frac{q(\boldsymbol{\theta} \mid \boldsymbol{\eta})}{p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})}\right)\right] (\textit{expectation with respect to } q(\boldsymbol{\theta} \mid \boldsymbol{\eta})) \\
&= \mathbb{E}\left[\log\left(\frac{q(\boldsymbol{\theta} \mid \boldsymbol{\eta})}{p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M}) p(\boldsymbol{\theta} \mid \mathcal{M})} p(\mathcal{D} \mid \mathcal{M})\right)\right] (\textit{using conditional probability rules}) \\
&= \mathbb{E}\Big[\log(q(\boldsymbol{\theta} \mid \boldsymbol{\eta})) - \log(p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})) - \textit{log}(p(\boldsymbol{\theta} \mid \mathcal{M})) + \log(p(\mathcal{D} \mid \mathcal{M}))\Big] (\textit{writing out the logarithm}) \\
&= \mathbb{E}\Big[\log(q(\boldsymbol{\theta} \mid \boldsymbol{\eta})) - \log(p(\mathcal{D} \mid \boldsymbol{\theta}, \mathcal{M})) - \textit{log}(p(\boldsymbol{\theta} \mid \mathcal{M}))\Big] + \log(p(\mathcal{D} \mid \mathcal{M}) (\textit{the last logarithm does not depend on } \theta) \\
&= VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) + \log(p(\mathcal{D} \mid \mathcal{M}))
\end{aligned}
\tag{20}
$$

The negative value of the variational free energy $VFE(\boldsymbol{\theta}, \boldsymbol{\eta})$ can also be seen as the Evidence Lower Bound, $ELBO(\boldsymbol{\theta}, \boldsymbol{\eta})$. Since $KL\big[q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \mid\mid p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})\big] \geq 0$, $-VFE(\boldsymbol{\theta}, \boldsymbol{\eta}) \leq \log(p(\mathcal{D} \mid \mathcal{M}))$ and therefore the KL divergence can be written as:

$$
KL\big[q(\boldsymbol{\theta} \mid \boldsymbol{\eta}) \mid\mid p(\boldsymbol{\theta} \mid \mathcal{D}, \mathcal{M})\big] = -ELBO(\boldsymbol{\theta}, \boldsymbol{\eta}) + \log(p(\mathcal{D} \mid \mathcal{M}) \tag{21}
$$

Therefore, minimizing (17) of Section 3.3.2 as in (16) of Section 3.3.2 is similar to maximizing the ELBO:

$$
\boldsymbol{\eta}^{\star} = \underset{\boldsymbol{\eta} \in \Omega}{\operatorname{argmax}} \; ELBO(\boldsymbol{\theta}, \boldsymbol{\eta}) \tag{22}
$$

C Operators in GAs

This Appendix provides detailed information regarding the cross-over and mutation operators, which are used to create offsprings, and is based on Yannibelli et al. (2020).

C.1 Cross-over operator in GAs

In the (single-point) cross-over stage, two chromosomes from the population in generation g, P_g, are selected based on their fitness score FI. The higher FI, the more likely they are selected. Then, in generation g offsprings $Q = [Q_1, Q_2]$ are created from parents $Parent = [Parent_1, Parent_2]$ by interchanging a specific part of the chromosomes based on a cross-over probability P^c:

$$Q_{1c} = 0.5\Big((1 + P_c^c)Parent_{1c} + (1 - P_c^c)Parent_{2c}\Big) \tag{23}$$

$$Q_{2c} = 0.5\Big((1 - P_c^c)Parent_{1c} + (1 + P_c^c)Parent_{2c}\Big), \tag{24}$$

where lowercase c equals the cth locus of the parent or offspring chromosome, i.e. gene c of the chromosome. The cross-over probability P^c follows a polynomial probability distribution according to:

$$P_c^c = \begin{cases} (2u_c)^{(1/(Dc+1))} & \text{if } u_c \leq 0.5 \\ \left(\frac{1}{2(1-u_c)}\right)^{(1/(Dc+1))} & \text{if } u_c > 0.5 \end{cases}, \tag{25}$$

where u_c is a random number generated on the interval $[0, 1]$ and Dc equals the index of the cross-over distribution, which is a predetermined non-negative real number. The higher (lower) Dc, the higher the probability that the offspring chromosomes are close to (more different from) the parent chromosomes.

C.2 Mutation operator in GAs

In the mutation stage, one gene in a chromosome at a specific locus of each off-spring is changed by some probability P^m:

$$Q_{1c} = \begin{cases} Parent_{1c} + P_c^m(Parent_{1c} - L_c) & \text{if. } u_c \leq 0.5 \\ Parent_{1c} + P_c^m(U_c - Parent_{1c}) & \text{if. } u_c > 0.5 \end{cases} \tag{26}$$

$$Q_{2c} = \begin{cases} Parent_{2c} + P_c^m(Parent_{2c} - L_c) & \text{if. } u_c \leq 0.5 \\ Parent_{2c} + P_c^m(U_c - Parent_{2c}) & \text{if. } u_c > 0.5 \end{cases}, \tag{27}$$

where again lowercase c equals gene c of the chromosome, L_c equals the lower-bound of locus c of the chromosome and U_c equals the upper-bound of locus c of the chromosome. L_c and U_c ensure that the mutation operator creates values inside the bounds. The mutation probability P_c^m follows a polynomial probability distribution according to:

$$P_c^c = \begin{cases} (2u_c)^{(1/(Dm+1))} - 1 & \text{if } u_c \leq 0.5 \\ 1 - \big(2(1-u_c)\big)^{(1/(Dm+1))} & \text{if } u_c > 0.5 \end{cases}, \tag{28}$$

where Dm equals the index of the mutation distribution, which is a predetermined non-negative real number.

D Crowding Distance

This Appendix provides detailed information regarding the Crowding Distance (CD) and is based on Konak et al. (2006). Obtaining the CD starts with ranking the population and identifying the non-dominated fronts $F_1, F_2, ..., F_R$ (by using the fast non-dominated sorting algorithm). Then, for each front $j = 1, \ldots, R$ the following steps are repeated:

1. For each objective function k, the chromosomes in F_j are sorted in ascending order. Define $\mathbf{x}_{F_j,k}$ as the vector containing all chromosomes in front F_j for objective function k.

2. Then, a CD value of ∞ is assigned to the minimum and maximum chromosomes in F_j: $CD(min(\mathbf{x}_{F_j,k})) = CD(max(\mathbf{x}_{F_j,k})) = \infty$

3. All other values except from $CD(min(\mathbf{x}_{F_j,k}))$ and $CD(max(\mathbf{x}_{F_j,k}))$ are valued according to:

$$CD(\mathbf{x}_{F_j,k}) = \frac{z_k(\mathbf{x}_{F_{j+1},k}) - z_k(\mathbf{x}_{F_{j-1},k})}{z_k^{max} - z_k^{min}}, \tag{29}$$

   where $z_k(\cdot)$ equals the value of objective function k, z_k^{max} and $z_k(\cdot)$ and z_k^{min} equal the maximum and minimum value of $z_k(\cdot)$ observed so far during the search, respectively.

4. To find the total CD of chromosomes $\mathbf{x}$, sum all CDs over all objectives: $CD(\mathbf{x}) = \sum_k CD_k(\mathbf{x})$

E Details regarding NSGA-III

This Appendix provides detailed information regarding NSGA-III and is based on (Deb & Jain (2013*a*) and Deb & Jain (2013*b*).

E.1 `Normalize` operator

The `Normalize` operator starts with finding the optimal point of survival population S by finding the minimal value of each objective function. Thereafter, each objective is transformed by subtracting the minimal value from each objective function. Then, the extreme points in each objective axis are distinguished and each transformed objective function $z_k''(\mathbf{s})$ is now an extreme objective vector z_k^{max}). The extreme objective vectors z_k^{max} are used to form the hyperplane. Finally, the intercept of each objective axis k with the hyperplane is computed and the objective functions are normalized. Algorithm 5 shows the pseudocode of this procedure.

Algorithm 5 `Normalize`(z_k, S, F_L, Z_r) operator.

Input: Normalized survival population S^{norm} and structured points Z^s

for $k = 1$ `to` $|\,\mathbf{z}\,|$ **do**
  `Compute optimal point:` $z_k^{min} = \min_{\mathbf{s}\in S} z_k(\mathbf{s})$
  `Transform objectives:` $z_k''(\mathbf{s}) = z_k(\mathbf{s}) - z_k^{min}$
  `Compute the extreme points:` r_k^{max}
end for

`Each transformed objective function` $z_k''(\mathbf{s})$ `is now an extreme objective vector` z_k^{max}.
`The hyperplane is formed by the extreme objective vector` $\mathbf{z}^{max}$.

for $k = 1$ `to` $|\,\mathbf{z}\,|$ **do**
  `Compute intercepts` $inter_k$ `of the` k`th objective axis and the hyperplane.`
end for
for $k = 1$ `to` $|\,\mathbf{z}\,|$ **do**
  `Normalize objective functions:` $z_k^{norm} = \frac{z_k - z_k^{min}}{inter_k - z_k^{min}}$
end for
$Z_r^{norm} = Z^s$

`return` $[\mathbf{z}^{norm}, Z_r^{norm}]$

E.2 `Associate` operator

After normalization of each objective, for each reference point r in the normalized reference set Z_r^{norm} a reference line l is defined by connecting the reference point with the origin. Thereafter, every chromosome s in survival population S is associated with a reference point. Then, the perpendicular distance of each chromosome s in S from each reference line l is computed. Finally, the reference line l closest to the chromosome s in the normalized objective space is associated with the chromosome and returned. Algorithm 6 shows the pseudocode of this procedure.

Algorithm 6 `Associate`(S^{norm}, Z_r^{norm}) operator.

Input: Normalized reference set Z_r^{norm} and normalized survival population S^{norm}

for `each reference point` r `in` Z_r^{norm} **do**
    `Compute reference line` $\mathbf{l}$
end for
for `each chromosome` s `in` S^{norm} **do**
    for `each` $\mathbf{l}$ `in` Z_r^{norm} **do**
        `Compute` $\Delta^{\perp}(\mathbf{s}, \mathbf{l}) = \mathbf{s} - \frac{\mathbf{l}^T \mathbf{s}}{||\mathbf{l}||}$
    end for
end for

$\pi_{\mathbf{s}} = \arg\min_{l \in Z_r^{norm}} \ \Delta^{\perp}(\mathbf{s}, \mathbf{l})$
$\Delta(\mathbf{s}) = \Delta^{\perp}(\mathbf{s}, \pi_{\mathbf{s}})$

`return` $[\pi_{\mathbf{s}}, \Delta(\mathbf{s})]$

E.3 `Niching` operator

It might be the case that reference points are not associated with any chromosomes or do have multiple associations. The niche count ρ_r counts the number of chromosomes in P_{g+1} associated with each reference point.

First, the set with reference points that have minimum ρ_r is determined. If there are multiple reference points with minimum ρ_r, then one reference point is randomly chosen and denoted by $\tilde{r}$. If $\rho_{\tilde{r}=0}$, which means that no chromosome in P_{g+1} is associated with $\tilde{r}$, there are two possibilities for reference point $\tilde{r}$ in front F_L.

1. It could be that one or more chromosomes in F_L are already associated with $\tilde{r}$. Then, the reference point with the shortest perpendicular distance from the reference line is added to P_{g+1}. Then, the niche count $\rho_{\tilde{r}}$ is increased by one.

2. None of the chromosomes in F_L is associated with reference point $\tilde{r}$. Then, $\tilde{r}$ is not considered at all for the current generation.

If $\rho_{\tilde{r}} \geq 1$, meaning that one chromosome in P_{g+1} is already associated with reference point, a chromosome from F_L associated with reference point $\tilde{r}$ is randomly chosen and added to P_{g+1}. Then, the niche count $\rho_{\tilde{r}}$ is increased by one. In every iteration the niche counts are updated and the operation is repeated until P_{g+1} is completed with the missing number of chromosomes. Algorithm 7 shows the pseudocode of this procedure.

Algorithm 7 `Niching`$(\rho_r, \pi, \Delta, Z_r^{norm}, F_L, P_{g+1})$ operator.

Input: Niche count ρ_r, $\pi_{\mathbf{s}}$ (closest reference point s), $\Delta(s)$ (distance between s and $\pi_{\mathbf{s}}$), Normalized reference set Z_r^{norm} and F_L

while $P_{g+1} < C$ **do**
  $J^{min} = \arg\min_{r \in Z_r^{norm}} \rho_r$
  $I_{\tilde{r}} = \{\pi_{\mathbf{s}} = \tilde{r}, \mathbf{s} \in F_L\}$
  $\tilde{r} = rand(J^{min})$
  if $I_{\tilde{r}} \neq \emptyset$ **then**
    if $\rho_{\tilde{r}} = 0$ **then**
      $P_{g+1} = P_{g+1} \cup \left(\arg\min_{\mathbf{s} \in I_{\tilde{r}}} \Delta(\mathbf{s}) \right)$
    else
      $P_{g+1} = P_{g+1} \cup rand(I_{\tilde{r}})$
    end if
    $\rho_{\tilde{r}} = \rho_{\tilde{r}} + 1$
  else
    $Z_r^{norm} = Z_r^{norm} / \tilde{r}$
  end if
end while

`return` P_{g+1}

F Details regarding the data

This Appendix gives details regarding the data used in this research.

F.1 Data definitions and assumptions

Table 12 summarizes the details, assumptions and restrictions of the data set.

Table 12: Details regarding the used data.

Data	Unit	Source	Notes
Environmental footprint	Points per g of packaging	Idemat (Delft University of Technology, 2022)	
Life-cycle GHG emissions	CO2E in g	Kissinger, Sussmann, Moore & Rees (2013)	Life-cycle GHG emissions are emissions from cradle-to-gate, meaning that also transportation costs are taken into account. These transportation costs are not exactly equal to the transportation costs of Unilever's products. Moreover, the life-cycle emissions of PP are only based on one source and there is no difference made between virgin and recycled materials. CO2E is carbon-dioxide equivalent, which makes it possible to compare different GHG.
Packaging life-cycle costs of 2022	€ per g of packaging	Packaging Portfolio (Unilever, 2022a), Food database (Unilever, 2022b)	Specifically, this data is available for aluminium, glass, HDPE, LDPE, paper, virgin PP, recycled PP, virgin PET, recycled PET, steel. In this research solvents, ink, nylon, rubber and other plastics that have a small contribution to the total weight of the packaging are ignored. Also, it is assumed that the costs of the raw materials are fixed and do not depend on the quantity.
Percentage of material of total weight of package	%	Packaging Portfolio (Unilever, 2022a), Food database (Unilever, 2022b)	Specifically, this data is available for aluminium, glass, HDPE, LDPE, paper, virgin PP, recycled PP, virgin PET, recycled PET, steel. In this research solvents, ink, nylon, rubber and other plastics that have a small contribution to the total weight of the packaging are ignored.

Process Energy GHG Emissions	CO2E in g	Containers & Good (2016)	The process energy of recycled PP was not available in Containers & Good (2016). Therefore, it is approximated as follows: to get the process emissions of recycled PP, the process emissions of virgin PP is reduced by the average decrease in percentages for which PET is reduced to become recycled PET.
Technical recyclability rate	%	Packaging Portfolio (Unilever, 2022a), Food database (Unilever, 2022b)	

F.2 Simulation data set

Table 13 summarizes how the simulated data is generated.

Table 13: Computation of simulated data.

Data	Computation of simulated version
Environmental footprint	The environmental footprint of every package is calculated using the environmental footprint of every material of Delft University of Technology, 2022) and the randomly generated material proportions.
Life-cycle packaging costs	The life-cycle packaging costs are calculated using the average life-cycle packaging costs from the Packaging Portfolio (Unilever, 2022a) and the randomly generated material proportions.
Materials (aluminium, glass, HDPE, LDPE, paper, PET, PP, RPET, RPP and steel)	The material composition of each package is randomly generated between zero and hundred. It is made sure that the sum of all materials in every package equals hundred.
Technical recyclability rate	The technical recyclability rate is calculated by computing the proportion of RPET and RPP in every package.

Process emissions of a package	The process emissions of every package are calculated using the process emissions of every material of Containers & Good (2016) and the randomly generated material proportions.
Life-cycle emissions of a package	The life-cycle emissions of every package are calculated using the life-cycle emissions of every material of Kissinger, Sussmann, Moore & Rees (2013) and the randomly generated material proportions.

F.3 Descriptive statistics

Table 14 gives an overview of the distribution of packages and materials in the data set.

Table 14: Distribution of packaging types and materials in the Unilever data set.

Packaging type	Frequency (in %)	**Materials**	Frequency (in %)
Bag	13.3	Aluminium	0.8
Bottle	12.3	Glass	2.0
Bucket	9.7	Paper/Carton	22.6
Can	5.5	Plastic	69.4
Box	21.5	Steel	5.1
Container	18.8	Other	0.1
Jar	12.1		
Portion pack	3.8		
Other	3.0		

Table 15 shows the descriptive statistics of both the Unilever data set and the simulation data set.

Table 15: Range of variables.

	Unilever data set			**Simulation data set**		
Number of observations	4225			20.000		
	Min	Mean	Max	Min	Mean	Max
Dependent variables						
Environmental footprint (points/gram of packaging)	0	0.02	1.00	0.08	0.15	0.30
Life-cycle packaging costs (€/gram of packaging)	0	0.84	466.38	2.62	28.06	52.34
Independent variables and features						
Aluminium (%)	0	2.72	100	0	9.99	43.67
Glass (%)	0	2.24	99.73	0	9.96	34.69
HDPE (%)	0	1.38	100	0	9.97	35.83
LDPE (%)	0	6.87	100	0	9.95	35.63
Paper (%)	0	26.15	100	0	10.03	32.33
PET (%)	0	10.53	100	0	9.91	33.31
PP (%)	0	43.19	100	0	10.05	36.07
RPET (%)	0	0.05	32.24	0	10.06	41.08
RPP (%)	0	0.54	99.43	0	10.2	35.14
Steel (%)	0	6.32	100	0	10.06	42.89
Technical recyclability rate (%)	0	78.72	100	0.18	20.08	89.22
Process emissions of a package (CO2E in grams)	36.96	32073.69	414906	1.03	8398.00	51416.45
Life-cycle emissions of a package (CO2E in grams)	47.58	22621	585000	47.02	52574.08	227038.6

G Details regarding the code

This Appendix summarizes the details regarding the programming code conducted in R. The networks are constructed from scratch using the `torch` library (Falbel & Luraschi, 2021). The `mco` library (Mersmann, Trautmann, Steuer, Bischl & Deb, 2020) and the `MaOEA` library (Irawan, 2020) are used to solve the multi-objective optimization using NSGA-II and NSGA-III, respectively.

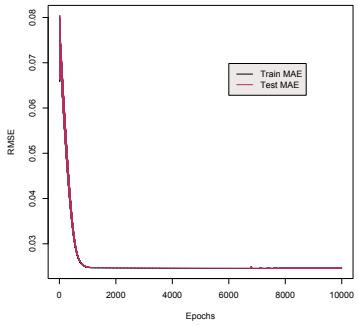
To optimize the learning process of the networks, in the R code exponential decay and step decay were also considered. This is not discussed in the paper, since it did not give a substantial improvement.

Due to confidentiality reasons and due to a small number of data points, the program code shared in Google Colab shows the code using a simulation data set based on Unilever's data of 20.000 observations and can be found using the following link: https://colab.research.google.com/drive/1HwC7E1npLGfKw7pJGPkWLL7ybZZRuNPc?usp=sharing. The original Rmarkdown code can be found using this link: https://drive.google.com/drive/folders/1PGeF8g7sdNVzHcLE4TdNYdOd5oGuxe41?usp=sharing

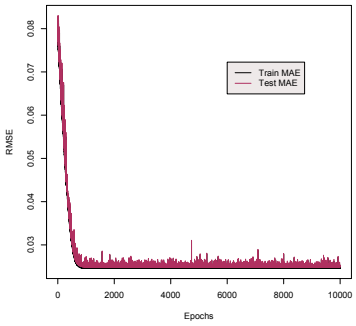
H RMSE loss

This Appendix provides the plots of the RMSE loss for the original Unilever data, the simulation data, ReLu activation function and sigmoid activation function.

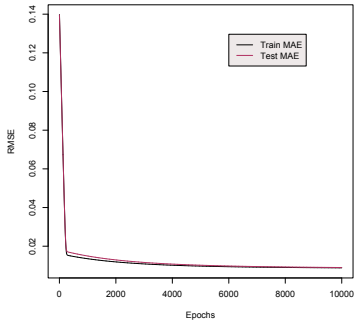
Figure 6: RMSE loss using simulation data, training for 10.000 epochs and in case of the BNN then priors follow a standard Normal distribution.



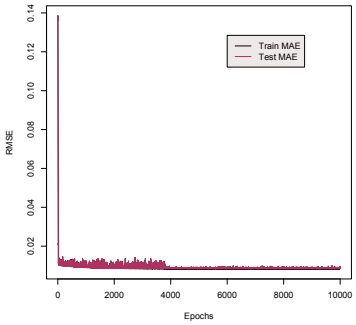
(a) All - NN - ReLu



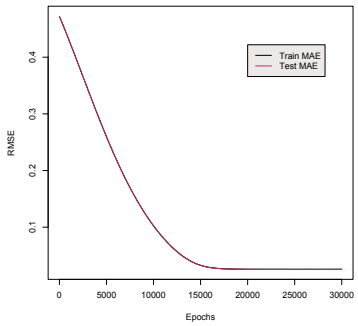
(b) All - BNN - ReLu



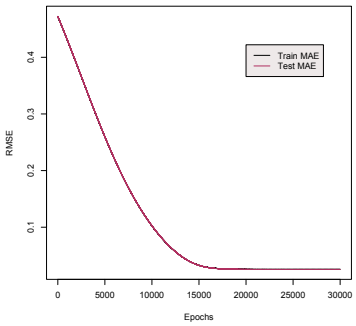
(c) Plastics - NN - ReLu



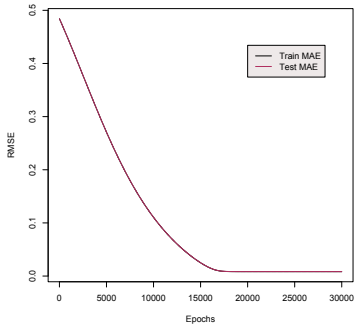
(d) Plastics - BNN - ReLu



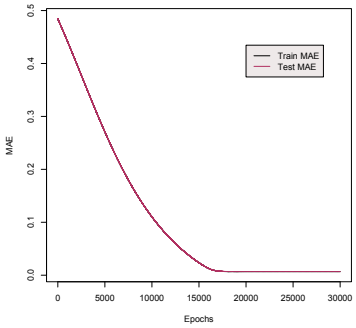
(e) All - NN - Sigmoid



(f) All - BNN - Sigmoid

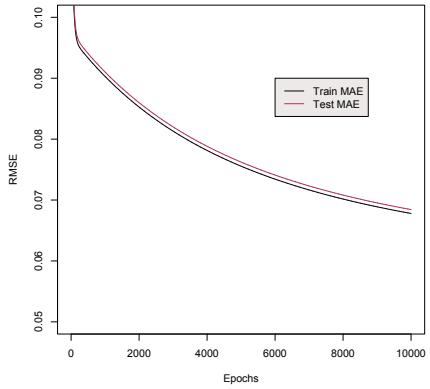


(g) Plastics - NN - Sigmoid

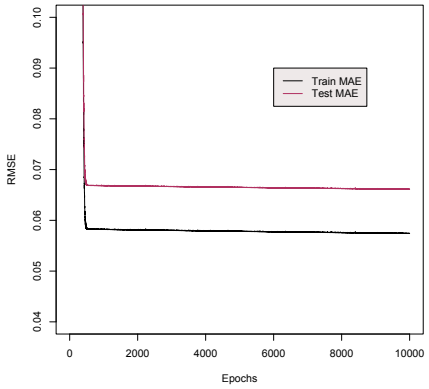


(h) Plastics - BNN - Sigmoid

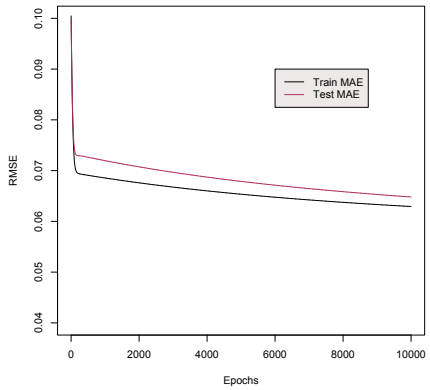
Figure 7: RMSE loss using Unilever data, training for 10.000 epochs and in case of the BNN then priors follow a standard Normal distribution.



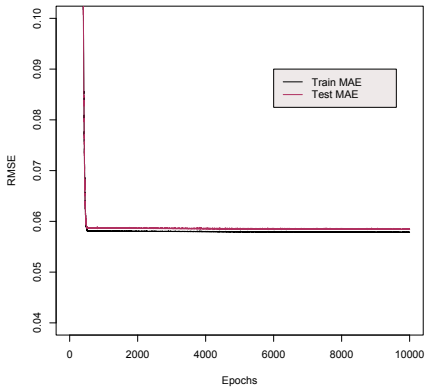
(a) All - NN - ReLu



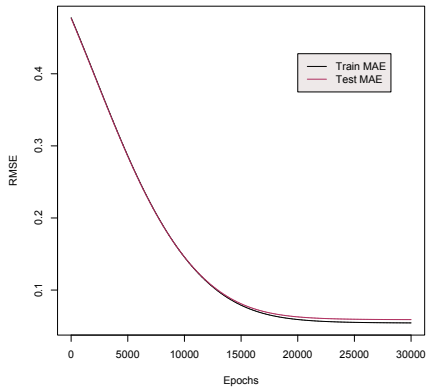
(b) All - BNN - ReLu



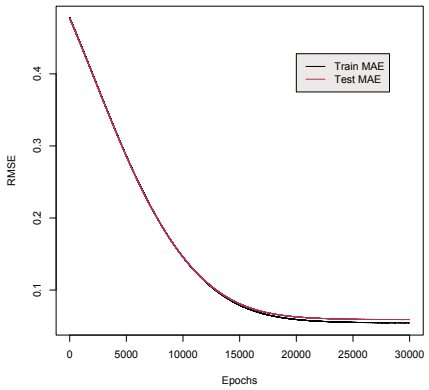
(c) Plastics - NN - ReLu



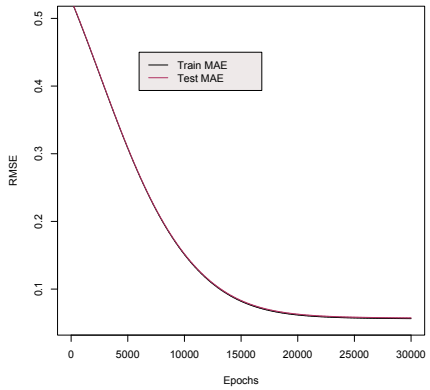
(d) Plastics - BNN - ReLu



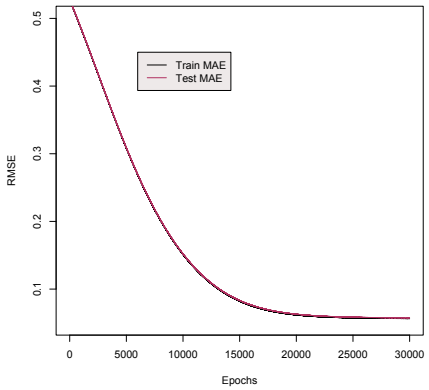
(e) All - NN - Sigmoid



(f) All - BNN - Sigmoid



(g) Plastics - NN - Sigmoid



(h) Plastics - BNN - Sigmoid

I NSGA results on Unilever data set

This Appendix gives the NSGA results on the original Unilever data set.

Table 16: Optimization results using NSGA-II with a NN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	2.40344	0.62505	-	-	1.73165	0.70898
	Costs (€/g)	-	-	0.04	0.08	0.25	0.36
	Aluminium (%)	15.9	-	5.5	-	13.6	-
	Glass (%)	7.2	-	8.4	-	8.2	-
	HDPE (%)	11.7	24.3	28.2	41.7	8.1	6.1
	LDPE (%)	15.7	5.6	0.7	9.1	6.3	16.5
	Paper (%)	7.3	-	2.2	-	6.2	-
	PET (%)	10.4	20.4	13.3	8.7	6.0	6.9
	PP (%)	4.0	16.8	6.3	18.4	10.9	35.6
	RPET (%)	11.8	12.2	31.0	22.0	10.4	17.1
	RPP (%)	7.9	20.3	2.8	0.2	13.9	17.9
	Steel (%)	8.1	-	1.6	-	16.4	-
	Time (sec.)	< 1000	< 1500	< 5	< 5	< 1000	< 1000
	MAE train	0.0383	0.0484	-	-	0.0419	0.0478
	MAE test	0.0400	0.0497	-	-	0.0435	0.0491

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

Table 17: Optimization results using NSGA-II with a BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.95975	0.79621	-	-	0.99675	0.93841
	Costs (€/g)	-	-	0.06	0.05	0.24	0.38
	Aluminium (%)	18.3	-	18.6	-	10.0	-
	Glass (%)	3.5	-	20.3	-	10.0	-
	HDPE (%)	15.6	16.8	13.4	41.1	10.8	7.8
	LDPE (%)	4.0	29.2	1.86	0.7	3	17.5
	Paper (%)	9.5	-	11.8	-	5.8	-
	PET (%)	14.4	16.9	7.3	11.2	7.0	12.8
	PP (%)	10.9	18.0	0.9	0.2	15.7	29.1
	RPET (%)	3.1	7.8	19.5	43.4	13.7	16.2
	RPP (%)	9.5	11.5	0.1	3.4	12.6	16.5
	Steel (%)	11.3	-	6.2	-	11.5	-
	Time (sec.)	< 4000	< 6000	< 5	< 5	< 3500	< 8000
	MAE train	0.0398	0.0483	-	-	0.0397	0.0473
	MAE test	0.0421	0.0483	-	-	0.0420	0.0472

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

Table 18: Optimization results using NSGA-III with a NN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	1.834	0.439	-	-	1.16966	0.56897
	Costs (€/g)	-	-	0.24	0.25	0.23	0.31
	Aluminium (%)	9.2	-	10.7	-	11.9	-
	Glass (%)	12.2	-	13.9	-	10.2	-
	HDPE (%)	7.3	11.98	5.73	17.01	4.84	17.18
	LDPE (%)	14.3	17.7	10.9	15.1	10	16.5
	Paper (%)	4.6	-	7.9	-	5.1	-
	PET (%)	9.4	23.3	9.7	21.4	6.9	19.3
	PP (%)	12.9	18.5	11.1	21.9	13.7	17.8
	RPET (%)	8.8	15.6	9.5	14.6	10.1	10.2
	RPP (%)	10.5	13.0	7.23	10.1	16.9	19.0
	Steel (%)	10.8	-	13.2	-	10.5	-
	Time (sec.)	< 1500	< 2000	< 5	< 5	< 1500	< 2000
	MAE train	0.0395	0.0496	-	-	0.0453	0.0477
	MAE test	0.0412	0.0510	-	-	0.0472	0.0489

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

Table 19: Optimization results using NSGA-III with a BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.84448	0.96563	-	-	0.72350	0.83110
	Costs (€/g)	-	-	0.15	0.40	0.21	0.36
	Aluminium (%)	9.3	-	12.1	-	10.7	-
	Glass (%)	12.0	-	11.1	-	8.4	-
	HDPE (%)	14.3	19.8	12.0	14.0	11.1	11.5
	LDPE (%)	12.6	13.7	13.2	17.8	16.3	17.7
	Paper (%)	9.0	-	8.4	-	6.3	-
	PET (%)	13.3	17.6	13.74	19.6	9.3	22.1
	PP (%)	10.1	11.5	9.1	14.4	5.5	16.0
	RPET (%)	7.1	14.7	8.8	14.0	7.9	15.6
	RPP (%)	1.8	22.7	2.7	20.3	8.8	17.2
	Steel (%)	10.6	-	9.0	-	15.7	-
	Time (sec.)	< 6000	< 15000	< 5	< 5	< 6000	< 9000
	MAE train	0.0396	0.0455	-	-	0.0410	0.0467
	MAE test	0.0420	0.0454	-	-	0.0434	0.0467

Settings: learning rate = 0.0005, epochs = 500, simulation data, ReLu activation.

J NSGA results without NN or BNN

This Appendix gives the NSGA results using simulation data, where there is no NN or BNN included inside the NSGA algorithms. In this way hypothetical material fractions predicted by the networks could be compared with this potential DGP. A limitation of this study is that the true DGP presumably is a more complex relation that requires more advanced data. In the results shown below the assumed relation between the environmental footprint (z_{LCA}) in points per gram and the material (m_k) fractions in percentages is:

$$z_{LCA}(\mathbf{m}) = \sum_{k=1}^{M} m_k z_{k,LCA}, \tag{30}$$

where m_k is the material fraction of material k measured in percentages of the total packaging weight and $z_{k,LCA}$ is the total environmental footprint of material k in points per gram. Tables 20 and 21 show the results of the NSGA-II and NSGA-III without NN or BNN, respectively.

Table 20: Optimization results using NSGA-II without a NN or BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.01806	0.01049	-	-	0.03650	0.02673
	Costs (€/g)	-	-	0.55	0.21	0.49	0.39
	Aluminium (%)	1.4	-	5.1	-	8.0	-
	Glass (%)	10.1	-	4.5	-	20.3	-
	HDPE (%)	4.7	2.8	2.2	2.7	8.5	5.2
	LDPE (%)	1.7	14.9	1.6	14.0	1.4	7.3
	Paper (%)	24.5	-	32.3	-	10.9	-
	PET (%)	4.3	9.1	11.2	13.3	4.9	17.6
	PP (%)	5.1	28.4	5.7	8.7	8.1	8.4
	RPET (%)	2.1	29.8	18.3	9.5	13.0	16.7
	RPP (%)	23.3	15.0	3.1	51.7	2.9	44.9
	Steel (%)	22.8	-	15.9	-	22.1	-
	Time (sec.)	0.026	0.028	0.913	0.388	0.643	1.498

Settings: simulation data.

Table 21: Optimization results using NSGA-III without a NN or BNN.

		Scenario Ia		Scenario Ib		Scenario II	
		All	Plastics	All	Plastics	All	Plastics
Optimized values	Environmental footprint (points/g)	0.05733	0.05125	-	-	0.05744	0.04950
	Costs (€/g)	-	-	0.91	0.68	0.94	0.78
	Aluminium (%)	8.7	-	12.8	-	11.8	-
	Glass (%)	12.1	-	14.7	-	12.8	-
	HDPE (%)	13.8	10.7	13.2	20.2	14.8	13.9
	LDPE (%)	12.0	11.1	8.4	3.7	9.4	7.3
	Paper (%)	12.5	-	8.5	-	9.9	-
	PET (%)	8.5	15.3	6.8	14.8	7.5	12.7
	PP (%)	6.1	25.9	6.6	19.7	5.6	26.6
	RPET (%)	9.0	24.4	9.4	29.6	9.1	28.8
	RPP (%)	11.0	12.6	10.5	12.0	11.3	10.7
	Steel (%)	6.3	-	9.1	-	7.7	-
	Time (sec.)	0.119	0.096	1.773	0.854	1.462	1.075

Settings: simulation data.