Please cite the Published Version

Additional Information: This is an Open Access article which appeared in Expert Systems with Applications, published by Elsevier
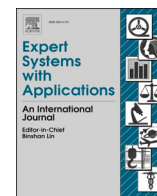
A new weakly supervised learning approach for real-time iron ore feed load estimation

Li Guo [a,1,*], Yonghong Peng [a], Rui Qin [a], Bingyu Liu [b]

[a] *Department of Computing and Mathematics, Manchester Metropolitan University, United Kingdom*
[b] *Ansteel Mining Corporation, Anshan, Liaoning, PR China*

ABSTRACT

Iron ore feed-load control is one of the most critical settings in a mineral grinding process. It has direct impact on the quality of final iron products. The setting of the feed load heavily replies the characteristics of the ore pellets. However, such characteristics are challenging to acquire in many production environments, requiring speical equipments and complicated modelling process with a high cost. To provide an low-cost and easier-to-implement solution, in this paper, we present our work on using deep learning models for direct ore feed load estimation from ore pellet images. To address the challenges caused by the large size of ore images and the shortage of accurately annotated data, we proposed to use a weakly supervised learning apporach with a two-stage model training algorithm and two neural network architectures developed. The experiment results show competitive model performance, and the trained models can be used for real-time feed load estimation for grind process optimisation.

1. Introduction

The ball mill grinding circuit (GC) plays an essential role in production in the mineral industry. The low-grade iron ores must go through a beneficiation process for further concentration (Fig. 1).

The quality of the product depends on the control of a range of set-points within a GC process, such as ore load (feed load), mill pressure, water level, and pump pressure, as shown in Table 1.

In mineral industries in China, the GC control practices are manually designed based on the laboratory analysis result of the product as feedbacks for adjusting those setpoints. Amongst all possible setpoints, the feed load setpoint needs to be modified as the others' adjustments require a highly knowledgeable and experienced human operator. Also, frequent change of those setpoints considerably reduces the lifetime and stability of the equipment. In other words, the product quality largely depends on the ore characteristics and the feed load settings. For example, for ore pellets that are hard to grind, the feed load is set low to ensure the product quality, and that is set higher for easier-grinding ore for better product throughput.

Even with this simplified mechanism (single setpoint feedback control), the current practice often leads to volatile product quality or inefficient productivity due to the considerable latency between the actual control operation and the lab result generation. The laboratory analysis is usually conducted every four hours because of the resources required and the chemical process involved. This interval is much longer than the time needed for a single GC-cycle (20–30 min, from ore feed to final product). Before the latest lab test results arrive, human operators must use their experience to set feed load for a better-quality product. However, as a more typical case, inexperienced human operators often use the latest lab test result as the only indicator for feed load adjustment for the upcoming ore pellets. It is not surprising to see that such practices lead to poor production performance. Therefore, there is a need for methods that either provide instant feedback on product qualities or provide real-time guidance on tuning the feed load to ensure acceptable product quality.

The mainstream methods/equipment deployed are based on ray sensors such as XRT and XRF (Donskoi, Suthers, Fradd, Young, Campbell, Raynlyn, & Clout, 2007; Donskoi, Raynlyn, & Poliakov, 2018; Li, Klein, Sun, & Kou, 2020; Robben, Condori, Pinto, Machaca, & Takala, 2020) to the ore's characteristics to provide real-time information. Although ray-sensors come with high classification accuracies, their cost and radiation are very high. In recent years, as alternatives, vision/
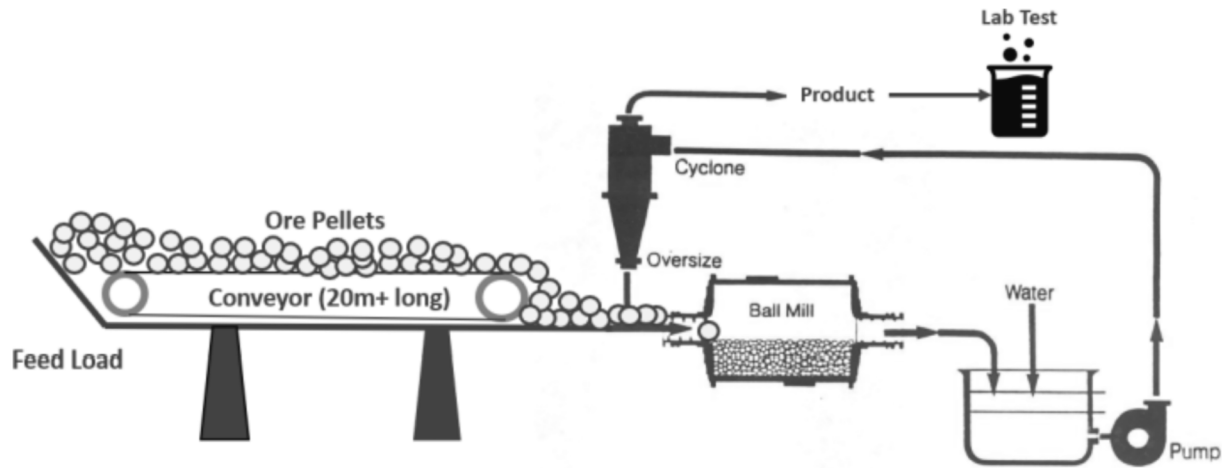
Fig. 1. The single state ball mill grinding circuit process.

Table 1
Setpoints for a Grinding Circuit.

Symbol	Setpoints
s_1	Feed load (range from 120 tons to 146 tons). This setpoint controls how much ore pellets are fed onto the conveyor
s_2	Ball mill water level. Water is regularly injected into the ball mill to ensure the grinding operation does not damage the ball mill.
s_3	Hydraulic inlet pressure. When the hydraulic pressure of the hydro-cyclone is lower than the stable value, the fine slurry cannot be separated, and it causes a pulp overflow.
s_4	Dilution water level setpoint for the pump pool.
s_5	Slurry density setpoint

image-based approaches have attracted significant interest from both academia and industry because of their low cost, easy/flexible installation and maintenance process, and competitive performance compared to the other solutions. Many attempts have been made to apply machine learning or deep learning-based methods for ore image analysis. These early works primarily focus on the tasks of ore sorting (Singh & Mohan Rao, 2005; Liu, Zhang, Liu, Wang, et al. 2021b; Gülcan & Gülsoy, 2017), ore grade estimation (Perez, Estévez, Vera, Castillo, Aravena, Schulz, & Leonel, 2011a; Maitre, Bouchard, & Paul Bédard, 2019; Donskoi et al. 2007; Liu, Zhang, Liu, Lei, et al. 2021) and particle/blast fragment size estimation (Dong & Jiang, 2014; Hamzeloo, Massinaei, & Mehrshad, 2014; Donskoi et al. 2007; Donskoi et al., 2018; Liu, Zhang, Liu, Wang, et al. 2021a). Despite the promising results, there exist significant differences between the GC process and the other ore production processes. For instance, for ore sorting based analysis, the
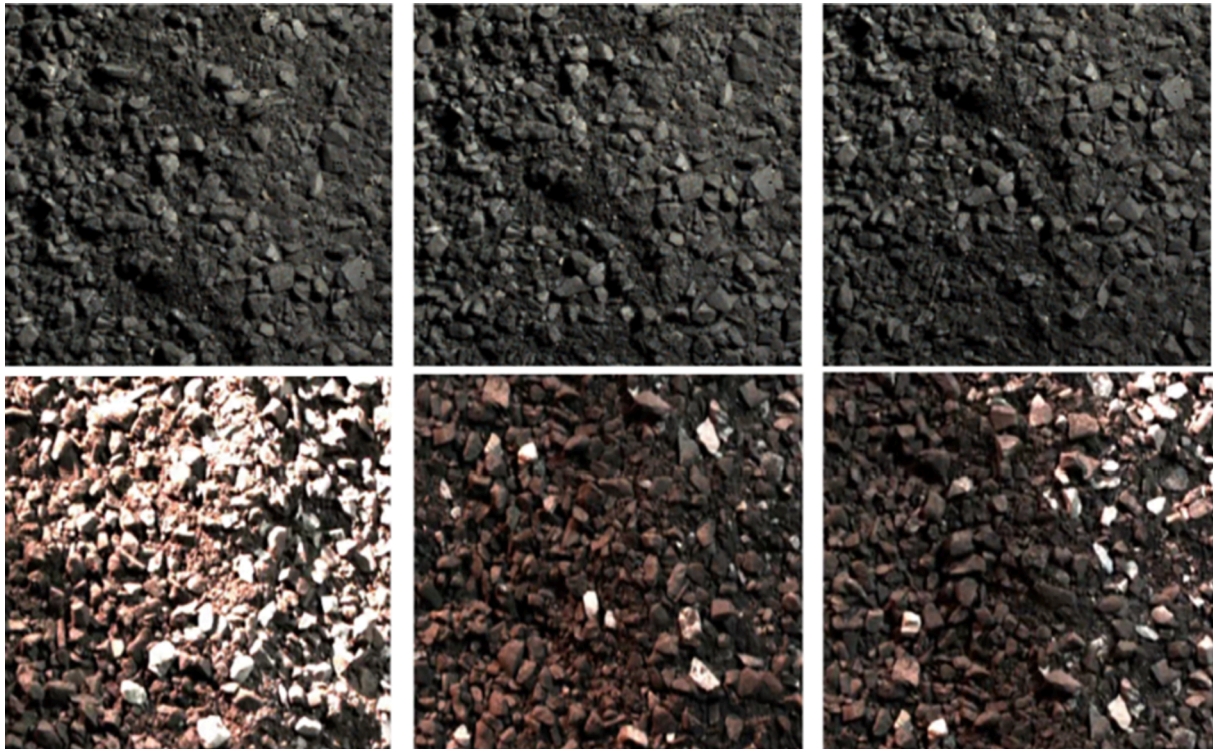

Fig. 2. Example images of ore pellets on the conveyor. The top row shows the ore pellets under normal lighting conditions, while the bottom row shows the ore pellets under different lighting conditions.
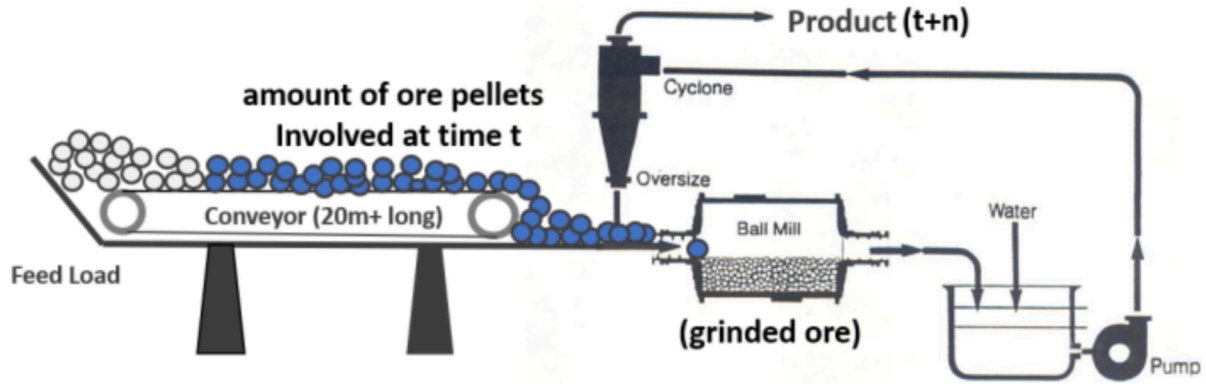
Fig. 3. The amounts of ore pellets required for a single product result.

size of ore rock is usually bigger. Also, in the early reported work, ore rocks are often arranged on a conveyor or testbench sparsely with no overlap for image capturing. Images are then fed into a machine learning model for extracting remarkable features of each rock. In addition, ore images are all manually annotated to ensure the best-quality training samples are used. The above conditions, especially the last one, barely hold in practice in a grinding process. There are a few GC specific challenges that cannot be easily addressed with the existing methods.

First, in a grinding process, the ore is crushed and stir-mixed as an initial step. The ore pellets produced should have no significant difference in size and are much smaller than that from the ore sorting stage. They are then transited to the ball mill through the conveyor in heaps with random occlusion, as shown in Fig. 2 (top row). This makes it hard for a simple model to extract individual pellet features or directly predict ore characteristics. In other work, microscopy images are often used for ore particles with small sizes instead of from commodity cameras. Second, apart from the lab test result, there is no ground truth of the ore characteristics (ore hardness, ore types) available in the GC production environment. The ore type data from the upstream stock are often inaccurate or wrong due to many uncontrolled operations. Third, as only a single lab test result is generated every 4 h, it is costly and time-consuming to collect enough accurately annotated training samples, covering various types of ore pellets under different conditions, such as lighting (Fig. 2, bottom row) feed load. Furthermore, knowing ore characteristics does not optimise the grinding process directly, as such information must be transferred into executable operations through additional modelling.

To address the above challenges, in this paper, we present an image-based approach for real-time ore feed load estimation using deep learning models and weakly supervised learning methods. Our work's fundamental hypothesis is that it is possible to directly infer the optimal feed load settings from ore pellet images without explicitly modelling the ore characteristics (latent variables).

The main contributions of this work are as follows.

1> To the best of our knowledge, this work is the first attempt that applies image-based methods for direct ore load estimation in the ball mill grinding process. Due to its low cost and easy deployment/maintenance, it has clear potential for improving the productivity of a grinding process, thus having direct economic benefits for mineral industries.

2> We propose to use a new two-stage modelling algorithm, with which two neural network architectures are designed for both patch-image level feature extraction and global classification.

3> We propose treating the problem as a weakly supervised learning problem and demonstrating how smaller patch images with wrong labels can be utilised directly for model training with theoretical and empirical studies.

4> We conduct a series of experiments and provide empirical evidence, showing that the proposed work not only perform well with the testing data but also yields a significant economic boost in actual production.

The remainder of this paper is organised as follows. Section 2 explains our methodology, including problem formulation, the main algorithm, the network architectures and their design rationales. Experiments and results analysis are presented in Section 3. Section 4 discusses challenges and restrictions for evaluation and demonstrates how this work is evaluated in a real production environment. Conclusion and future work are discussed in Section 5.

2. Methodology

2.1. Problem analysis and formulation

In the literature, the optimal control of a GC process is usually modelled as a dynamic system either through direct dynamic system modelling (Chen, Zhai, Li, & Fei, 2007; Herbst, Pate, & Oblad, 1992; Yianatos, Lisboa, & Baeza, 2002) or data-driven models (Guo, Wang, & Zhang, 2019; Lu, Kiumarsi, Chai, Lewis, & Control Theory, 2016; Yin, Gao, & Kaynak, 2014), since many setpoints determine the production performance together. For this work, as explained earlier, only the feed load setpoint is considered for production performance (as required by the production process). Hence the whole GC process can be formulated as a probability distribution: $P(Q_{t+n}|s_t, O_t)$, where s_t is the feed load setting at the time t, O_t is a set of latent variables that denote the ore pellets' characteristics (type, hardness, sizes) at time t, and $P(Q_{t+n}|s_t, O_t)$ denotes the probability measure of final product quality values at a later time $t + n$, given s_t and O_t. In addition, a GC process typically has a product quality threshold (q_m), which governs the whole process. It is the most important constraint of the system that has substantial economic impacts. Other production performance indicators, such as product throughput, get measured only when the q_m constraint is satisfied. Putting everything together, we can write the objective as:

$$argmax_{s_t}(s_t * P(Q_{t+n} = q_m|s_t, O_t)) \qquad (1)$$

Eq. (1) can be rewritten to:

$$argmax_{s_t}\left(s_t * \frac{P(s_t, O_t|Q_{t+n} = q_m) * P(Q_{t+n} = q_m)}{p(s_t, O_t)}\right) \qquad (2)$$

As q_m is the constraint that must be satisfied, to maximise the Eq. (2), we could make $P(Q_{t+n} = q_m) = 1$. The denominator, $P(s_t, O_t)$ can also be ignored as it is for the normalisation purpose. Therefore, Eq. (2) is simplified to:

$$argmax_{s_t}(s_t * P(s_t, O_t|Q_{t+n} = q_m)) \qquad (3)$$

We still need to work out the value of O_t and this is where the ore

pellet images come into play. The assumption made here is that O_t can be approximated by a function $w(I_t) \approx O_t$, where I_t denotes the image captured at the time t. After substituting O_t with $w(I_t)$, we drive the final objective function:

$$argmax_{s_t}(s_t * P(s_t, w(I_t)|Q_{t+n} = q_m)) \qquad (4)$$

Maximising Eq. (4) is equivalent to maximising $P(s_t, w(I_t)|Q_{t+n} = q_m)$, as after being normalised, the variance of the component s_t is small and has little impact on the result of the whole equation.

It is not difficult to imagine using a neural network model for estimating this distribution. However, the key challenge for implementing $P(s_t, w(I_t)|Q_{t+n} = q_m)$ arises from the difficulties in collecting enough and appropriate (s_t, I_t) samples that make statistical sense. We have briefly discussed this issue earlier. Q_{t+n} is generated every 4 h, which means, maximumly, only six image samples can be collected. It should be noted that even for these six images, their corresponding Q_{t+n} values might be smaller than the q_m threshold and therefore cannot be used for solving Eq. (4).

Furthermore, another more challenging issue is from the potential size of an image sample (I_t). The product generated at $t+n$ is ground from a cluster of ore pellets, which requires a huge image receptive field (much more than 20 m long) to cover (shown in Fig. 3). It is practically infeasible to use a single image as a representation for the involved ore samples. Sequence analysis of a video clip neither suits this application, as there is no actual temporal relation between the ore pellets from each video frame.

To address the above two issues, we propose to use a set of smaller images $\mathscr{P}_t = [\rho_i | \rho_{t-m}, \cdots, \rho_{t+m}]$ to replace I_t. The $\mathscr{P}_t$ can be understood as subsamples of I_t [2] (a set of small patches from a large image). In this way, we can also collect substantially more images (2 m times more per day) for later training. After substituting I_t with $\mathscr{P}_t$, the $w(I_t)$ becomes $w(I_t) = w_1(w_p(\rho_i)), s_t$ becomes $s_t \approx \frac{\sum_{i=t-m}^{i=t+m} s_i}{2m}$ and the Eq. (4) becomes:

$$argmax_{s_t}(s_t * P(s_t, w_1(w_p(\rho_i))|Q_{t+n} = q_m)) \qquad (5)$$

which can then be transformed to:

$$argmax_{s_t} * \omega(P(s_{1t}, w_p(\rho_i)|Q_{t+n} = q_m)) \qquad (6)$$

We can see, in Eq. (6), $\omega(P(s_t, w_p(\rho_i)|Q_{t+n} = q_m)$ is used to replace the $P(s_t, w_1(w_p(\rho_i))|Q_{t+n} = q_m)$ in Eq. (5). We believe there exists a function ω, which approximates the probability of $P(s_t, w_1(w_p(\rho_i))|Q_{t+n} = q_m)$ using a list of joint probabilities of $P(s_t, w_p(\rho_i)|Q_{t+n} = q_m)$.

At last, since we are unable to find the value of $P(s_t, w_p(\rho_i)|Q_{t+n} = q_m)$ for each ρ_i, we first assume they share the same probability with I_t as a starting point.

$P(s_t, w_p(\rho_i)|Q_{t+n} = q_m) = P(S_t, I_t|Q_{t+n} = q_m)$ (asm. 1).

This assumption gets refined in the next step of our work with weakly supervised learning methods.

2.2. Training models with Weakly-Supervised learning

To solve the function ω, we could use a convolutional neural network (CNN) (Lecun & Bengio, 2021) model. In the ideal situation, to train such a model, we can collect enough small images patches (ρ_i) and label each of them with a s_t. All unique s_t values (generally under 20 in the actual production line) can be considered as classes that the model tries to classify. A SoftMax function is then used to generate a probability for each s_t class, and we only need to choose the one with the maximum value. However, because of the assumption (asm. I) made, the training sample (p_i, s_t) may not satisfy $Q_{t+n} = q_m$, or in other words, the p_i has an incorrect annotation. Training a model using data with wrong labels

surely degrades the model performance. In the machine learning community, there have been many efforts in training a deep learning model with incorrectly labelled or unlabelled samples with weakly supervised learning approaches(Zhou, 2018; Li, Guo, and Zhou 2021; Oquab, Bottou, Laptev, & Sivic, 2015; Engelen, 2019; Zhai, Oliver, Kolesnikov, & Beyer, 2019). Such methods fit nicely into our situation.

We can define the training process formally as a weakly supervised learning problem. The task is to learn two functions:

$$\tau : \rho_{i,t} \rightarrow s_{i,t}; \theta \text{ and.} \omega : [\tau(\rho_{i,t})] \rightarrow s_t; \widehat{\theta}$$

(θ and $\widehat{\theta}$ denotes model parameters) from a training dataset:

$$\mathscr{D} = \left\{ (I_{t_1}, s_{t_1}) \cdots, (I_{t_n}, s_{t_n}) \right\},$$

$$where I_t = \left\{ \rho_{1,t}, \rho_{2,t}, \cdots, \rho_{i,t} \right\}$$

I_t is defined as a bag and $\rho_{i,t}$ are the instances from this bag. I_t is a positive bag, i.e., $\omega(I_t) = s_t$, if the fusion of all its instances $\rho_{i,t}$ is positive ($\omega(\tau(\rho_{i,t})) = s_t$). The goal is to learn model parameters θ and $\widehat{\theta}$ that maximise the data likelihoods:

$$\tau = \underset{\theta}{argmax} \prod_{\rho_{i,t}, s_{i,t} \in \mathscr{D}} P(\rho_{i,t}, s_{i,t}|\theta) \qquad (7a)$$

$$\omega = \underset{\widehat{\theta}}{argmax} \prod_{\rho_{i,t}, s_t \in \mathscr{D}} P(\tau(\rho_{i,t}), s_t|\widehat{\theta}) \qquad (7b)$$

To update θ and $\widehat{\theta}$, we propose an algorithm that is based on the other two algorithms, namely, the expectation–maximisation (EM) algorithm (Moon, 1996) and the k-means++ clustering algorithm (Arthur & Vassilvitskii, 2006). The k-means++ algorithm is applied first to find out the correct classes in stage one, followed by the EM algorithm for updating $\widehat{\theta}$ values in stage two.

2.3. The algorithm: stage one

Human operators tend to use rough numbers for setting the feed load value, such as 125,130 and 145. Although these settings may satisfy the product quality constraint (q_m), there may exist better values which, although not being reflected in the training data, can maximise the Eq. (7a) better. To verify this, we first apply a k-means algorithm to find the best possible clusters/classes .

1 Initial E Step: At the initial step, we initialise all the instance labels with the bag's label (I_t, s_t) from $\mathscr{D}$, and form the dataset :

$$\mathscr{D}_1 = \left\{ \left(p_{i,t}, s_t\right)_t, \left(p_{i,t_1}, s_{t_1}\right)_{t_1}, \cdots, \left(p_{i,t_n}, s_{t_n}\right)_{t_n} \right\} \text{ for all possible.} I_t \in [I_{t_1},$$
$$I_{t_2}, I_{t_3}, \cdots, I_{t_n}]$$

M Step: We then update the parameter θ to maximise Eq. (7a) using the data from $\mathscr{D}_1$ till convergence.
E step: Through applying $\tau(\rho_i; \theta)$, we extract features f_i for each ρ_i; store them with ρ_i's original label and form a new dataset:

$$\mathscr{D}_k = \left\{ (f_{i,t}, s_{i,t})_t, (f_{i,t}, s_{i,t_1})_{t_1}, \cdots, (f_{i,t_n}, s_{1i})_{t_n} \right\}$$

The k-means++ algorithm is then applied to $\mathscr{D}_k$ to form clusters with a range of initial k values.
We use the k-elbow method to select the best k value and perform k-means++ with this value to form clusters, $\mathscr{C}_k$.
For each cluster, $c_i \in C_k$, we update the corresponding $\left(\rho_{i,t}, s_{i,t}\right)$ in $\mathscr{D}_1$ with the mode value and generate the relabelled dataset :

$$D_1 = \left\{ \left(p_{i,t}, s_{i,t}^{\wedge}\right)_t, \left(p_{i,t}, s_{i,t_1}^{\wedge}\right)_{t_1}, \cdots, \left(p_{i,t}, s_{i,t_n}^{\wedge}\right)_{t_n} \right\}$$

---

[2] Unless otherwise indicated, in the rest of the paper, I_t refers to the "virtual large image", which is a collection of patch-images taken by the camera.
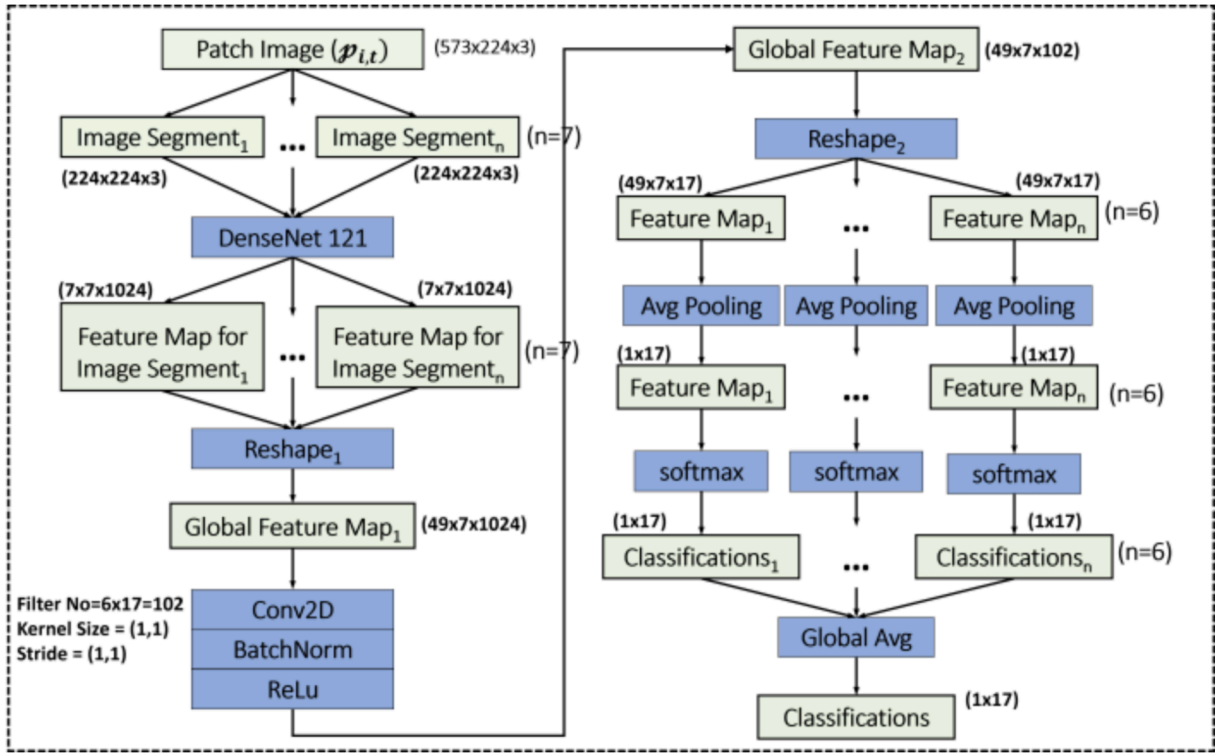
Fig. 4. Network architecture of the patch prediction model.

We iterate back to step 2 till convergence. It should be noted, if new classes are generated in step 6, θ needs to be re-initialised.

The main goal for stage one is to regenerate labels for image patches $p_{i,t}$ automatically using their structural similarity and discover new labels that do not exist in the original dataset. In addition, a model is trained for making patch level predictions.

2.4. The Algorithm: stage two

In stage two, we start with a new CNN model instance, $\omega(s_t; \tau(\rho_{i,t}), \widehat{\theta})$ that is trained using outputs from $\tau(\rho_{i,t})$.

We use the model, τ, from stage one to transform all $\rho_{i,t}$ into $\tau(\rho_{i,t})$ and form a new dataset $\mathcal{D}_t$:

$$\mathcal{D}_t = \left\{ \left(\{ \tau(\rho_{1,t}), \cdots, \tau(\rho_{i,t}) \}, s_{t_1} \right), \cdots \right\}$$

We update the parameter $\widehat{\theta}$ to maximise Eq. (7b) using the data from $\mathcal{D}_t$.

The second function ω fuses the representations of all instances ($\rho_{i,t}$) for a bag I_t. The representation format it takes depends on the outputs of the first model τ. If the outputs of τ are target labels (s_t), the ω function can be as simple as a weighted average function or even a max-pooling function. The τ function, on the other hand, can produce more comprehensive representations of $\rho_{i,t}$. In this case, ω acts as a proper fusion model that combines all features derived from ($\rho_{i,t}$) and makes predictions.

2.5. Model architectures

Unlike the other classification/segmentation tasks, ore pellet images

do not contain straightforward features that a model can easily learn. For a task like this, significant amounts of training data are required for the model to pick up useful features. Although augmented data can be used, the number of final training samples still depends on the number of images collected before augmentation. Another approach for increasing training data size, especially image data, is using image segments [3] (Oga et al., 2020). This approach naturally fits our application. It is reasonable to assume that for a single ore pellet image patch, its segments have a similar distribution to the whole image. Also, training a model using overlapped segments has good augmentation and regularisation effects. A possible side-effect is that we may lose some global features at the image level. However, this can be overcome by combining feature maps from all segments to form global feature maps.

Fig. 4 shows the complete network architecture that we used for approximating function τ. The boxes in lime are data passed into/out from each network layer, and the boxes in blue are network layers. The network's input is a set of patch image segments, and the network's output is one-hot-encoded classification results. The network has three main blocks: the encoder block, the feature aggregation block, and the multi-head classification block.

2.6. CNN architecture for patch prediction (Function τ)

We use 121 layers DenseNet (Huang, Liu, van der Maaten, & Weinberger, 2016) block as the encoder for feature extraction for each input segment. We choose DenseNet over the other popular network architectures such as ResNet (He, Zhang, Ren, & Sun, 2015) because, with DenseNet, low-level features are pushed to the top of the network and the high-level ones. We find this property important for this work, as the size and colour of individual ore pellet may be key factors for making classification decisions. On the other hand, we still need filters with

---

[3] We use the term "*segment*" here to distinguish ourselves from the "*patch*" term used in the previous section. It refers to the image patch concept in others' work.
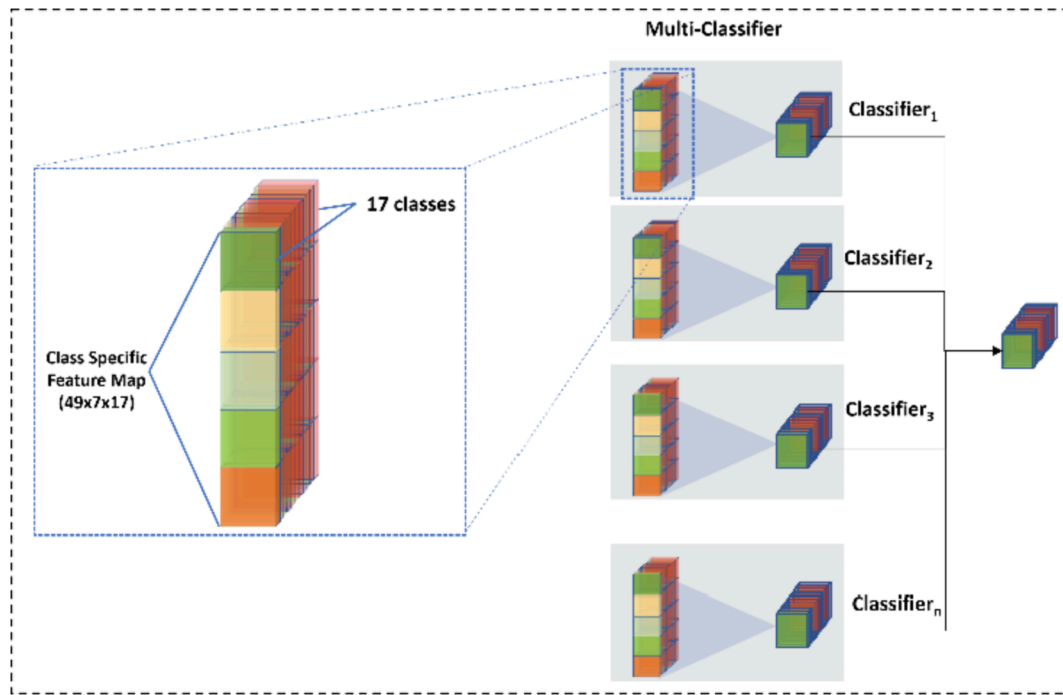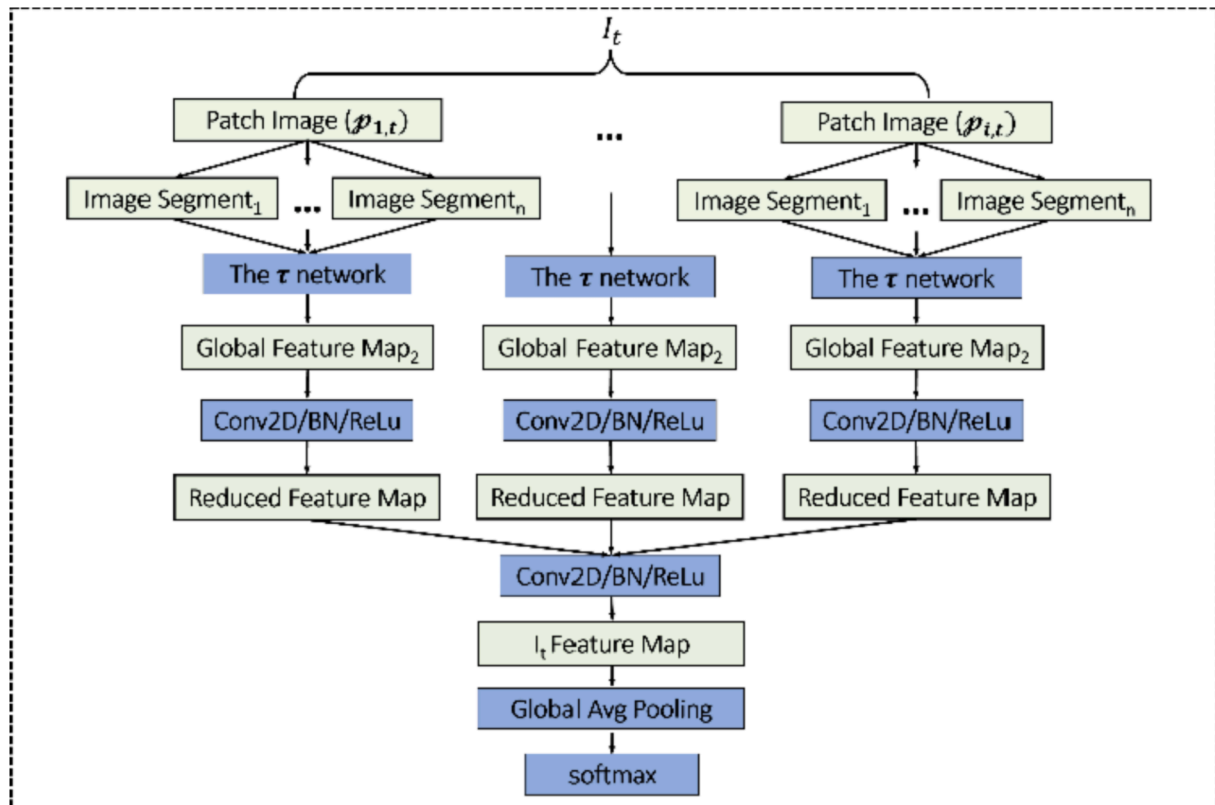
Fig. 5. Enforcement of class-specific feature maps.

Fig. 6. Network architecture of the image (I_t) prediction model.

larger receptive fields that provide global views of the segment.

It is important to point out that the DenseNet block is applied to all the incoming segments simultaneously. There is only one DenseNet block instance in our network. During the training process, the weights of the DenseNet block get updated using all the gradients backpropagated from its next layer. Unlike that with a sequence model, no specific order is required for the input image segments. This can be understood as: the DenseNet block is trained N times in one backpropagation iteration with N data samples. Similarly, the same filters from the DenseNet block are applied to all segments for feature
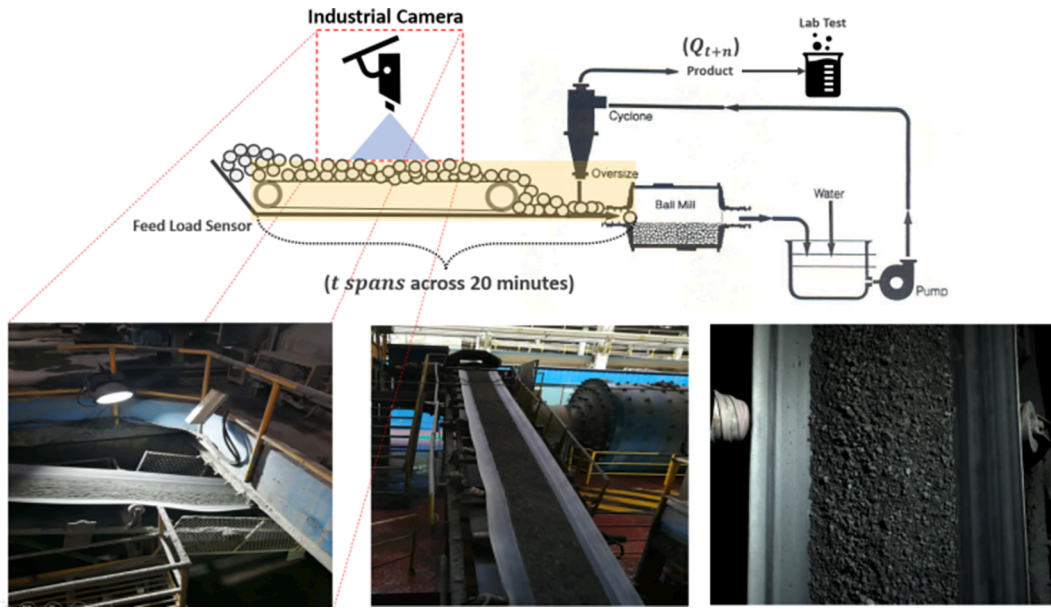
Fig. 7. Experiment setup for data collection.

extraction and generate their feature maps separately during the feed-forward phrase.

The feature maps of segments are aggregated into a global feature map (Global Feature Map1), which then goes through a 2D convolutional (conv2d) layer with (1,1) kernel. The conv2d layer combines filter (1024) activations at different scales into the global ones for each receptive view (Global Feature Map2). Afterwards, the final global feature map2 is reshaped to multi-head feature maps, with each one corresponding to a classifier.

Fig. 5 gives an in-depth look at the multi-head classification block. There are two design rationales behind this. First, the features contributing to a particular class are aggregated together, which helps later model interpretability analysis through visualisation. Second, multi-head structure helps with the model generalisation and robustness. We observed that the multi-head classifier yields better classification results than a single classifier through our experiments.

2.7. CNN architecture for I_t prediction (Function ω)

The second network is for approximating the function ω (shown in Fig. 6). It takes multiple global feature map2 that are output by the first network and fuses them into one feature map for I_t. Due to the size of global feature map2 and the number of patch images involved (must be big enough to cover the whole I_t receptive field), we further reduce the feature map's size before they are combined into the final "I_t feature map" through a convolutional layer.

There are two main reasons for using features maps from multiple patch images. First, although the first network predicts a class label for each patch image, the prediction results cannot be combined with mechanisms like voting or average pooling (Cruz-Roa et al., 2014; Poria, Cambria, & Gelbukh, 2015; Seff, Le, Cherry, Roth, Liu, Wang, Hoffman, Turkbey, & Ronald, 2014) g, as some class labels are generated from the clustering analysis and do not represent any real production setting.

This is illustrated in more detail in the following sub-section. Also, patch level predications contain errors, which are brought to the final prediction if used directly. At last, the patch level model might be biased due to the smaller receptive field adopted, and the image level model may learn to correct the biased cases that occurred at the patch level.

3. Experiments

We evaluate our method through experiments in a real production environment. We use the Tensorflow (Abadi, Barham, Chen, Chen, Davis, Dean, & Devin, 2016) framework for our model implementation, and the training of both models are performed on an Nvidia RTX 3090 GPU. Fig. 7 shows the basic experimental setup.

3.1. Data collection and pre-processing

To train the two models, we collected three types of data in 14 months between June 2020 and August 2021. The first type of data is the lab test result (Q_{t+n}) of the final product quality, the second type is the feed load data (s_t) extracted from the equipment sensor readings, and the third type is the image ($\varkappa_{i,t}$) data taken by an industrial camera mounted perpendicularly to the conveyor belt.

All three types of data are timestamped. Based on the minimum product quality requirements, the lab test results that are below 66 are dropped. It ensures the condition $Q_{t+n} = q_m = 66$ in Eq. (6). For each filtered test result Q at time t + n, we find its corresponding feed load data at time t with the value of n set to 90 min. It is worth mentioning that the data stamped at time t is not a single sample reading but an average reading of a collection of samples. The sampling rate of the feed load sensor is one minute, but it takes around 20 to 30 min for the conveyor to transmit ore pellets for producing the final product at the time Q_{t+n}. Therefore, we aggregated all feed load sensor readings within that 20-minutes window into a single reading s_t. The same principle applies to the image data. Our camera captures ore pellet images (1024 × 400 × 3) with the same sampling rate as the feed load sensor. All images ($\varkappa_{i,t}$) in the same 20-minutes window are grouped to form the virtual image (I_t), and are all annotated with a label s_t initially.

In total, we collected 29,952 samples (($\varkappa_{i,t}, s_t$)) as the dataset D_1 for training/validating the first model. From the datasets, we extracted 17 classes (0, 120, 125, 130, 132,135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146) using unique s_t values. The class '0' indicates no ore pellets on the conveyor, while each of the rest represents a possible feed load setting value. The patch images are all downsized to 573x224x3. Seven segments of size 224 × 224 × 3 are then extracted from each patch image with a stride value of 56 pixels. For the data augmentation purposes, segments are randomly flipped either horizontally or vertically, and the brightness and contrast values of the segments are also
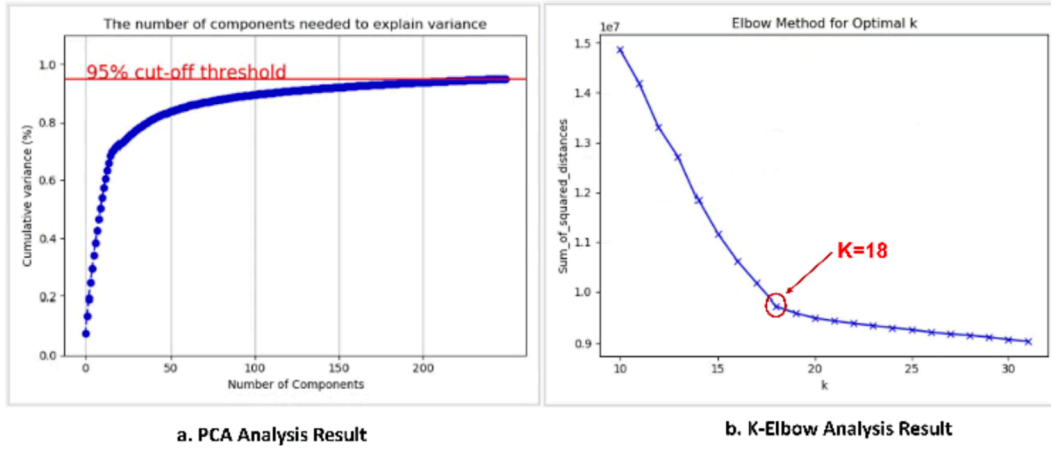
a. PCA Analysis Result

b. K-Elbow Analysis Result
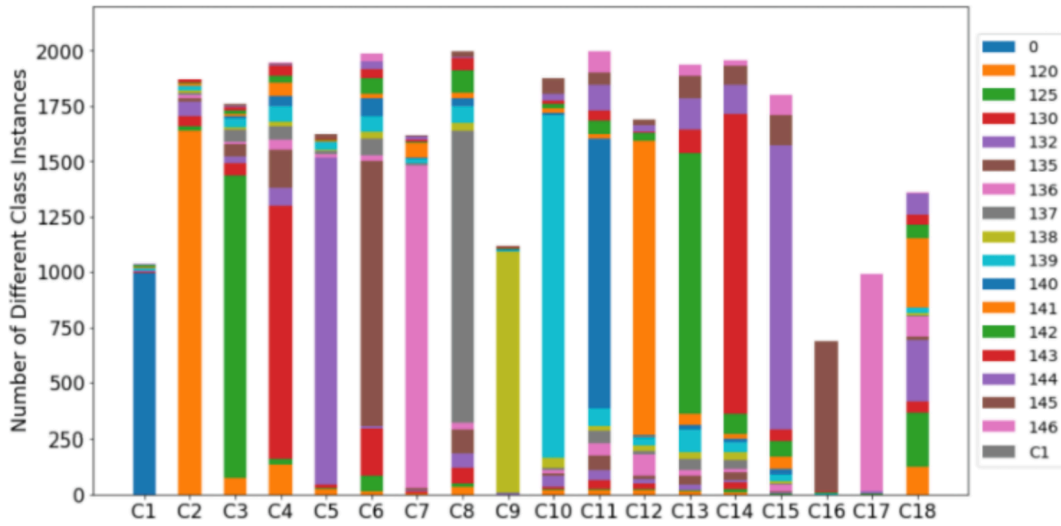
Fig. 8. The PCA and K-Elbow analysis results.

Fig. 9. Number of class instances within each cluster.

randomly adjusted between 0.5 and 1. 80% of data are sampled uniformly for training, with the remaining 20% for testing. It should be noted that we have applied the 10-fold cross validation approach to this work as well. Due to the paper space limit and the very closed result for each fold, we only take one fold as an illustration example in this paper. The process discussed in the following subsections has been applied to all folds.

3.2. Training the first model

Following the proposed algorithm, we start with training the model using the initial dataset D_1. The Stochastic Gradient Descent (SGD) (Bottou, 2010) is used as our optimiser with an initial learning rate of 10^{-2} and the momentum value of 0.9. The learning rate is gradually reduced to 10^{-4} with a deduction rate of 0.7, given that there has been no improvement in the past five epochs. The batch size used for training is 32. The training process converged around epoch 80, with the best validation accuracy of 87.82%. We then used this provisional model for extracting features from all patch images in D_1 and performed the principal component analysis[30] for feature reduction. We choose to keep the first 250 components as they explain 95% variance of the original feature space (Fig. 8. a).

A series of k-means++ clustering analysis were applied afterwards with the value of $k \in [10,31]$. The k-elbow method shows the best k value

is 18 (Fig. 8. b), which indicates there exist one additional possible class on top of the 17 initial classes from our dataset.

Fig. 9 shows that for the first 17 clusters (C1-C17), most of their members have consistent class labels corresponding to the original ones. However, for cluster C18, the class labels of its members are diverse. Such facts suggest that the patch images inside the C18 cluster have significant structural differences with the images from the other clusters; hence the initial class labels assigned to them may be inaccurate. Therefore, we reconstructed another dataset D_{1u} via regrouping the patch images based on their clusters and used the cluster labels as their class labels. $D_{1u} = \{ \left(\mathscr{N}_{1,t}, C_1 \right), \cdots, \left(\mathscr{N}_{i,t}, C_n \right) \}$.

We trained and validated a new network model using the D_{1u} dataset (with 80%/20% split) in four iterations. This model is like the first one, apart from being used for an 18-classes classification task. After each iteration, we relabelled the patch images using the predicated class labels and updated the dataset D_{1u} for the next iteration. The training accuracy for every class reached 100% after the fourth iteration, and the overall validation accuracy of the model increased from 88.12% (after the first iteration) to 96.45% (after the fourth iteration). We tried to conduct clustering analysis between each iteration using the latest model, but the number of generated clusters remained 18, and the cluster members did not vary significantly.
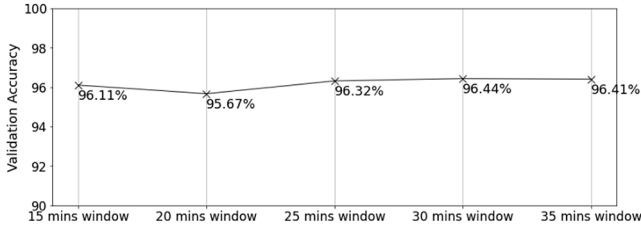
Fig. 10. Test accuracies for model instances trained with five different window sizes.

Table 2
Classification results of the best model.

Classes	Precision	Recall	F1	Most Misclassified
0	1.00	1.00	1.00	–
120	0.96	0.99	0.98	143: 0.1%
125	1.00	0.98	0.99	137: 0.67%
130	0.95	0.98	0.96	135: 0.54%
132	0.97	0.97	0.97	120: 1.2%
135	0.95	0.97	0.96	130: 1.56%
136	0.96	0.96	0.96	139: 0.56%
137	0.96	0.96	0.96	139: 0.61%
138	0.97	0.97	0.97	139: 2.31%
139	0.95	0.94	0.94	138: 1.06%
140	0.91	0.98	0.95	130: 0.63%
141	0.92	0.98	0.95	136: 1.15%
142	0.96	0.94	0.95	140: 1.74%
143	0.96	0.97	0.96	135: 0.69%
144	0.89	0.96	0.92	141: 0.82%
145	0.98	0.79	0.87	144: 7%, 141: 6.7%
146	1.00	0.89	0.94	144: 6.75%

3.3. Training the second model

For the second model, we grouped patch images ($\mathscr{p}_{(i,t)}$) from the dataset D_1 into (I_t, s_t) pairs for five different datasets $D_2 = \{D_{2,15mins}, D_{2,20mins}, D_{2,25mins}, D_{2,30mins} D_{2,35mins}\}$. In practice, only a rough estimation

is instructed on how long a ball mill is filled from empty to full. The ball mill is filled quicker for higher feed-load settings, while it takes longer for the lower feed-load settings. Therefore, we must try different window sizes to determine the best size for I_t. It also should be noted that using datasets with different input sizes has little impact on the first model, as such differences only cause small vibrations in the sample size of D_1. It is reasonable to assume that the performance of the first model at the patch image level remains the same for different D_2 s. We trained five instances of the second model, each using a different D_2 dataset. The same hyper-parameter values for training the first model are used for training all five instances.

Fig. 10 shows no significant difference between the instances' performance, with the best test accuracy (96.44%) achieved for the 30-minutes window and the worst (95.67%) for the 20-minutes window.

We also used the precision, recall, and F1 score to measure each class's model's performance. The results (only for the best model due to the paper space limits) are shown in Table 2. It also shows the proportion of most misclassified class instances against each ground-truth class, from which we can see that the worst two recall values are from the classes 145 (79%) and 146 (0.89%). 13.7% of the testing 145 class are misclassified as class 144 (7%) and class 141 (6.7%), while 6.75% of class 146 are misclassified as class 144.

There are three possible reasons for the descending performances of these three classes (144,145,146). First, the dataset may be unbalanced, which is not our case. Second, the features of these three classes are too close to be differentiated. If this is the case, further refinements (model architecture change, parameter turning) may be necessary for better feature extractions. However, as the precision values of these classes are high, which indicates the number of false positive cases is low, we have reasons to believe that the features used for classification are well learnt.

Another possible reason is the wrong data labelling, which is more likely to be our case. In the production environment for our work, it is not an unusual case that a human operator leaves the feed load unchanged even if the final product quality is a lot better than the quality threshold ($Q_{t+n} > 66$). For such circumstances, the data labelling ($(\mathscr{P}_{i,t}, s_t)$) is incorrect, as there should exist a larger feed load ($s'_t > s_t$) that can be used as the correct label for $\mathscr{P}_{i,t}$. Unfortunately, it is not
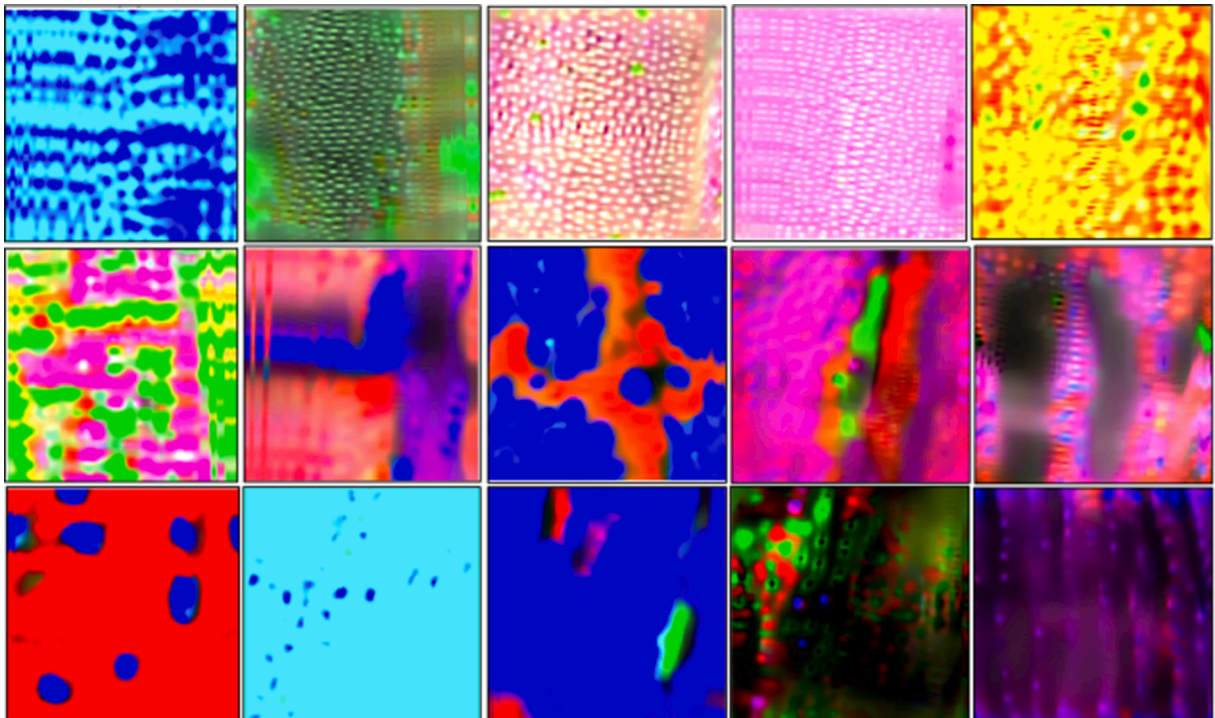

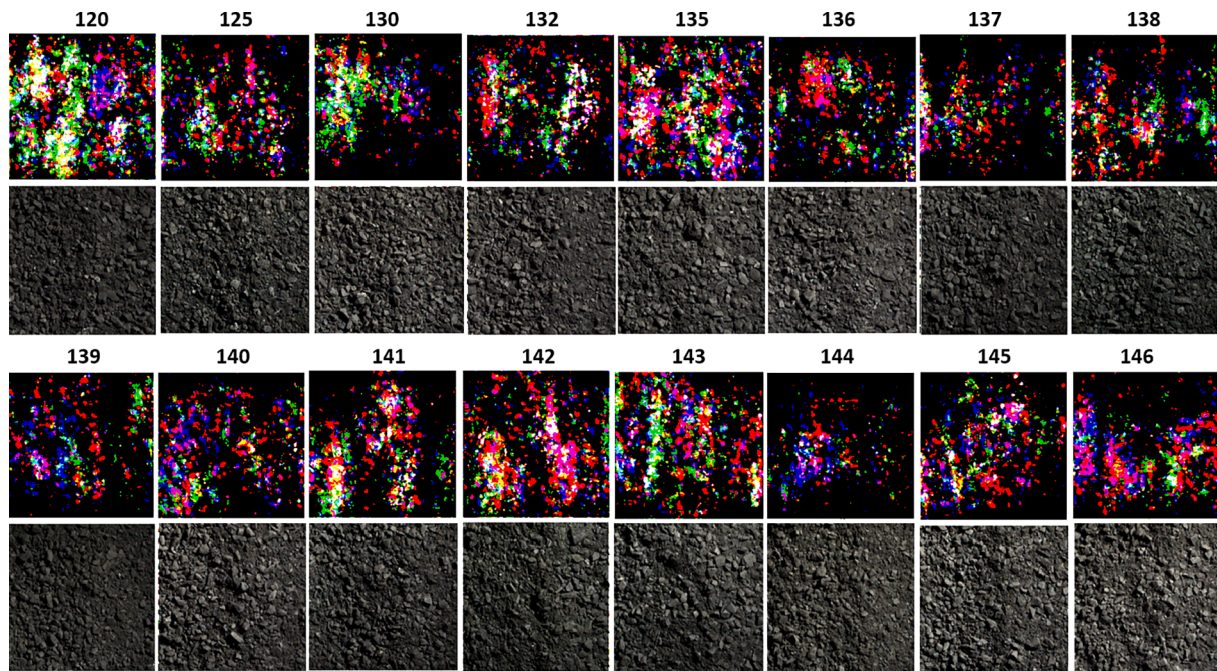**Fig. 11.** Visualisation of samples in the global feature Map2.

Fig. 12. Saliency maps for the best true positive class samples. The top image is the saliency map for each class, and the bottom is the ore pellet image. All the saliency map images are adjusted for their sharpness to give better visibility.
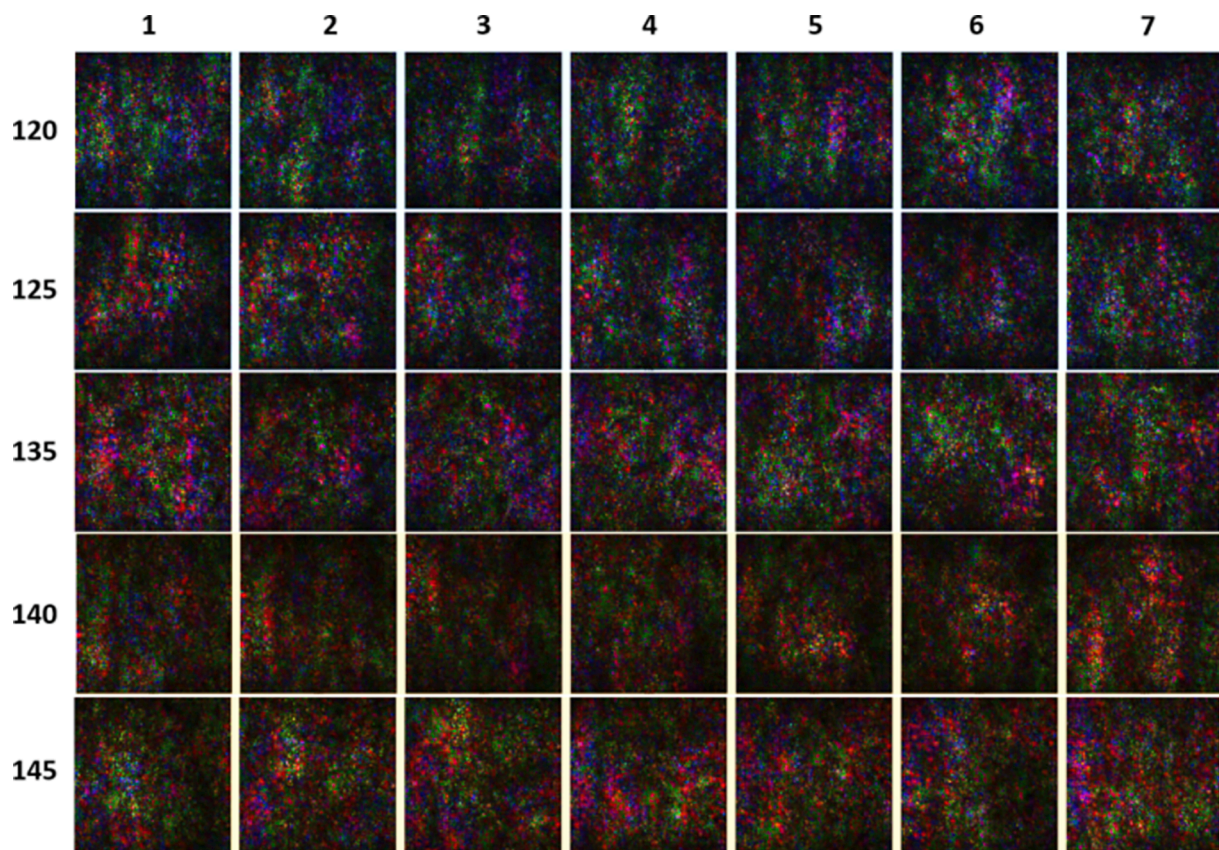

Fig. 13. Saliency maps for seven random samples of five classes.

possible to find the value of $s_t^{'}$ as it is never set in the system.

3.4. Model interpretability and visualisation

Even though the model shows promising classification results with the testing datasets, it is still essential to understand what the model has
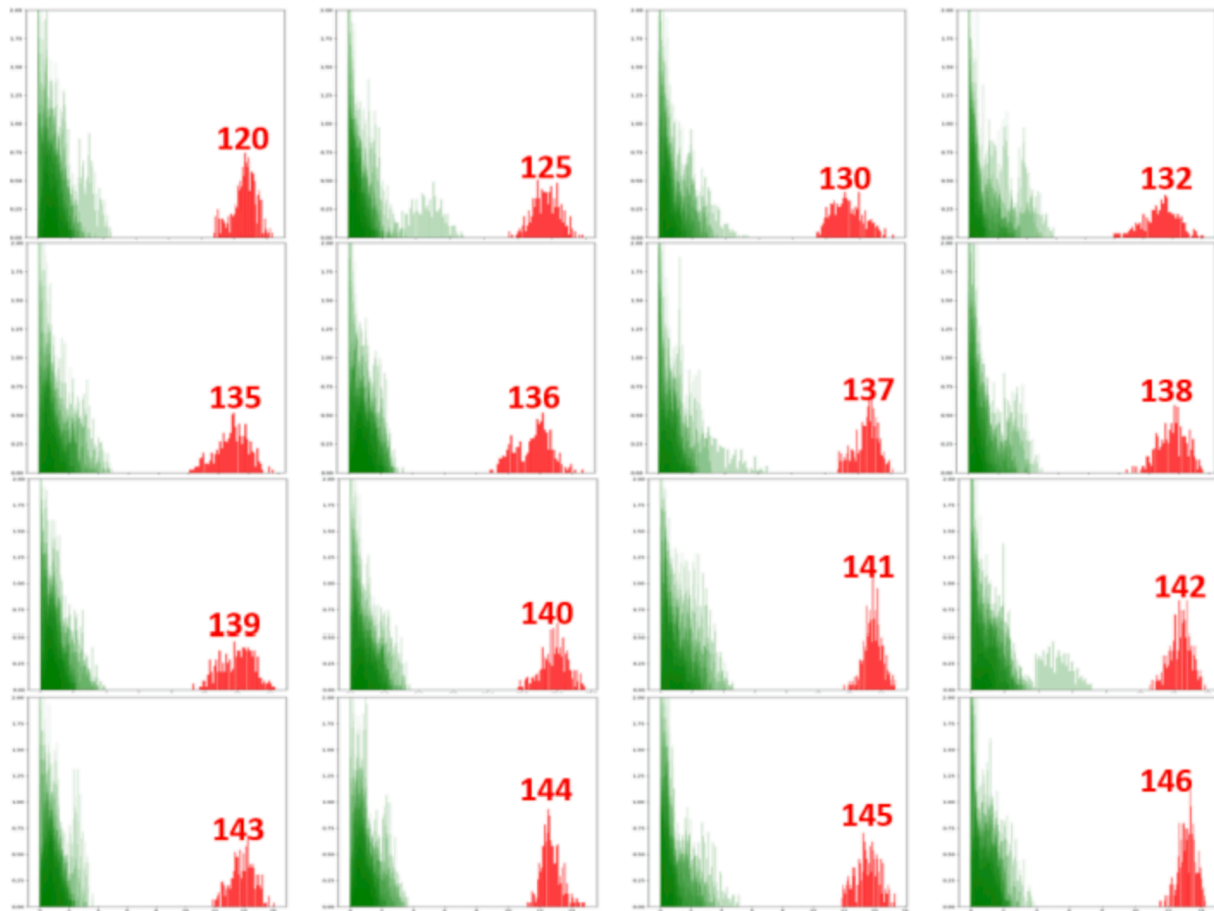
Fig. 14. Distributions of the enforced class-specific feature maps activated by the best true positive samples.

learnt and how it makes its decision. To visualise what the model has learnt, we adopt the activation maximisation method (Bengio, Courville, Erhan, Bengio, & Vincent, 2009). We first visualise the "global feature map2" from the first model, as it aggregates all features learnt before they get enforced into class-specific feature maps.

From sampled feature maps shown in Fig. 11, we can see that, to make a prediction, the model considers several factors at both ore pellet level and patch image level. The top row shows that the model focuses on the ore pellets level regarding their colour, sizes, and overall quantities, whereas the feature maps in the second row focus more on global shapes at the patch image level. Those shown in the third row are mainly colour channels. Since the following layers combine all these different features from the network for making predictions, it is reasonable to conclude that these are the key factors that affect the final classification result. It is consistent with the knowledge from experienced mineralogists who can tell ore's characteristics via looking at their sizes and colours.

To give a more intuitive impression of what the model looks at for the final classification, we have also used the saliency map(Simonyan, Vedaldi, & Zisserman, 2013) to outline the model's attention areas (shown in Figs. 12, 13). For each class in Fig. 12, we selected the best true positive sample (the image with the correct and the highest classification score) for maximising the saliency map. The image channels (RGB) are kept for this visualisation task. For all saliency maps, image areas/channels with significant contributions to the classification are highlighted. To show consistency of the enacted features for each class, in Fig. 13, we show the significant activations of seven randomly sampled images for each class. Due to the paper space limit, we only use five classes for the illustration purpose, as they are more friendly for human-eyes interpretation. The different colours in the maps potentially highlight different types of ores (hematite: red/purple, chlorite: green, limonite: brown and magnetite: dark). As hematite and limonite are easier to grind and contain higher amounts of iron, the feed load can be set higher for ore pellets mainly of these two types. In contrast, for chlorite (containing less iron) and magnetite (hard to grind), the feed load needs to be set lower to ensure the same product quality as the other two ore types. The generated saliency maps are aligned with such facts. For ore pellet images with lower feed load settings (from 120 to 132), a good proportion of highlighted areas are coloured green, while for middle and high range feed load settings (from 135 to 146), red/purple/pink colours become dominant.

At last, we show how the model makes classification decisions based on the learnt features. We plot the distribution of its "enforced class-specific feature map" for each class, using the best and the worst true positive samples, respectively. For the best cases (Fig. 14), there are clear decision boundaries between the feature distributions of the predicated class and the other classes, whereas, for the worst cases (Fig. 15), such decision boundaries are less determinate.

4. Evaluation

The difficulties in evaluating this work stem from several facts. First, there is no standard bench-test dataset available. Image datasets from the literature (Perez et al. 2011a; Tessier, Duchesne, & Bartolacci, 2007) are taken of ore rocks before crushing. Comparing our work with those does not lead to meaningful conclusions. Second, to the best of our knowledge, our work is the first of its kind that estimates feed load setting for the optimisation purpose. Most of the existing work (Dong & Jiang, 2014; Donskoi et al. 2007; Donskoi et al., 2018; Mukherjee, Potapovich, Levner, & Zhang, 2009; Mustafa, Zhao, Liu, Zhang, &
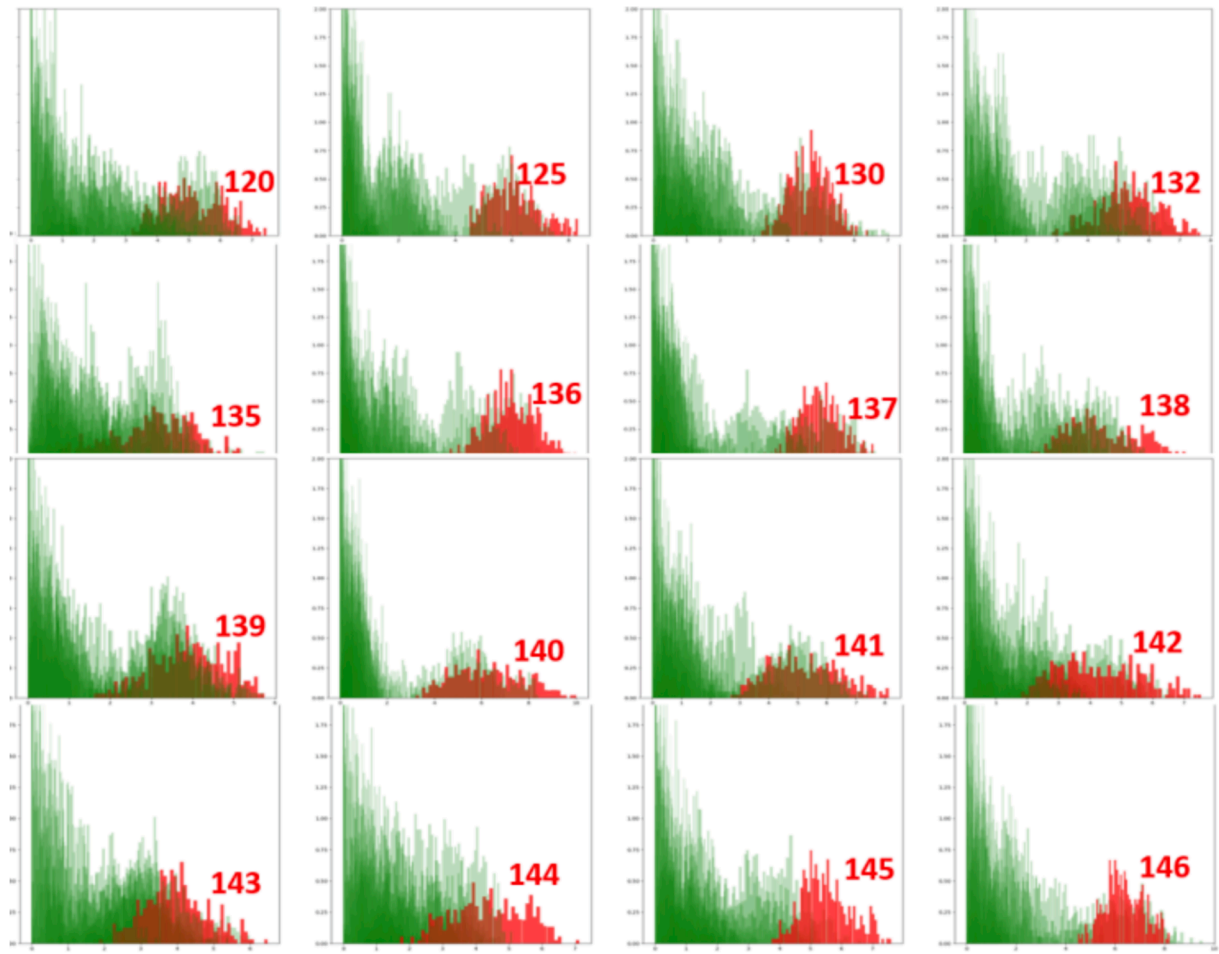
Fig. 15. Distributions of the enforced class-specific feature maps activated by the worst true positive samples.

Wenxian, 2020; Perez et al., 2011; Tessier et al., 2007) are focused on classification or segmentation of ore grade, ore shape and ore characteristic, hence are neither directly comparable. Third, it is infeasible to use our model in a separate production line and compare its performance with that from a human-operated environment due to the risk of potential economic loss if the model fails to work. This restriction makes it hard to carry statistical tests. Finally, in the actual production environment, the feed load adjusted by human operators are often less
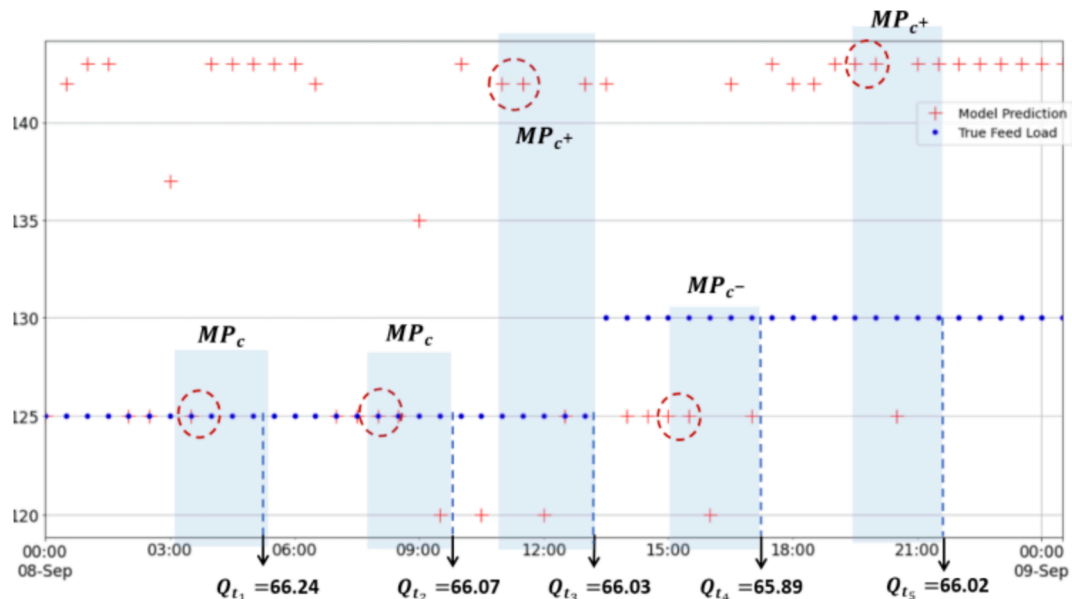

Fig. 16. Comparison result between the model prediction and the true feed load setting.

optimal. Comparing the model prediction results with actual feed load settings may lead to confusion. For instance, a feed load setting of 125 tons may produce a satisfiable product quality ($Q_{t+n} \geq 66$). A higher feed load setting may also meet this condition but is never set by human operators. In this case, our model may give a correct optimal estimation with a higher feed load setting but is different from the ground truth label.

Due to the above issues and restrictions, we have to use a compromised evaluation strategy. Between 7th and 30th September 2021, we ran our model in parallel with the existing grinding production system and collected 23 days of data for evaluation. For every lab test result, we compare the actual feed load and our model prediction (both are from 2 h earlier before the lab test time). For lab test results that meet the minimal product quality requirement (≥ 66), if our model prediction value is equal or greater than the true feed load setting, we consider it as a correct (MP_c) or possible correct prediction (MP_{c+}), as if the actual feed load was adjusted higher, we may still achieve the same product quality but with better productivity. For lab test results that miss the minimal quality requirement, if our model prediction value is smaller than the true feed load setting, it is also considered as a possible correct prediction (MP_{c-}) as lowering the feed load may lead to better product quality. All other cases are considered as wrong predictions (MP_w).

Fig. 16 shows the comparison results between our model and the actual feed load setting for a single working day. For the first two lab test results ($Q_{t_1} > 66, Q_{t_2} > 66$), our model agreed with the human operator's feed load setting at 125. For the third lab test results ($Q_{t_3} > 66$), the model believed the feed load should be set to 142, while the human operator left it to 125. Interestingly, the human operator lifted the feed load to 130 after the third lab test result was generated, as he probably noticed that all the last three Q_{t}s are satisfactory hence trying to increase the throughput. Unfortunately, such a change led to poor product quality, as reflected by the lab test results $Q_{t_4} < 66$. The model, however, suggested setting the feed load lower at 125 for that period. For Q_{t_5} greater than 66, there could be a throughput increment based on the model's output (set feed load to 144), but the actual operation just left everything untouched.

In summary, for the whole evaluation period (24 days), we collected 144 lab test results, with 89 of them greater than 66 and the rest 55 are less. It indicates that the human operators were only able to keep the product quality qualified in 61.8% of the time, not talking about whether the throughputs for these qualified products are optimal. In total, human operators made 67 changes to the feed load setting, which is on average 2 to 3 times a day. In contrast, our model suggested 577 feed load changes (if two consecutive predictions have the same value, it is not counted as a change) that is averagely 24 times a day or once an hour. The MP_C, MP_{C+}, MP_{C-} and MC_w values are 23.2%, 43.7%, 29.1% and 4% respectively. The total throughputs of qualified products are 11,419 tons based on humans' operations, and 12,827 tons should follow the model's predictions. It yields 12.3% throughput boosting for qualified products, which potentially has a huge economic impact on the organisation.

5. Conclusions

This paper presented an image-based solution for direct feed load estimation in grinding production in the mineral industry. Deep learning models and weakly supervised learning methods are the primary tools used. We proposed to use a two-stage training algorithm that first utilises inaccurately labelled patch images for feature extraction and then fuses extracted features for optimal feed load estimation in real-time. Through experiments, we showed a detailed demonstration of how the proposed algorithm works and provided comprehensive visualisation and discussion, showing what the model has learned and aligned with mineralogists' knowledge. We also discussed our evaluation approach and demonstrated the performance gain with the proposed work, comparing to human operators. Putting it all together, we are confident that this proposed work makes the whole grinding process much more productive and potentially brings substantial economic benefits to the mineral organisations.

We will seek methods that could generalise and transfer a learnt model to other production lines in the future. Due to the geographical distribution, the ore characteristics at different production lines vary significantly. A model trained on one production line is unlikely to scale. It requires large amounts of time for data collection and repetitive model training for such circumstances, which may become a barrier for the broader deployment of this work. We also plan to investigate the lightweight model architectures of the first model. Although the first model is used for feature extractions in this work, it outputs feed load prediction for a single patch image. The result will help provide instant feedback in production if the model can be deployed on portable devices in the environment without a network connection. It is not practical due to the current size and complexity of the model at present. Therefore, a mobile-friendly network architecture that does not compromise with too much performance.

CRediT authorship contribution statement

Li Guo: Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft. **Yonghong Peng:** Validation, Writing – review & editing, Supervision. **Rui Qin:** Software, Writing – review & editing, Visualization. **Bingyu Liu:** Resources, Data curation, Writing – review & editing, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary data

A dataset has been uploaded to Mendeley dataset service at Mendeley Data (DOI:10.17632/b9ng2y2j5p.1). Supplementary data to this article can be found online at https://doi.org/10.1016/j.eswa.2022.117469.

References

Abadi, Martin, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. *TensorFlow: A System for Large-Scale Machine Learning. Business Opp.* Vol. 10. https://www.usenix.org/system/files/conference/osdi16/osdi16-liu.pdf%5Cnhttps://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.

Arthur, David, and Sergei Vassilvitskii. 2006. "K-Means++: The Advantages of Careful Seeding," June.

Bengio, Y, Aaron Courville, Dumitru Erhan, Yoshua Bengio, and Pascal Vincent. 2009. "Visualizing Higher-Layer Features of a Deep Network Visualizing Higher-Layer Features of a Deep Network Département d'Informatique et Recherche Opérationnelle." https://www.researchgate.net/publication/265022827.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT 2010–19th International Conference on Computational Statistics, Keynote, Invited and Contributed Papers* (pp. 177–186). https://doi.org/10.1007/978-3-7908-2604-3_16

Chen, Xisong, Junyong Zhai, Qi Li, and Shumin Fei. 2007. "Fuzzy Logic Based On-Line Efficiency Optimization Control of a Ball Mill Grinding Circuit." In *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, 575–80. IEEE. https://doi.org/10.1109/FSKD.2007.329.

Cruz-Roa, A., Basavanhally, A., González, F., Gilmore, H., Feldman, M., Ganesan, S., et al. (2014). Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks. *Medical Imaging, 9041*, 904103–904104. https://doi.org/10.1117/12.2043872

Dong, Ke, and Dalin Jiang. 2014. "Automated Estimation of Ore Size Distributions Based on Machine Vision." *Lecture Notes in Electrical Engineering* 238 LNEE: 1125–31. https://doi.org/10.1007/978-1-4614-4981-2_122.

Donskoi, E., Raynlyn, T. D., & Poliakov, A. (2018). Image analysis estimation of iron ore particle segregation in epoxy blocks. *Minerals Engineering, 120*(May), 102–109. https://doi.org/10.1016/J.MINENG.2018.02.024

Donskoi, E, S P Suthers, S B Fradd, J M Young, J J Campbell, T D Raynlyn, and J M F Clout. 2007. "Utilization of Optical Image Analysis and Automatic Texture

Classification for Iron Ore Particle Characterisation." https://doi.org/10.1016/j.mineng.2006.12.005.

Engelen, Jesper E. van, and Holger H. Hoos. 2019. "A Survey on Semi-Supervised Learning." *Machine Learning 2019 109:2* 109 (2): 373–440. https://doi.org/10.1007/S10994-019-05855-6.

Gülcan, E., & Gülsoy, Ö. Y. (2017). Performance evaluation of optical sorting in mineral Processing – A case study with quartz, magnesite, hematite, lignite, copper and gold ores. *International Journal of Mineral Processing, 169*(December), 129–141. https://doi.org/10.1016/J.MINPRO.2017.11.007

Guo, L.i., Wang, H., & Zhang, J. (2019). Data-Driven Grinding Control Using Reinforcement Learning. In *Proceedings - 21st IEEE International Conference on High Performance Computing and Communications, 17th IEEE International Conference on Smart City and 5th IEEE International Conference on Data Science and Systems*. https://doi.org/10.1109/HPCC/SMARTCITY/DSS.2019.00395

Hamzeloo, E., Massinaei, M., & Mehrshad, N. (2014). Estimation of particle size distribution on an industrial conveyor belt using image analysis and neural networks. *Powder Technology, 261*(July), 185–190. https://doi.org/10.1016/J.POWTEC.2014.04.038

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. "Deep Residual Learning for Image Recognition." *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-December (December): 770–78. https://arxiv.org/abs/1512.03385v1.

Herbst, J. A., Pate, W. T., & Oblad, A. E. (1992). Model-based control of mineral processing operations. *Powder Technology, 69*(1), 21–32. https://doi.org/10.1016/0032-5910(92)85004-F

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2016). Densely connected convolutional networks. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*.

Lecun, Yann, Yoshua Bengio, and Rm 4g332. n.d. "Convolutional Networks for Images, Speech, and Time-Series." Accessed September 30, 2021.

Li, G., Klein, B., Sun, C., & Kou, J. (2020). Applying receiver-operating-characteristic (ROC) to bulk ore sorting using XRF. *Minerals Engineering, 146*, Article 106117. https://doi.org/10.1016/J.MINENG.2019.106117

Liu, Y., Zhang, Z., Liu, X., Lei, W., & Xia, X. (2021). Deep learning based mineral image classification combined with visual attention mechanism. *IEEE Access, 9*, 98091–98109. https://doi.org/10.1109/ACCESS.2021.3095368

Liu, Y., Zhang, Z., Liu, X., Wang, L., & Xia, X. (2021a). Performance evaluation of a deep learning based wet coal image classification. *Minerals Engineering, 171*, Article 107126. https://doi.org/10.1016/J.MINENG.2021.107126

Liu, Y., Zhang, Z., Liu, X., Wang, L., & Xia, X. (2021b). Deep learning-based image classification for online multi-coal and multi-class sorting. *Computers & Geosciences, 157*, Article 104922. https://doi.org/10.1016/J.CAGEO.2021.104922

Lu, X, B Kiumarsi, T Chai, FL Lewis - IET Control Theory & Applications, and undefined 2016. n.d. "Data-Driven Optimal Control of Operational Indices for a Class of Industrial Processes." *IET*.

Maitre, J., Bouchard, K., & Paul Bédard, L. (2019). Mineral grains recognition using computer vision and machine learning. *Computers & Geosciences, 130*, 84–93. https://doi.org/10.1016/J.CAGEO.2019.05.009

Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal Processing Magazine, 13*(6), 47–60. https://doi.org/10.1109/79.543975

Mukherjee, D. P., Potapovich, Y., Levner, I., & Zhang, H. (2009). Ore image segmentation by learning image and shape features. *Pattern Recognition Letters, 30*(6), 615–622. https://doi.org/10.1016/J.PATREC.2008.12.015

Mustafa, N., Zhao, J., Liu, Z., Zhang, Z., & Wenxian, Y.u. (2020). Iron ORE region segmentation using high-resolution remote sensing images based on res-U-Net. In *International Geoscience and Remote Sensing Symposium (IGARSS)*. https://doi.org/10.1109/IGARSS39084.2020.9324218

Oga, T., Harakawa, R., Minewaki, S., Umeki, Y.o., Matsuda, Y., & Iwahashi, M. (2020). River state classification combining patch-based processing and CNN. *PLOS ONE, 15*(12), Article e0243073. https://doi.org/10.1371/JOURNAL.PONE.0243073

Oquab, Maxime, Léon Bottou, Ivan Laptev, and Josef Sivic. 2015. "Is Object Localization for Free? – Weakly-Supervised Learning with Convolutional Neural Networks," June.

Perez, Claudio A, Pablo A Estévez, Pablo A Vera, Luis E Castillo, Carlos M Aravena, Daniel A Schulz, and Leonel E Medina. 2011a. "Ore Grade Estimation by Feature Selection and Voting Using Boundary Detection in Digital Image Analysis." https://doi.org/10.1016/j.minpro.2011.07.008.

Perez, C. A., Estévez, P. A., Vera, P. A., Castillo, L. E., Aravena, C. M., Schulz, D. A., et al. (2011). Ore grade estimation by feature selection and voting using boundary detection in digital image analysis. *International Journal of Mineral Processing, 101*(1–4), 28–36. https://doi.org/10.1016/J.MINPRO.2011.07.008

Poria, S., Cambria, E., & Gelbukh, A. (2015). Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing* (pp. 2539–2544).

Robben, C., Condori, P., Pinto, A., Machaca, R., & Takala, A. (2020). X-Ray-transmission based ore sorting at the san rafael tin mine. *Minerals Engineering, 145*, Article 105870. https://doi.org/10.1016/J.MINENG.2019.105870

Seff, Ari, Le Lu, Kevin M. Cherry, Holger Roth, Jiamin Liu, Shijun Wang, Joanne Hoffman, Evrim B. Turkbey, and Ronald M. Summers. 2014. "2D View Aggregation for Lymph Node Detection Using a Shallow Hierarchy of Linear Classifiers." *Medical Image Computing and Computer-Assisted Intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention* 17 (0 1): 544. /pmc/articles/PMC4350911/.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: visualising image classification models and saliency maps. *2nd International Conference on Learning Representations, ICLR 2014 - Workshop Track Proceedings, December*.

Singh, V., & Mohan Rao, S. (2005). Application of image processing and radial basis neural network techniques for ore sorting and ore classification. *Minerals Engineering, 18*(15), 1412–1420. https://doi.org/10.1016/J.MINENG.2005.03.003

Tessier, J., Duchesne, C., & Bartolacci, G. (2007). A Machine vision approach to on-line estimation of run-of-mine ore composition on conveyor belts. *Minerals Engineering, 20*(12), 1129–1144. https://doi.org/10.1016/J.MINENG.2007.04.009

Yianatos, J. B., Lisboa, M. A., & Baeza, D. R. (2002). Grinding capacity enhancement by solid concentration control of hydrocyclone underflow. *Minerals Engineering, 15*(5), 317–323. https://doi.org/10.1016/S0892-6875(02)00027-4

Yin, S., Gao, H., & Kaynak, O. (2014). Data-driven control and process monitoring for industrial applications—Part I. *IEEE Transactions on Industrial Electronics, 61*(11), 6356–6359. https://doi.org/10.1109/TIE.2014.2312885

Zhai, Xiaohua, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. 2019. "S4L: Self-Supervised Semi-Supervised Learning.".

Zhou, Z.-H. (2018). A brief introduction to weakly supervised learning. *National Science Review, 5*(1), 44–53. https://doi.org/10.1093/NSR/NWX106