Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 55-10 (2022) 1600–1605

Tacit knowledge in production sequencing: a Seq2Seq-LSTM approach

Dupuis, A. [*] **Dadouchi, C.** [*] **Agard, B.** [*]

[*] *Laboratoire en Intelligence des Données (LID)*
Département de mathématiques et génie industriel,
École Polytechnique de Montréal,
CP 6079, succursale Centre-Ville, Montréal, Québec, Canada
ambre.dupuis@polymtl.ca, camelia.dadouchi@polymtl.ca, bruno.agard@polymtl.ca

Abstract: In an increasingly complex production environment, production scheduling is more critical than ever to ensure productivity and profitability. Generally treated as an optimization problem, production scheduling faces a gap between theory and practice, but tacit knowledge on the shopfloor can influence production sequencing and scheduling. Recurrent neural networks (RNNs) are known for their ability to extract useful information from the sequential context. Seq2Seq architecture using RNNs, and specifically Long Term Memory Cells (LSTM), is known for its ability to predict discrete event sequences from sequential context. We propose a six-step methodology based on a Seq2Seq-LSTM architecture to predict the most likely scenarios used by the production manager to sequence the production, considering the actual state of the shop floor. This prediction allows the evaluation of a finite number of plausible scenarios based on past production experiences. This approach aims to bridge the gap between theoretical and practical scheduling since the historical data include the tacit knowledge present on the shopfloor.

Keywords: Industry 4.0, production scheduling and sequencing, sequence prediction, LSTM.

1. INTRODUCTION

Today's manufacturing environment is torn between cost reduction and mass customization (Womack et al., 2007). In this context, production scheduling is becoming even more critical to the profitability and productivity of companies (Cadavid et al., 2020; Guzman et al., 2021). In general, the scheduling problem is considered to be an optimization problem. Depending on the nature and complexity of the problem considered, different techniques are used. Mathematical programming can be useful to determine the optimal solution when a pure allocation problem is addressed (Baker and Treitsch, 2019). However, scheduling problems are known to be *NP-Hard*, which means that an optimal solution may not be found (Baker and Treitsch, 2019; Guzman et al., 2021). In these cases, simulation procedures, heuristic approaches and even combinatorial techniques can be used (Baker and Treitsch, 2019). Unfortunately, a gap still exists between theoretical and practical scheduling (Fuchigami and Rangel, 2018).

The goal of the present article is to propose a methodology to predict the most likely scenarios used by the production manager for sequencing production, considering the actual state on the on the shopfloor. This prediction will allow the evaluation of a finite number of plausible scenarios based on past production experiences. This approach aims to bridge the gap between theoretical and practical scheduling, since historical data takes into account tacit knowledge present on the shop floor. This approach could be used within a digital twin to simulate the effects of different predicted scenarios on production, or in reactive scheduling in order to consider the different scheduling options in case of an event disrupting the production line.

A brief overview of production scheduling can be found in section 2. The proposed methodology is detailed in section 3, and it is applied to a case study in section 4.

2. STATE OF THE ART

2.1 Scheduling and sequencing

Production scheduling is defined by Fuchigami and Rangel (2018) as both an activity and a decision-making process that aims to optimize multiple objectives by choosing the best option to perform jobs on a time basis, despite scarce resources and a complex environment.

Production scheduling can be described as a two step problem (Baker and Treitsch, 2019; Fuchigami and Rangel, 2018; Mehrsai et al., 2017). First, the sequence of tasks must be determined. Then the starting dates of those tasks is determined. Those two processes are respectively named **sequencing** and **scheduling**. Initially treated as a combinatorial optimization problem, the scheduling problem is solved by a multitude of metaheuristic algorithms, artificial intelligence (AI) methods such as neural networks or simulation (Jiang et al., 2021). The objective is generally to find the optimal or close to optimal solutions by the use of a combination of exact and non-exact methods. Several measures of performance may be used: makespan, flowtime, earliness and so on (Guzman et al., 2021; Fuchigami and Rangel, 2018). Recent studies show a great potential

[⋆] We thank IVADO, PROMPT and FRQNT for their support.

in production scheduling by using Neural Networks (NN) to solve the above mentioned optimization problem (Park et al., 2021; Ribault et al., 2021).

For Mehrsai et al. (2017), sequencing is based on inherent production parameters such as processing times or dependent setup times. As mentioned by Philipoom and Steele (2011), these parameters should not be assumed, as is generally the case in the simulation-based shop floor control literature, to be extrinsic to the context. The sequence of orders and the active role of workers have a significant influence over a shop's production or efficiency. "Often, it is the worker or supervisor who has the knowledge, developed through years of hands-on experience, which management and their information systems do not share but which must be used to schedule work efficiently" (Philipoom and Steele, 2011). Hence it is important to consider tacit knowledge in production scheduling and sequencing, as is already the case in manufacturing supply chain management (Al-Mutawah et al., 2009) or in the analysis of the innovation culture of companies (Nonaka and Takeuchi, 1995). Therefore, decision makers should exploit data collected from past production logs as it contains workers' expertise as well as information about the production context. However, this is not yet the case, since, as Fuchigami and Rangel (2018) highlighted, there is a lack of studies using actual data collected by manufacturing companies' information systems. This may partly explain the gap between scheduling theory and practice.

2.2 Recurrent neural networks

Rafiq et al. (2001) define NN as "artificial intelligence tools capable of learning and generalizing from examples and experiments to find meaningful solutions to problems". There are different types of NN such as artificial neural networks (ANN), convolution neural networks (CNN) and recurrent neural networks (RNN).

For (Bengio et al., 1994), RNNs are a specific type of NN able to handle sequences while taking into account contextual information in a flexible way. They are widely used to process sequential data as well as time series in order to predict and classify context-related events in different contexts (Fei and Tan, 2018). However, RNNs suffered from learning difficulties due to known phenomena such as vanishing and exploding gradients (Bengio et al., 1994). Hochreiter and Schmidhuber (1997) proposed Long-Short Term Memory (LSTM) as an RNN that is able to deal with such phenomena with its internal architecture. LSTM is now one of the most widely used RNNs.

The RNN auto-encoder architecture is used to answer sequence to sequence (Seq2Seq) prediction problems (Chujie et al., 2019). It is based on the coupling of an encoder model consisting of a layer of n_r RNN cells and a decoder model also consisting of n_r RNN cells associated with a dense layer. Known sequences are fed by an inputs layer to the encoder model. The encoder model outputs a vector context representative of the context proposed by the input sequences. This context vector is then fed to the decoder model as an input to predict a plausable future sequence based on the proposed context. To help train the model, the *Teach forcing* method can be used (Yaser et al., 2020). The principle is to correct the model at each step of the prediction in order to limit the propagation of errors in the predicted sequence. A set of DTI is generated to compare the predictions of the model with the real observed values, and the correction is made by comparing the predicted value at time t (s_t) with the value actually observed at time t (DTI_{t+1}).

To reduce the gap between scheduling theory and practice, the tacit knowledge of the production floor should be considered to propose plausible solutions. Considering that LSTMs are known to extract useful information from sequential data and that the Seq2Seq architecture predicts plausible sequences of events based on the sequential context, we propose a six-step methodology that is based on the analysis of production histories. It uses the Seq2Seq-LSTM algorithm to determine the most likely scenarios chosen by experienced workers in a specific production context. These scenarios could then be used in a simulation model to evaluate and select the most effective of the plausible scenarios based on business priorities.

3. METHODOLOGY

Figure 1 depicts the methodology proposed to predict plausible sequencing choices in a given production context.

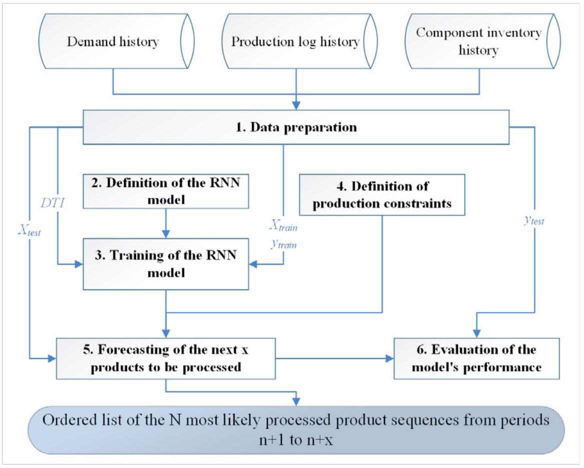

Demand history
Production log history
Component inventory history
1. Data preparation
2. Definition of the RNN model
4. Definition of production constraints
DTI
X_{test}
X_{train}
y_{train}
y_{test}
3. Training of the RNN model
5. Forecasting of the next x products to be processed
6. Evaluation of the model's performance
Ordered list of the N most likely processed product sequences from periods n+1 to n+x

Fig. 1. Proposed methodology

(1) Data preparation aims to format the data according to the requirements of the RNN prediction model. Since the demand, the production log and the component inventory are information that influence the choice of production sequencing, they will be used as contextual information for the model. The importance of each piece of information on the production sequences can be evaluated by varying the information in the prediction model and observing the effects of this variation on the performance of the model **(step 6)**. Data preparation includes the management of missing data and the modeling of sequences using two hyperparameters n_{in} (size of the context sequence) and n_{out} (size of the sequence to be predicted). The sequences are then transformed into encoded and normalized matrices. Finally, the data is divided into a training set and a test set for the next steps of the methodology. The labels of the training set are used to generate the DTI dataset used in the teach forcing training.

(2) Definition of the RNN model. A Seq2Seq architecture using single layer LSTM cells and a teach forcing training method is chosen. As explained in section 2.2, this kind of architecture assumes the definition of a minimum of 5 layers, namely: two input layers (one for the input data and one for the teach-forcing input), an LSTM layer for the encoder and an LSTM layer as well as a dense layer for the decoder. As the production sequencing problem uses three separate input data (demand history, production log history and component inventory history), the architecture of the model is defined in Table 1.

Table 1. Architecture's layers and parameters

Layer	Used for	Parameter	Value
Input	Demand	# Steps for prediction	n_{in}
		# Elements	$\|P\|$
	Production log	# Steps for prediction	n_{in}
		# Elements	$\|P\|+1$
	Component	# Steps for prediction	n_{in}
		# Elements	$\|W\|$
	Teach forcing	# Steps to be predicted	n_{out}
		# Elements	$\|P\|+1$
LSTM	Encoder	# Neurons	n_r
	Decoder	# Neurons	n_r
Dense	Decoder	# Elements	$\|P\|+1$

With # standing for "Number of"

$\|W\|$ is the number of unique components present in the component inventory dataset. $\|P\|$ is the number of unique products manufactured by the company (product catalog). n_{in}, n_{out} and n_r are hyperparameters chosen arbitrarily for the NN. Figure 2 represents the NN layout. A third dimension is added to the input layers to represent the number of examples present in each data set. This dimension can vary, so it is set to the "None" value.

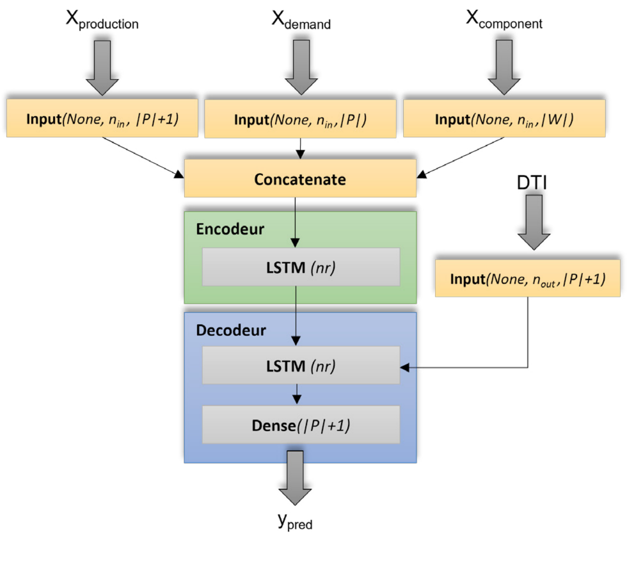


Fig. 2. Proposed architecture

With the model architecture defined and the data prepared, **(3) Training of the RNN model** can be performed. This supposes the definition of the loss function (compared to the predict labels y_{pred} to the real labels y_{train}) as well as the metric to measure the performance of the model. It also assumes the definition of an optimizer, allowing the loss function that was previously defined to be minimized. Then, the model is trained using the training data set. The DTI dataset is used as an input to the LSTM (decoder) layer (Figure 2). At each time step *t*, the prediction is based on the actual value that should have been predicted at the previous time step *t-1*. This method, known as teach forcing, allows for the learning of the model to be faster and more accurate, limiting the propagation of errors in time. The output of the trained model is, for each record, a matrix of size ($n_{out} \times \|P\|+1$) constituted of the probability vectors of occurrence of each product P_i at each time step in $\|n_{out}\|$. Then, a BeamSearch algorithm is used to determine the N most probable $\|n_{out}\|$ sequences of products (Top-N), with a BeamScore to quantify the prediction. Finally, production constraints may be implemented.

(4) Definition of production constraints consists of a set of constraints defined by decision makers to manage occasional or recent changes that may have an impact on production. They can be formulated in different ways, such as linear programming (equation 1) or association rules. For example, a transition between products X and Y can be penalised or encouraged using, respectively, a positive or a negative rate Z, such as:

$$If\ X \rightarrow Y\ Then\ penality\ = Z\%\ of\ BeamScore \quad (1)$$

This set of constraints is easy to modify and allows greater flexibility to incorporate changes in policy and production strategies. These rules will be combined with the predicted scenarios proposed by the RNN models **(5)**.

(5) Forecast of the next x products to be processed: constraints defined in **(4)** are applied to each of the N proposed scenarios obtained using the trained model from step **(3)** on the test set. A "compliance score" is computed for each scenario according to the certainty of the RNN model (BeamScore) and the adequacy of the considered scenario with the production constraints. The scenarios are then ranked by their "compliance score" leading to an ordered list called **TOP-N** of the N most probable product sequences for the period n+1 to n+x.

(6) Evaluation of the model performance. Top-N metrics often used in recommender systems (Ren et al., 2013) are used to evaluate the model. It evaluates "the percentage of known relevant items in the test set y_{test} that appear in the first N predicted items y_{pred}" (Herlocker et al., 2004). The percentage of information contained in an item in the list of the first N items can also be measured. Cremonesi et al. (2010) calls these performance measures Recall(N) and Precision(N) (see equations 2).

$$\begin{aligned} Recall(N) &= \frac{\sum_{k=1}^{|y_{test}|} TP_k}{|y_{test}|} \\ Precision(N) &= \frac{Recall(N)}{N} \end{aligned} \quad (2)$$

$$\begin{aligned} &With: \\ &|y_{test}| = number\ of\ records\ in\ y_{test} \\ &TP_k = \begin{cases} 1\ If\ y_{test_k} \in Top-N_k \\ 0\ Else \end{cases} \\ &N = Number\ of\ considered\ scenarios \end{aligned}$$

4. CASE STUDY

A simplified production batches sequencing problem is used to exemplify the proposed methodology. Consider a workshop made of one machine $\boldsymbol{M}$and whose product catalog $\boldsymbol{P}$ contains 4 items such as $\boldsymbol{P}$=$\{A,B,C,D\}$. These products are themselves derived from 4 components $\boldsymbol{W}$=$\{s,t,u,v\}$. It should be noted that the machine $\boldsymbol{M}$ can only produce one type of product P_i at a time.

The production manager is responsible for the sequencing of the jobs to be performed on machine $\boldsymbol{M}$. As shown in the example below, the machine's production log records the type and quantity of products processed each day and the sequence in which the processing is done.

Production history of the machine $\boldsymbol{M}$:

- **Day 1** : C=3 ; A=4 ; D=6 ; B=3 ; A=10 ; STOP ; C=1
- **Day 2** : D=8 ; B=5 ; STOP ; C=2 ; C=3 ; A=3 ; D=2
- **...**

The objective of the study is to forecast the future production sequences most likely to de proposed by the production manager. These scenarios will then be used as inputs to a simulation model allowing the calculation of the different performance indicators that will help in the final scheduling decision. Using past experience and simulation to determine the best sequencing scenario based on different key performance indicators allows for the consideration of habits and non-explicit rules that govern the company's production. This will help bridge the gap between sequencing theory and practice.

4.1 Data preparation

To fulfill the above objective, the production history, the "net" demand history and the component inventory history are available. Tables 2, 3 and 4 represent this information for the example presented above. Each period refers to a production order.

Table 2. Production log

Period / Day	1	2	3	4	5	6	7
1	C = 3	A = 4	D = 6	B = 3	A = 10		C = 1
2	D = 8	B = 5		C = 2	C = 3	A = 3	D = 2
...	...						
n	A = 5	D = 8	B = 2	C = 3	D = 5	D = 7	

Table 3. Demand history

Periode	**1**				**...**	**7**			
P_i / **Day**	A	B	C	D	...	A	B	C	D
1	6	2	3	8	...	2	2	2	8
2	3	1	1	9	...	4	2	2	6
...	...								
n	1	2	2	6	...	3	1	3	7

Table 4. Component inventory history

Periode	**1**				**...**	**7**			
W_i / **Day**	s	t	u	v	...	s	t	u	v
1	3	2	9	3	...	7	3	8	0
2	6	3	7	11	...	7	3	5	16
...	...								
n	6	2	7	1	...	10	1	9	10

Table 2 shows that on day 1, machine M did the following production: 3 parts C in the first period, then 4 parts A, 6 parts D, 3 parts B, 10 parts A. We do not have information for period 6 because the data is missing (for any reason), then in period 7, 1 part C has been produced. The same applies for the following days. Table 3 illustrates the demand over 7 periods, in period 1, it required 6 parts A, 2 parts B, 3 parts C, 8 parts D, and so on for the following days and periods. In a similar way, Table 4 shows the component inventory history.

Missing data in production sequences are encoded as "XX" and the produced quantity is set to 1. In the other datasets, missing values are set to 0. Subsequently, the production sequences are modeled using a moving window over all the periods considered. In this case study, hyperparameters are set to $n_{in} = 3$ and $n_{out} = 2$ (Table 5).

Table 5. Modeling the sequences

dataset	$\chi_{prod} = n_{in} = 3$			$\gamma_{prod} = n_{out} = 2$	
	-2	**-1**	**0**	**1**	**2**
1-1	C=3	A=4	D=6	B=3	A=10
1-2	A=4	D=6	B=3	A=10	XX
1-3	D=6	B=3	A=10	XX	C=1
...	...				
n-3	B=2	C=3	D=5	D=7	XX

The first n_{in} columns represent the context data used for the prediction while the last n_{out} columns are the data to be predicted. In the context of the prediction of production sequence, only information related to the sequences of products to be processed must be predicted. Thus, four distinct data sets must be determined:

- $\chi_{prod} = n_{in}$ first columns of the production data set extracted by the moving window (columns -2 to 0 in table 5)
- χ_{stock} = inventory level of the component associated with the temporality of χ_{prod} data
- χ_{demand} = level of demand for the products associated with the temporality of the χ_{prod} data
- $\gamma_{prod} = n_{out}$ last columns of the production data set extracted by the moving window (two last columns in table 5)

The categorical data, χ_{prod} and γ_{prod}, are subsequently encoded as binary matrices X_{prod} and y_{prod} of size (# records $\times\ n_{in} \times \|P\|+1$) and (# records $\times\ n_{out} \times \|P\|+1$) respectively. The $|P|+1$ columns are taken into account in order to allocate a specific column to each product in the product catalog, as well as to the 'XX' value previously defined. The X_{prod} matrix is weighted by the quantity produced in each period. $X_{prod_{11}}$ and $y_{prod_{11}}$ (equation 3) is the matrix representation of the first n_{in} columns and the last n_{out} columns of record 1-1 in Table 5.

$$\chi_{prod_{11}} = \begin{array}{|c|c|c|}\hline \textbf{-2} & \textbf{-1} & \textbf{0} \\ \hline \text{C=3} & \text{A=4} & \text{D=6} \\ \hline \end{array} \rightarrow X_{prod_{11}} = \begin{bmatrix}\begin{bmatrix}0&0&3&0&0\end{bmatrix}\\ \begin{bmatrix}4&0&0&0&0\end{bmatrix}\\ \begin{bmatrix}0&0&0&6&0\end{bmatrix}\end{bmatrix} \tag{3}$$

$$\gamma_{prod_{11}} = \begin{array}{|c|c|}\hline \textbf{1} & \textbf{2} \\ \hline \text{B=3} & \text{A=10} \\ \hline \end{array} \rightarrow y_{prod_{11}} = \begin{bmatrix}\begin{bmatrix}0&1&0&0&0\end{bmatrix}\\ \begin{bmatrix}1&0&0&0&0\end{bmatrix}\end{bmatrix}$$

A fifth dataset is needed to train the model according to the teach forcing method. The DTI set is generated from the y_{prod} matrix by deleting its last vector and adding a null vector in the first position, as shown in equation 4.

$$DTI_{11} = \begin{bmatrix}\begin{bmatrix}0&0&0&0&0\end{bmatrix}\\ \begin{bmatrix}0&1&0&0&0\end{bmatrix}\end{bmatrix} \tag{4}$$

The numerical datasets χ_{stock} and χ_{demand} are transformed into matrices X_{stock} and X_{demand} of respective size $(\#records \times n_{in} \times \|P\|)$ and $(\#records \times n_{in} \times \|W\|)$. The datasets are normalized by min/max normalization to facilitate model convergence during training (equation 5).

$$X_{demand_{11}} = \begin{bmatrix} [6 \ 2 \ 3 \ 8] \\ [8 \ 4 \ 2 \ 10] \\ [6 \ 4 \ 1 \ 3] \end{bmatrix} \rightarrow \begin{bmatrix} [0.54 \ 0.50 \ 1.00 \ 0.55] \\ [0.72 \ 1.00 \ 0.50 \ 0.77] \\ [0.54 \ 1.00 \ 0.00 \ 0.00] \end{bmatrix} \tag{5}$$

$$X_{stock_{11}} = \begin{bmatrix} [3 \ 2 \ 9 \ 3] \\ [9 \ 2 \ 7 \ 6] \\ [4 \ 3 \ 9 \ 5] \end{bmatrix} \rightarrow \begin{bmatrix} [0.09 \ 0.50 \ 0.57 \ 0.27] \\ [0.63 \ 0.50 \ 0.42 \ 0.54] \\ [0.18 \ 1.00 \ 0.57 \ 0.45] \end{bmatrix}$$

Finally, the data sets are separated into training and test sets (75%-25%).

4.2 Definition of the RNN model

As mentioned, $\|P\|$ and $\|W\|$ can be inferred from the data, while hyperparameters n_{in}, n_{out} and n_r are fixed arbitrarily. In the present case study, $\|P\| = 4$, $\|W\| = 4$, $n_{in} = 3$, $n_{out} = 2$ and $n_r = 282$. Figure 3 represents the architecture of the model used for this case study.

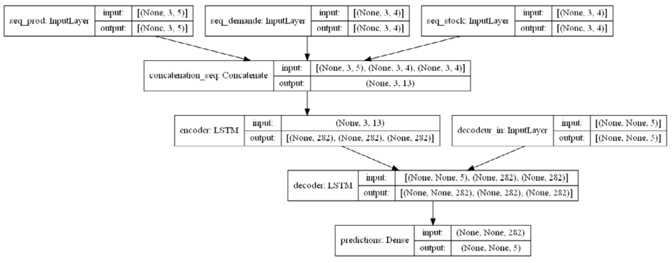


Fig. 3. Representation of the proposed architecture

4.3 Training of the RNN model

$X_{prod_{train}}$, $X_{demand_{train}}$, $X_{stock_{train}}$, and DTI are used as inputs while $y_{prod_{train}}$ is used as output to train the model. ***Python*** is used with the optimizer $ADAM$ to minimize the loss function *Categorical_Cross_Entropy*. This loss function is chosen since the real value $y_{prod_{train}}$ towards which the prediction y_{pred} must tend is a binary matrix. This also explains why the *Categorical_accuracy* metric is chosen to evaluate the performance of the model during training. **Keras** *EarlyStopping* function avoids any overfitting, with 25% of the training set used as a validation set. After more than 50 iterations without improvement of the accuracy on the validation set, training is stopped.

4.4 Definition of production constraints

A constraint is used to penalize or encourage a transition. For instance, the transition from product C to product D can be penalized using equation 6. This can be seen as the result of a shortage of raw materials involved in this specific series change. Hence the need expressed by decision makers to penalize this transition without completely prohibiting it using a 70% penalty rate.

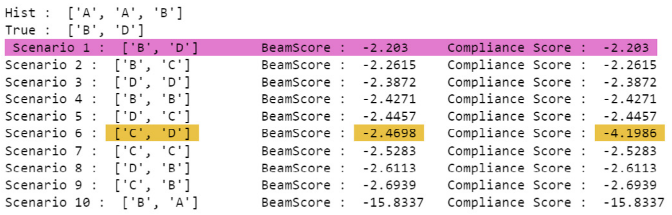
$$If\ C \rightarrow D\ Then\ penalty\ = +70\%\ of\ BeamScore \tag{6}$$

The rate, within an interval [-100%;+100%], is chosen according to the flexibility given to the constraint. The higher the rate chosen, the stricter the constraint. A negative rate encourages the specific behavior defined in the constraint while a positive rate penalizes it.

4.5 Forecasting of the next x products to be processed

The test sets $X_{prod_{test}}$, $X_{demand_{test}}$, $X_{stock_{test}}$ are used as an input to the model. The DTI set used during the training of the model is replaced by a null vector since no information about $y_{prod_{test}}$ is available during the test phase. Using the BeamSearch algorithm, N scenarios are selected as the most probable to occur in the following x steps. Figure 6 is an example of a prediction proposed by the RNN model with N=10.

Table 6. Example of RNN model's prediction



```
Hist :  ['A', 'A', 'B']
True :  ['B', 'D']
 Scenario 1 :  ['B', 'D']          BeamScore :  -2.203     Compliance Score :  -2.203
Scenario 2 :  ['B', 'C']           BeamScore :  -2.2615    Compliance Score :  -2.2615
Scenario 3 :  ['D', 'D']           BeamScore :  -2.3872    Compliance Score :  -2.3872
Scenario 4 :  ['B', 'B']           BeamScore :  -2.4271    Compliance Score :  -2.4271
Scenario 5 :  ['D', 'C']           BeamScore :  -2.4457    Compliance Score :  -2.4457
Scenario 6 :  ['C', 'D']           BeamScore :  -2.4698    Compliance Score :  -4.1986
Scenario 7 :  ['C', 'C']           BeamScore :  -2.5283    Compliance Score :  -2.5283
Scenario 8 :  ['D', 'B']           BeamScore :  -2.6113    Compliance Score :  -2.6113
Scenario 9 :  ['C', 'B']           BeamScore :  -2.6939    Compliance Score :  -2.6939
Scenario 10 :  ['B', 'A']          BeamScore :  -15.8337   Compliance Score :  -15.8337
```

The sequence ['A', 'A', 'B'] is introduced in the model along with the demand and the stock level of the components of the period. The model proposes 10 probable scenarios in decreasing order of conviction (*BeamScore*). The production constraint imposes a 70% penalty on each scenario containing a transition from C to D (scenario 6 is penalized). The compliance score quantifies the certainty of the model. The lower the compliance score and the larger the difference between two consecutive scenario scores, the more certain the prediction.

4.6 Evaluation of the model's performance

Performance of the model can be evaluated using the metrics defined in equations 2. The performance of the model is highly dependent on the quality of the data of the case study. Each prediction scenario is compared to the actual sequence used on the machine, i.e. ['B', 'D'] in this example (table 6). The first scenario (in pink) proposed by the model is the same as the real sequence (True) used on the machine. It will therefore be recorded as a good Top 1 prediction. If the True sequence is predicted in scenario 2 or 3, it will be recorded as a good Top 3 prediction. The same logic applies to Top 5 and Top 10 predictions. Each Top N prediction is divided by the number of predicted records to obtain the corresponding Recall(N).

5. CONCLUSION

The proposed methodology suggested N probable sequences of production scenarios. Considering the tacit knowledge of the shop floor included in historical data, it aims to bridge the gap between theoretical and practical scheduling, and can be considered as a tool to use in production rescheduling (reactive scheduling) (Sun and Xue, 2001; Mihoubi et al., 2021).

To further reduce this gap, the following research perspectives could be explored. The visualisation of the scenarios could allow for better understanding of available options and facilitated discussion between decision makers and workers. This could be accomplished using a Gantt chart, as a visual support for further reflection and decision making (Wilson, 2003) and several key performance indicators could be calculated.

Additionally, companies are driven to make multi-criteria decisions based on the simultaneous analysis of several key indicators (Fuchigami and Rangel, 2018). Since N plausible scenarios are proposed, and considering that processes are known by the company, the effect of each proposed scenario could be simulatedfor example using a digital twin, in order to provide insightful input for a multi-criteria decision process. Then, the AHP decision-making method could consider preferences of decision makers (Charles and Kumar, 2014) and allow more flexibility in the decision making process.

Finaly, further research should be undertaken to determine if the practices learned, both good and bad, can be used to quickly create a responsive optimization scheme for the studied context.

REFERENCES

Al-Mutawah, K., Lee, V., and Cheung, Y. (2009). A new multi-agent system framework for tacit knowledge management in manufacturing supply chains. *Journal of Intelligent Manufacturing*, 20, 593–610.

Baker, K. and Treitsch, D. (2019). *Principles of sequencing and scheduling*. John Wiley & Sons, second edition.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2), 156–166.

Cadavid, J., Lamouri, S., Grabot, B., Pellerin, R., and Fortin, A. (2020). Machine learning applied in production planning and control : a state-of-the-art in the area of industry 4.0. *Journal of Intelligent Manufacturing*, 31, 1531–1558.

Charles, V. and Kumar, M. (2014). *Business Performance Measurement and Management*, chapter 22, 566–567. Cambridge Scholars Publisher, 12 Back Chapman Street, Newcastle upon Tyne, NE6 2XX, UK.

Chujie, W., Lin, M., Rongpeng, L., Tariq, D., and Honggang, Z. (2019). Exploring trajectory prediction through machine learning methods. *IEEE Access*, 7, 101441–101452.

Cremonesi, P., Koren, Y., and Turrin, R. (2010). Performance of recommender algorithms on top-n recommendation tasks. *RecSys2010*, 39–46.

Fei, H. and Tan, F. (2018). Bidirectional grid long short-term memory (bigridlstm): A method to address context-sensitivity and vanishing gradient. *Algorithms*, 11(11), 172.

Fuchigami, H. and Rangel, S. (2018). A survey of case studies in production scheduling : analysis and perspectives. *Journal of Computational Science*, 25, 425–436.

Guzman, E., Andres, B., and poler, R. (2021). Models and algorithms for production planning, scheduling and sequencing problems: A holistic framework and a systematic review. *Journal of Industrial Information Integration*, 100287.

Herlocker, J., Konstan, J., Terveen, L., and Riedl, J. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1), 5–53.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

Jiang, Z., Yan, S., Ma, J., and Wang, Q. (2021). The evolution of production scheduling from industry 3.0 through industry 4.0. *International journal of production research.* doi: https://doi.org/10.1080/00207543.2021.1925772.

Mehrsai, A., Figueira, G., Santos, N., Amorim, P., and Almada-Lobo, B. (2017). Decentralized vs. centralized sequencing in complex job-shop scheduling. *International Federation for Information Processing*, 513, 467–474.

Mihoubi, B., Bouzouia, B., and Gaham, M. (2021). Reactive scheduling approach for solving a realistic flexible job shop scheduling problem. *International journal of production research*, 59(19), 5790–5808.

Nonaka, I. and Takeuchi, H. (1995). *The Knowledge-creating Company: How Japanese Companies Create the Dynamics of Innovation.* Oxford University Press, New York.

Park, J., Chun, J., Kim, S., Kim, Y., and Park, J. (2021). Learning to schedule job-shop problems: Representation and policy learning using graph neural network and reinforcement learning. *International journal of production research*, 59(11), 3360–3377.

Philipoom, P. and Steele, D. (2011). Shop floor control when tacit worker knowledge is important. *Decision Sciences*, 43(3), 655–688.

Rafiq, M., Bugmann, G., and Easterbrook, D. (2001). Neural network design for engineering applications. *Computers & Structures*, 79(17), 1541–1552.

Ren, Y., Li, G., and Zhou, W. (2013). A learning method for top-n recommendations with incomplete data. *Social Network Analysis and Mining*, 33, 1135–1148.

Ribault, A., Vercraene, S., Henry, S., and Ouzrout, Y. (2021). Economic optimisation of cold production: a matheuristic with artificial neural network approach. *International journal of production research*, 59(22), 6941–6962.

Sun, J. and Xue, D. (2001). A dynamic reactive scheduling mechanism for responding to changes of production orders and manufacturing resources. *Computers in industry*, 46(2), 189–207.

Wilson, J. (2003). Gantt charts: A centenary appreciation. *European Journal of Operational Research*, 149, 430–437.

Womack, J., Jones, D., and D.Roos (2007). *The machine that changed the world.* Simon & Schuster, New York.

Yaser, K., Tian, S., Naren, R., and Chandan, R. (2020). Deep reinforcement learning for sequence-to-sequence models. *IEEE Transactions on Neural Networks and Learning Systems*, 31(7), 2469–2489.