Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)
Edited by Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P177-cd

Jam detection in waste sorting plant based on an Autoencoder Neural Network

Calliane You, Olivier Adrot, Jean-Marie Flaus

Univ. Grenoble Alpes, CNRS, Grenoble INP, G-SCOP, 38 000 Grenoble, France*
**Institute of Engineering Univ. Grenoble Alpes*
E-mail: calliane.you@gscop.fr; olivier.adrot@gscop.fr; jeanmarie.flaus@gscop.grenobleinp.fr

By 2060, the estimated amount of worldwide plastic waste will triple and half of it will be sent to landfill. Yet, waste sorting plants that treat this waste face an unsolved recurrent main problem that is the occurrence of jams in conveyor belts. These jams limit the quantity of waste sorted, whereas waste sorting plants have a constraint to sort a certain amount of waste per week. The main causes of these jams are the complexity and variability of the composition of the waste flow (dirt, humidity, etc.). Therefore, a method based on an autoencoder artificial neural network is used to detect potential jams on conveyor belts in waste sorting plants. The method and results on real industrial data are explained and allow concluding on the feasibility of using an autoencoder model to detect potential jams.

Keywords: Fault detection and diagnosis, artificial intelligence, autoencoder neural network, reliability and safety, industrial data, waste management.

1. Introduction

From 2005 to 2019, the recyclable waste from household waste increased by 41% in France. Ademe (2023). This problem does not only affect France. Waste in the world has doubled since two decades ago. OECD (2022a). Moreover, by 2060, the estimated amount of worldwide plastic waste will triple according to the Organization for Economic Co-operation and Development (OECD). In addition, the OECD estimates that half of it will be sent to landfill and less than a fifth will be recycled. OECD (2022b). Also, by 2023, in France, regional policies have the will to attain zero waste in landfill. La Région Auvergne-Rhône-Alpes (2022). Therefore, it is necessary to limit the amount of unsorted waste going to the landfill or incineration, and to optimize the recycling of waste. Many constraints lead to unsorted waste. In particular, the time to sort a theoretical maximum amount of waste depends on the initial sorting capacity of the waste sorting plant, that was decided when it was built. The imposed land and the budget allocated by the city to build the whole sorting plant mainly limit it. Also, over time, with the urbanization, homes get closer to existing waste sorting plants, and this adds health and safety constraints to respect. So, waste sorting plants (figure 1) have to limit the stock of waste waiting to be processed to limit the expansion of smells, rats, disease and rotting. It is also difficult to build, renovate, or expand existing or new sorting plants due to time, cost and territorial restrictions.

Fig. 1.Waste sorting plant of household waste. Aktid (2023).

Therefore, it is hard to answer to the increase in the amount of waste to sort. Plus, the waste not sorted in time can be sent to incineration or landfill. This adds ecological pollution, human and financial costs.

The household waste segment is the focus of this research article. With its specificities, it is the most complex segment in waste sorting. The composition of the waste flow incoming is dependent on the weather, on the consumption habits of inhabitants and their education to sort correctly. These contribute to the complexity and variability of the composition of the waste flow

(dirt, entanglement of waste, humidity, shapes, material, etc.) inducing important jams.

The main objective of waste sorting plants is to optimize the quantity of waste sorted and to not miss any objects to sort; thus, there are recirculation loops to sort it again (figure 2).

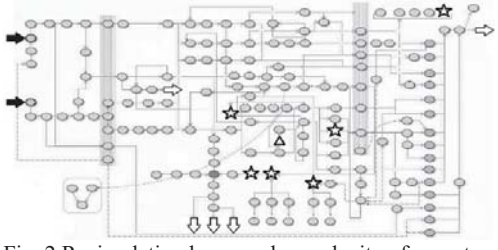

Fig. 2.Recirculation loops and complexity of a waste sorting plant. Aktid (2023).

In a waste sorting plant, the flow of waste to sort enters through one or two machines represented by black arrows, is treated by different types of machines (as conveyor belts) represented by circles and ends through multiple output lines represented by the white arrows. The machine represented by a white triangle is used to train and test the model, and all the machines represented by a star are then used to test this model. The details will be given in the next sections. In sum, waste sorting plants have all kinds of different conveyor belts and machines assembled with many recirculation loops that increase and complicate the risk of jams.

Fig. 3.Jam preventing the flow of waste from advancing. Aktid (2023).

When waste piles up on a conveyor belt, it gets stuck and leads to a jam, as shown in figure 3 in the black frame. That jam causes the motor of the machine to force until it breaks, so the machine stops. This accumulation causes overflowing waste that falls all over the other machines below or next to the jammed conveyor belt and the machines sending their treated flow of waste to the current jammed machine also overflow, and so on. It leads to a chain reaction that causes jams all over the waste sorting plant. Then the operators need to go inside each machine because waste is so stuck and entangled, that it is only possible to remove it manually. It means arduous and dangerous additional work for the operators going into the cogs, corners difficult to access. To ensure safety, the whole sorting line is stopped, and clearing a jam can take several hours. It results in less productivity and volume of treated waste. Plus, the waste taken out when clearing the jam is not going back into the flow of waste to treat to avoid causing another jam. This increases the pile of waste to incinerate or to send to landfill. Hence, these waste sorting plants face an unsolved recurrent and main problem that is the occurrence of jams on conveyor belts. Therefore, it is needed to detect these potential jams.

This research deals with the feasibility of early detection of potential jams on conveyor belts, with the goal of warning the operators soon enough, so they can try to push the excess of waste without stopping the machine. Thus, the operators will be able to clear the incipient jam before it becomes too hard to treat it. On top of that, two operators at best monitor a hundred of machines of the waste sorting plant. With a model capable of detecting jam as soon as possible, they do not need to rush to the problematic machine. This decreases the risk of accidents, eases the work, and consequently optimizes the productivity of the waste sorting plant.

In the second section, the problem and constraints of detecting jams on real industrial data from a waste sorting plant in production are explained. The third section details the autoencoder artificial neural network, i.e. the method implemented to detect jams on a control conveyor belt and the process to build autoencoder models. Yokkampon et al. (2020). The fourth section presents and discusses the first results obtained from actual historical raw production data. Then the conclusion open to potential perspectives.

2. Problem statement and constraints

In our previous work a first baseline algorithm, a supervised k-nearest neighbors algorithm has been used to develop a classifier model to distinguish normal operation from the occurrence of a jam. You et al. (2022). This model is currently used in production in a household waste sorting plant. However, because it is a supervised algorithm, it

requires constructing as many training data sets (in this specific case composed of five classes) as there are machines with different behaviors (screens, conveyor belt, ballistic separators, optical sorter, etc.). In addition, each machine has its own specificities one-way or two-way operation, load capacities, with or without variable speed, different amperage, different types of waste to treat, etc. Also, it requires building a dataset with balanced classes. Nevertheless, several classes of data are difficult to collect, such as start-ups because they only last up to 5 seconds, and jams that are rare on some machines. Additionally, the operators do not systematically record every jam, and many different cases need to be scrutinized to build the training data set.

Another limitation to developing any model with any algorithm is that the only data at our disposal is the electrical intensity of the machines in amperes, whether for current inverter machines with constant or variable speed, forward and reverse.

Therefore, it is necessary to test and find an algorithm to build a model that does not require a lot of tedious manual work up front, like waiting for new data, collecting, analyzing and selecting the most representative data for each class.

3. Autoencoder models for jam detection on different conveyor belts

3.1. *Autoencoder neural network*

First, it is useful to start explaining shortly from artificial intelligence and then to narrow down to autoencoder for better understanding. Artificial intelligence is the engineering science of building intelligent computer programs capable of achieving goals. McCarthy (2007). It contains machine learning and deep learning is a subset of machine learning. Jakhar and Kaur (2020).

Machine learning includes algorithms trained with data to achieve a task without explicitly being programmed for this specific task. Mitchell (2006). The algorithm processes the data and learns parameters in order to minimize the error between the real data and the predicted data, this is called the training or learning, and then can produce a decision (prediction, classification, or detection).

Then deep learning, a subset of machine learning, includes computational algorithms imitating the architecture of biological neural networks in the human brain. It also learns to process data by trying to understand the data and the relationship between the features and also making the classification or prediction. Bengio and LeCun (2007).

A neural network is an architecture composed of layers of interconnected neurons. These connections enable to pass and process the data from the input to the output aka forward propagation of the neural network through the layers of neurons, and to pass information back from the output results to the neural network to improve the results with backpropagation. Van Veen and Leijnen (2019) have listed approximately thirty types of neural network by architecture, and this is without considering their variants nor the new ones yet to be created like the last version of the chatbot ChatGPT. Zhou et al. (2023).

An autoencoder is a neural network, it is represented in a simplified version in figure 4.

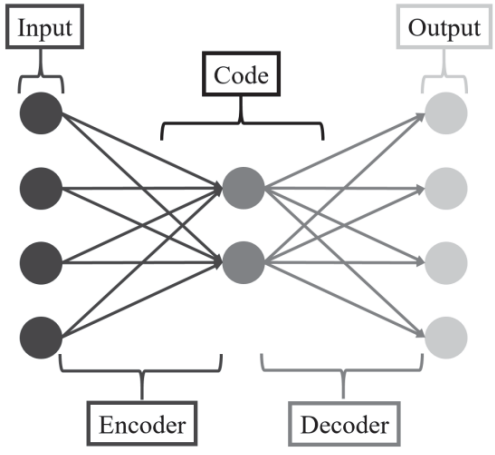

Fig. 4.Autoencoder neural network

The neurons are represented by circles, and the lines represent the connections between the different layers of neurons, and the different shades of gray express that the data are processed and transformed. The input layer is composed of the input data. The output layer is composed of the final output data. And the hidden layer is any layer between the input and output layers. It is composed of the transformed input data, and a neural network can have as many hidden layers as needed. One connection connects two neurons together. A connection is a calculation formula composed of features, i.e. the input data,

parameters like weights and bias, a summation function, and an activation function to propagate the data from one neuron to the next one. To train the neural network, hyperparameters like epochs, a batch size and an optimizing function are needed. Parameters are values estimated by the neural network from the input data to minimize the error. So, these parameters are updated until the error is minimized, whereas hyperparameters are manually set and are not updated by the neural network. Hyperparameters help the neural network to estimate the parameters and control the process of training. Weight is a value multiplied with the features to express the importance of the feature among all the features. It can be a negative or positive value. Bias is a necessary value added to correct and move the activation function to the left or right of the plan; it can be a negative or positive value, too. The summation function sums the product of the features and weights with the bias. The activation function is a curve (e.g. sigmoid) to be fit and reshaped by the product of the feature and the weight plus the bias corresponding to the connection, and the addition of all the activation functions reshaped results in an approximation of the input data. An iterative optimizer aka the optimizing function is required to modify the weights to reduce the error after each prediction or classification, or reconstruction in the case of an autoencoder. It iterates as many times as there are batches, and the parameters are updated after the entire batch is passed to the neural network. A batch size is a hyperparameter used to divide the training input data into several mini-batches, e.g. when the input data to learn is too large for the computer or to improve the error better than feeding all the data once to the neural network and updating the parameters only once. To use all the training data, it is preferable to use a multiple of the total number of the training data because some deep learning libraries do not take the remaining or last batch if it is not of the same size of the specified batch size. Then when all the batches have been passed through the neural network forward (forward propagation) and backward (backpropagation) one time, it is an epoch. It is common that one epoch is not enough for a neural network to achieve the minimum error. At every launch of a new epoch, the parameters found in the previous epoch are kept to try to minimize the error; it is the input to the new epoch. In addition, at every epoch, the training dataset is shuffled before splitting into mini batches or the mini-batches are feed in different order to the optimizer, and each of the starting parameters are different. However, it is important to decide to stop the training before the model is overfitting; it means it fit almost perfectly the training data, so any data that does not resemble the training data will have a bad prediction.

Finally, the autoencoder is a neural network that is comparable to a data compressor. It tries to map the input to the output by retaining relevant information. Mienye et al. (2020). It is useful for helping a neural network to learn by simplifying the training data when you have a large number of features, noisy features, and useless features. The input data is propagated from its input layer to the next layer, a hidden layer, all this process is called the encoder. It means the data has been encoded to results in the hidden layer. And then the hidden layer data is propagated from the hidden layer to the output layer, this process is called the decoder. It decodes the encoded data from the hidden layer to result in the final output. So, the output is a compressed and reconstituted version of the input.

But an autoencoder can also be used to detect anomaly in the input data. The anomaly detection part is when the difference between the reconstituted input, i.e. the output, and the training data input is superior to a reconstruction threshold of the trained data, then it is considered anomaly data, else it means that the reconstruction is similar to the uncompressed training data, then it is considered normal data. Further details are given in section 3.2.

3.2. *Definitions and data annotations*

From waste sorting plants only one type of data is available for this research work, it is the electric current intensity of the motor of the conveyor belts. These data can be classified according to two states of the machine: the normal operating state and the abnormal one. Data are considered normal when there is no kind of anomaly causing the machine to stop. Normal data are especially collected only on the working shift, where no anomaly or jam is recorded by the operators on site. It includes the periods when the machine is stopped, i.e. the electric current intensity is close to zero, the start-ups of the machine (except the start-ups due to a stop occurring after a jam or anomaly), and the normal operating state with an

electric current intensity representative of the nominal state of the machine. Abnormal data correspond to the time from when a jam starts to when the machine stops. To train the autoencoder, only the normal data of a chosen reference machine is used. Therefore, the model only represents the normal data of this machine; it means that any data with a different behavior will be detected as abnormal data. The training dataset is balanced with as many periods of start-ups, stops and nominal states to limit bias. For the tests, full shifts of about seven hours of production where there are only recorded and confirmed jams by the operators on the site are used. Figure 5 shows an example of both the normal and the abnormal states of a machine.

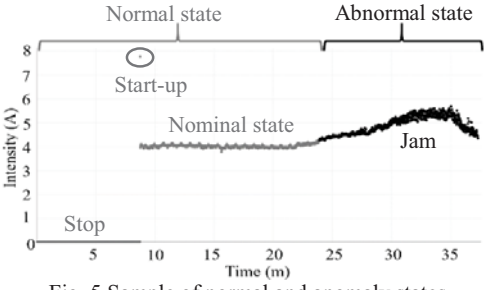

Fig. 5.Sample of normal and anomaly states

To build a training dataset, raw data of electric current intensity per second are retrieved in the time series format. A chronologically ordered sequence of values that represents the evolution of the same observation is a time series. A univariate time series means that only a single value for each instant is recorded. Rajpurkar et al. (2017). The annotations are kept as close as possible as in our previous work for coherence. A time series is defined by $\boldsymbol{V_i} = \boldsymbol{v_{(hi)}}$. You et al. (2022), where i is the time index, the window size $\boldsymbol{h}$ of the time series $\boldsymbol{V_i}$ is $\boldsymbol{h} = 10$ seconds and $\boldsymbol{v_i}$ defines the electrical intensity value normalized by the nominal state of the machine at time $\boldsymbol{i}$.
The formula used for normalizing is:

$$\boldsymbol{v_i} = \frac{(\boldsymbol{v_{i,m}} - \boldsymbol{v_{min}})}{(\boldsymbol{v_{nominal}} - \boldsymbol{v_{min}})} \quad (1)$$

With the normalized data $\boldsymbol{v_i}$, the measured data $\boldsymbol{v_{i,m}}$, the stop state $\boldsymbol{v_{min}}$ and the average of the raw data during nominal periods $\boldsymbol{v_{nominal}}$. This normalization enables to test the use of the same training data on multiple machines with similar behaviors. Then the vector $[\boldsymbol{v_{i-h+1}}, \dots, \boldsymbol{v_{i-1}}, \boldsymbol{v_i}]$ represents the time series $\boldsymbol{V_i} = \boldsymbol{v_{(hi)}}$. After iterating experiments, $\boldsymbol{h}$ has been determined as a compromise between a small size to limit the delay of the jam detection and a sufficient time horizon to filter data noises from sensors. Each time series $\boldsymbol{V_i}$ is reduced by a features vector $\boldsymbol{F_i} = [\boldsymbol{f_i^1}, \boldsymbol{f_i^2}, \dots, \boldsymbol{f_i^7}]$ composed of seven calculated features, $\boldsymbol{f_i^n} \in [$*'mean', 'median', 'minimum', 'maximum', 'slope', 'intercept', 'standard deviation'*$]$. $*^{\boldsymbol{train}}$ and $*^{\boldsymbol{test}}$ are the exponents of the time series $\boldsymbol{V_i^*}$ and the associated class $\boldsymbol{C_i^*}$ to help to know which dataset is used. The autoencoder is trained only with the $\boldsymbol{F_i^{train}}$ features of each series $\boldsymbol{V_i^{train}}$. $\boldsymbol{F_i^{train}}$ contains only normal data, and $\boldsymbol{F_i^{test}}$ contains normal and jam data. The $\boldsymbol{C_i^{train}}$ class is used as a reminder of which class the sample of data belongs to, it helps to build datasets. For each $\boldsymbol{V_i}$, so each $\boldsymbol{F_i}$, a class vector $\boldsymbol{c_{(hi)}} = [\boldsymbol{c_{i-h+1}}, \dots, \boldsymbol{c_{i-1}}, \boldsymbol{c_i}]$ is assigned, with $\boldsymbol{c_i} \in$['normal', 'jam'] for normal and abnormal states associated to the intensity values $\boldsymbol{v_i}$. Also, the last class $\boldsymbol{c_i}$ of $\boldsymbol{c_{(hi)}}$ determines $\boldsymbol{C_i}$, it means that in $\boldsymbol{c_{(hi)}}$ all the $\boldsymbol{h}$ classes are identical.

For the autoencoder to decide if the data is normal or abnormal, one threshold $\boldsymbol{t^{train}}$ to test the autoencoder output is calculated from the training data:

$$\boldsymbol{t^{train}} = \overline{\left(\boldsymbol{F_{i,enc}^{train}} - \boldsymbol{F_{i,dec}^{train}}\right)} + \boldsymbol{\sigma}\left(\boldsymbol{F_{i,enc}^{train}} - \boldsymbol{F_{i,dec}^{train}}\right) \quad (2)$$

Where the operators $\overline{(\ )}$ represent the mean and $\boldsymbol{\sigma}(\ )$ the standard deviation of the terms between brackets. $\boldsymbol{F_{i,enc}^{train}}$ are the encoded input data that become the code and $\boldsymbol{F_{i,dec}^{train}}$ are the code data decoded, i.e. the output of the model (Figure 4). This threshold is an indicator of how well the reconstructed data is close to the trained data. So, the average of test data $\boldsymbol{m^{test}}$ is calculated by:

$$\boldsymbol{m^{test}} = \overline{\left|\left(\boldsymbol{F_{i,enc}^{test}} - \boldsymbol{F_{i,dec}^{test}}\right)\right|} \quad (3)$$

And is compared to the threshold $\boldsymbol{t^{train}}$. In case of $\boldsymbol{m^{test}} < \boldsymbol{t^{train}}$ consequently $\boldsymbol{F_i}$ and $\boldsymbol{C_i}$ are assigned ['normal'] otherwise are assigned ['jam']. Then the first occurrence of $\boldsymbol{C_i}$ of $\boldsymbol{F_i} \in$['jam'] is compared to the timestamp of the jam recorded by the operators on site.

3.3. *Modeling process*

First, records of one intensity data value per second per machine are collected from the waste sorting plant database. These data are normalized and summarized into seven features that are detailed in section 3.2. Then the data are split into the train and test datasets. The choice of the machines to study is made by the experts and the operators to answer to the needs on the production site. These machines cumulate the most jams in total in frequency and duration for a year period. So, they cost the majority of patrol and manual clearing time. Additionally, the operators monitor closely these machines, and this is important for their adhesion to using the model and to obtain their feedback for future real-time testing phases on site. These machines are one-way conveyors belt with a direct motor, i.e. with constant speed, therefore with an electric intensity curve without strong oscillations and where jams are more easily identifiable post-events. However, they all have different amperage range and different waste flow to handle. And to kill two birds with one stone because these conveyor belts are highly demanded by the operators, and they have different motors, data from conveyor belts with a variable speed, different intensity curve and strong oscillations than what the model was trained on are collected. Figure 6 schematizes the data collection, preparation and model construction. Lee et al. (2021).

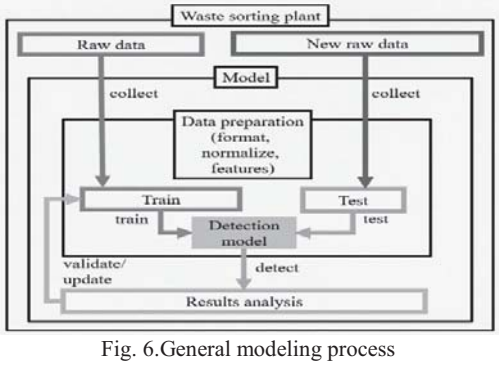

Fig. 6.General modeling process

These data are collected to test two hypotheses.

- Hypothesis 1: can one model trained on one single machine be applied on other machines that have a similar shape of electric intensity curve?
- Hypothesis 2: can a model obtained on a constant speed machine be used to monitor a variable speed machine?

With these hypotheses, it is possible to determine if the model succeeds in covering a maximum of different conveyor belts.

After the data preparation, the model is trained, i.e. it learns from the training data produced from a direct motor machine chosen as a reference (white triangle on figure 2). Next, it is tested on data that the model has never seen from a panel of the machines composed of direct or variable motors. And if the model is not satisfactory, then the model is updated with the steps since the data collection to the production of the model. These iterations help to improve the model by enriching the training data. Finally, the model is valid when the detection results are satisfactory on the test set on.

3.5. *Training and validating of the autoencoder model*

For training and testing, the model performances are evaluated by comparing the first occurrence of the detection of an anomaly by the model to the jam timestamp recorded by the operators on site. Then the experts validate the results. Operators' top priority is to limit the number of missed jams, i.e. the number of false negatives. Sokolova and Lapalme (2009). When the number of correctly detected jams is at its peak without containing false positive, i.e. normal data detected as jam, the model is validated and saved.

4. Autoencoder model results

Presently, obtaining other data from the waste sorting plant is not possible for diverse reasons like uncollected or missing data, confidentiality, lack of resources, etc. It explains the reason why only seven machines could be used and tested for now.

The model trains on normal data transformed into 1006 unannotated features vectors $\boldsymbol{F_i^{train}}$ from one conveyor belt only (white triangle on figure 2). And it is tested on normal and abnormal data transformed into 189000 unannotated feature vectors $\boldsymbol{F_i^{test}}$ from all the conveyor belts (white triangle and stars on figure 2).

To test hypothesis 1, only five one-way conveyor belts with direct motor with similar shape of electric intensity curve are available (machine

used for training included). The model is tested on each of the five machines. The performances obtained for the machine used for training (white triangle on figure 2) and for the four other machines are similar, since the model detects the jams recorded by the operators. As an example, figure 7 zooms on a detected jam on one direct motor conveyor belt that is not the machine used for training the model. Normal and abnormal (jam) states predicted by the autoencoder model appear respectively in gray and black.

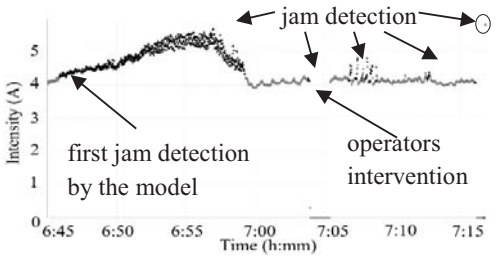

Fig. 7.Detection on one machine with direct motor

It is a conveyor belt with a direct motor, i.e. moving in one direction and having a constant speed. Its nominal current intensity is around 3,5 A whereas the machine used for the training is at 2,1 A. The autoencoder model detects correctly the jam before the intervention of the operators. At 07:03:27 the operators have confirmed that they saw a jam starting. And the model detects the first anomaly at 06:45:52, i.e. 00:17:35 before the operators. It means that the model can help warn the operators upstream, especially when they are not near the machine or supervising it. As a reminder, there are no operators posted full time on a supervising screen to watch over all the machines in the waste sorting plant. Instead, there is one or at most two patrolling operators for the whole sorting plant, and they patrol all day long up to 12 km of walking and using their pole to push the waste while the machines run, to limit any creation of jams. Some points after the machine restarts at 07:05:28 after the jam, are still detected as jam, and it is normal because often the operators clear jam as much as possible manually, but some waste remains stuck and is easier to clear by restarting the machine, but that waste can still generate a new jam.

In addition, some points on the normal operating state in gray are detected as jam, this is explainable by the fact that time series are used to train the model and there are time series that are composed of overlapping normal and abnormal states (jam). Moreover, the training data set is still at their early stage and does not contain all the possibles high peaks when the machine starts. Thus, it would be necessary to enrich the training set to push further the experiments on the future works. Overall, on the five machines tested, the model detected from 14 seconds to 17 minutes and 35 seconds before the operators, with a mean of 5 minutes and 15 seconds, and a median of 2 minutes and 43 seconds.

For hypothesis 2, as expected on two conveyor belts that do not have similar shape of electric intensity curve, the autoencoder model predicts the majority of points as abnormal states. This implicates that a new training set must be made that fit these machines because if the training set contains too different data, it may lead to the model performing averagely on both types of machine.

5. Conclusion

This work investigates if an autoencoder model trained on data representative of a normal operating state from only one machine could be used to detect potential jams and to warn the operators to help ease their work. More precisely, the aim is to see if it is possible to apply the same model trained on one machine on other machines, without retraining the model. Additionally, these tests investigate the possibility to save time on building the model by only using data considered as a normal operating state of the machine, which is more easily available and recognizable than the abnormal state. This means that there is no need to have to wait a year or more to obtain enough jam data from the production line. In addition, it may not be needed to build as many models as machines needing the algorithm applied on. Also, to train an autoencoder it takes about two and a half minutes. Financially, the autoencoder can help gain around eleven hours of production time like the k-NN from our previous work, which is currently deployed on a real production site.

As a matter of fact, the results indicate that on other machines that have similar shape of electric intensity curve, the first point detected as a jam happened before the machine is stopped due to a jam that is recorded in the database by the operators, the autoencoder model detects potential

jams before the operators intervene to stop the machine. And, as expected, the model did not perform well on machines with totally a different behavior.

By extrapolation, it may be possible to group all the machines from one waste sorting plant into different groups of comparable machines together, produce one model per group, and reduce the number of models to build. This is a major time saver in the entire process of developing a detection model. Nevertheless, it will be necessary to ask the operators to check the results on site.

The perspectives are to establish criteria to group machines with comparable normal state behaviors, build a model for each group, and evaluate their performance. However, monitoring the behavior evolution of the machines is necessary. Over time, machines age and multiple maintenance are done. Moreover, by using the model detection, the operators may treat the jams sooner, so it will affect the machines' behaviors, and it may be necessary to update the models.

Also, to avoid the time-consuming training dataset building step each time there will be a need to update or create a new model, it may be necessary to check if it is possible to build a model capable of following the machines' behaviors evolution like with online learning or reinforcement learning.

Acknowledgement

Aktid company authorizes the use of the raw data and the pictures of waste sorting plants, for the experiments.

References

Ademe. (2023). *Déchets Chiffres-clés - Edition 2023*. La librairie ADEME. https://librairie.ademe.fr/dechets-economie-circulaire/6108-dechets-chiffres-cles-edition-2023-9791029720536.html

Aktid. (2023).

Bengio, Y., and Y. LeCun (2007). Scaling learning algorithms towards AI. *Large-scale kernel machines, 34(5)*, 1-41.

Jakhar, D. and I. Kaur (2020). Artificial intelligence, machine learning and deep learning: definitions and differences. *Clinical and experimental dermatology, 45*(1), 131-132.

La Région Auvergne-Rhône-Alpes. (2022). Zéro enfouissement des déchets d'ici 2030. https://www.auvergnerhonealpes.fr/actualites/zero-enfouissement-des-dechets-dici-2030

Lee, T. H., G. Skofteland, and MA. Lundteigen (2021). Performance Management of Safety Instrumented Systems for Unmanned Facilities Using Machine Learning: Decision Support System for SIS. *Proceedings of the 31st European Safety and Reliability Conference (ESREL 2021), University of Angers, France*, 2878-85. Research Publishing Services.

McCarthy, J. (2007). From here to human-level AI. *Artificial Intelligence, 171*(18), 1174-1182.

Mienye, I. D., Y. Sun, and Z. Wang (2020). Improved sparse autoencoder based artificial neural network approach for prediction of heart disease. *Informatics in Medicine Unlocked, 18*, 100307.

Mitchell, T. M. (2006). *The discipline of machine learning (Vol. 9)*. Pittsburgh: Carnegie Mellon University, School of Computer Science, Machine Learning Department.

OECD. (2022a). *Plastic Pollution is Growing Relentlessly as Waste Management and Recycling Fall Short, Says OECD*. OECD Publishing.

OECD. (2022b). *Global Plastics Outlook: Policy Scenarios to 2060.* OECD Publishing.

Rajpurkar, P., A. Y. Hannun, M. Haghpanahi, C. Bourn, and A. Y. Ng (2017). Cardiologist-level arrhythmia detection with convolutional neural networks. *arXiv preprint arXiv:1707.01836*

Sokolova, M., and G. Lapalme (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management, 45(4)*, 427-437.

Van Veen, F., and S. Leijnen (2019). A mostly complete chart of Neural Networks. *The Neural Network Zoo. Retrieved December, 28*, 2020.

Yokkampon, U., S. Chumkamon, A. Mowshowitz, and E. Hayashi (2020). Autoencoder with Spiking in Frequency Domain for Anomaly Detection of Uncertainty Event. *J. Robotics Netw. Artif. Life, 6(4)*, 231-234.

You, C., O. Adrot, and JM. Flaus (2022). Jam detection in waste sorting conveyor belt based on k-Nearest Neighbors. In Leva, M.C., Patelli, E., Podofillini, L. and Wilson, S. (©2022 ESREL2022 Organizers)*, Proceedings of the 32nd European Safety and Reliability Conference (ESREL 2022),* 8. Research Publishing, Singapore.

Zhou, C., Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, ..., and L. Sun (2023). A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *arXiv preprint arXiv:2302.09419*.