Hybrid Static-Sensory Data Modeling for Prediction Tasks in Basic Oxygen Furnace Process

Davi Alberto Sala[1*], Andy Van Yperen-De Deyne[2], Erik Mannens[1] and Azarakhsh Jalalvand[1]

[1*]IDLab, Ghent University–imec, Ghent, Belgium.
[2] ArcelorMittal, Ghent, Belgium.

*Corresponding author(s). E-mail(s): davialberto.sala@ugent.be;

Abstract

In this paper, we propose a novel data-driven prediction system for Multivariate Time Series (MTS) in an industrial context, where classic relational data contain key information in order to properly interpret the MTS. Particularly we focus on the accurate endpoint prediction of temperature and chemical composition at the basic oxygen furnace, which is a step in the steel production pipeline where liquid iron is refined to steel. The precise prediction of temperature is important for proper process control while reaching the target chemical composition is essential for quality control. Our deep learning methodology employs two modules followed by an aggregation block; a Convolutional Neural Network (CNN) handles the MTS, while in parallel, the static data is processed by a Fully Connected Network (FCN). We enhance the CNN performance by adding two Squeeze-and-excitation (SE) blocks, which act like an attention module over the different channels. By taking the MTS data into account we improve the prediction by up to **10%** relative over the models which only consider the static data. The hybrid FCN-CNN-SE architecture slightly improves the state-of-the-art MTS approaches by **2%**, with less outliers on the prediction of final temperature and phosphorus concentration, while being easier to implement and more scalable to larger datasets and input space than current solutions.

Keywords: basic oxygen furnace, data-driven prediction models, multivariate time series, neural networks, steel production

1 Introduction

In basic industry, such as steel production factories, the whole process pipeline is heavily instrumented and constantly monitored by means of numerous sensors. In the past, the main purpose of sensors at a production installation has been to provide process control, monitoring and automation. Nowadays and as a result of the recent revolution in big data warehousing, logging of this sensory data is becoming a norm and time series are omnipresent in automated industrial processes. This opens an astonishing opportunity for data analysis and enables the development of data-driven models that are usually faster than physical models and can augment classic approaches, with the aim of improving performance.

The Basic Oxygen Furnace (BOF) process, the focus of this work, is an essential step in steel production pipeline in which the hot metal is refined to liquid steel. We consider this process as a use case to demonstrate the feasibility of advanced analytics in the optimization of highly

instrumented industrial processes, based on large amounts of historical data.

These processes are traditionally modelled based upon the so-called "static data", which are usually measured or calculated prior blow. Static data are single variables that remain constant during the production of a batch, such as amount of iron ore used, types of recycled scrap, coolants and other additions.

In this paper, the aim is to enhance the above approach by including so-called *dynamic data* which is frequently captured during the BOF process by potentially hundreds of sensors to be used in control systems and steer the process. These Multivariate Time Series (MTS) are expected to contain real-time information that is not grasped within the *static data*, and potentially enhance the modelling of the process.

In the BOF process, conventional control systems are designed to automatically predict endpoints of the crucial variables (such as temperature, phosphorus and carbon concentrations) which will determine the final quality of the steel.

Although these models play an important role in controlling the BOF process, developing them is usually a big challenge due to the complexity of the process.

Prediction takes place in two phases: (1) prior blow to set up the parameters of the process based on approximately 50 selected static values, and (2) at about 80% progress of the blow phase to verify or rectify the initial setup based on a fraction of the dynamic data collected during the process so far.

Despite the quantity and diversity of the available data, most of the current prediction systems still rely only on a few selected static and very small portion of the logged data. Model-driven techniques rely on a deep understanding of the system and usually benefit from a scientifically established knowledge. However, such models cannot accommodate large complexities, hence, must be simplified and usually do not handle the noisy data and the possible influence of unincluded variables is not accounted for. On the other hand, data-driven models based on machine learning are data hungry, requiring large amounts of historical data to produce meaningful results.

The objective of our work is to improve the performance of the second prediction phase using fully data-driven models and by taking into account a larger portion of the available time series data.

Several time series regression and classification algorithms have been developed throughout the years. Classical regression approaches in time series data analysis usually include autoregressive models to predict the next few values, based on the history of the same time series. Prediction of prices in the stock market is a common example in this field. Typical approaches are autoregressive integrated moving average (ARIMA) models and lately more advanced techniques such as neural networks [1] are used. However taking into account correlations among the multivariate time series is much less common, even though multivariate extensions to these classic techniques do exist [2]. Distance based metrics, such as Dynamic Time Warping, alongside K-nearest neighbors have successfully been used for classification tasks based on multivariate time series [3]. Other possible approaches are traditional feature extraction algorithms allied with a classification or regression models [4]. Little research [5] was found for single value regression that combines inputs from multivariate time series and static data on large datasets.

As far as the more modern data-driven models concern, there is evidence that Deep Learning is a competitor to the conventional multivariate time series classification [6]. The main advantage of Deep Learning approaches is that instead of heavy feature engineering to extract the information from the data, only a small preprocessing step is needed and the model is expected to extract and learn the most informative features automatically. Consequently, the model is less biased towards the domain-expert's prior knowledge and investigating the extracted features could even provide a new insight to the domain knowledge. On the other hand, such techniques usually require much larger labeled training data to learn the features effectively. It is possible to extend these set of techniques for regression tasks and by adding further complexity, such models can handle static data along with the MTS.

With the continuing growth of computing power, graph computations on GPUs have enabled the deployment of larger and more complex models. A very recent research [2] presents the potential of deep learning architectures based on Feed

Forward Neural Networks, Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) in particular, for multivariate time series processing. While RNNs have shown great performance in the past years, specially brain-inspired methodologies such as [7], CNNs have shown higher model explainability in recent studies [8]. Identification of which regions and signals of the input data that are important for predictions is highly desirable in industrial applications, as it might lead to further process insights.

In this paper, we will extend the Neural Network approach on multivariate time series, with the inclusion of static data. With this approach, we avoid the time series feature extraction and engineering downsides, and allow to take into account valuable global information from the sensors. In particular, we propose a Deep Learning framework based on Fully Connected Networks (FCN) and CNN suitable for regression tasks on real-world, industrial data, containing imbalanced, noisy samples that takes into account both static and multivariate time series information.

The rest of this paper is organized as follows: Section 2 provides a brief overview of the BOF process. In Section 3, a literature study on multivariate time series is provided, covering both industrial and academic oriented research. The proposed methodology and model architecture are presented in Sections 4 and 5, respectively, followed by results and discussions in Section 6.

2 Basic Oxygen Furnace

The first step for steel production starts at the blast furnace, where iron ore (mainly composed of iron and oxygen) is melted in a reducing atmosphere by lowering its oxygen content. The exothermic reaction provides the necessary heat for melting the reduced ores. The liquid metal is then transported to the steel plant, where the pipeline begins with the Basic Oxygen Furnace (BOF), the main focus of this study.

During the BOF process, scrap and hot metal are charged into the converter vessel and pure oxygen is blown on the metal bath by means of a water-cooled lance. A diagram depicting this process is shown in Fig. 1. Burned lime and other additives are added during blowing to regulate the process and achieve targeted composition. The blowing phase takes 16 minutes on average. Meanwhile, an inert gas is injected from the bottom of the BOF vessel to maintain the mixture homogeneous. The blow partly oxidizes the carbon, silicon, manganese, phosphorus and iron in the bath. These transformations liberate a huge amount of heat, which melts the scrap and raises the bath temperature. The impure elements are converted into gas or slag, the latter floating on the top of the liquid bath. By the end of the blow, the liquid steel has reached a temperature of approximately 1650^{o}C. After the blowing phase, the vessel is tilted, steel is tapped into a steel ladle and the slag is tapped into a slag pot. The converter is ready for the next batch, while the liquid steel is further alloyed and casted into slabs in continuous casting to the slabyard, ready to be hot rolled.

The target of this case study is to have an improved performance of models estimating the chemical composition and the temperature, as they directly influence the metallurgical properties of the produced steel. Even though the steel is alloyed in a later stage, a good performance in reaching chemical composition at the BOF reduces the necessity of adding expensive alloys.

Chemical composition has a direct impact on the steel properties, such as tensile strength, formability, toughness and weldability, which are set by the customer to meet the requirements for the specific application of the end product. The end-of-blow temperature of the liquid steel is essential, especially for planning purposes. Too high temperature requires additional coolants or cool down time which disturbs the (energetically) optimal planning. Too low temperature on the other hand, is even more problematic as the solidification will occur before the planned casting. This requires interruption of the tapping procedure and reheating the converter vessel, with huge impact on hot metal logistics.

3 Related work

In this section, we will first review the current state of the art on data-driven prediction tasks in the industry, especially steel production and BOF applications, where most works focus on static features and analytical models. We then provide a brief overview of the current progress in multivariate time series regression and classification solutions, thus creating the link required to

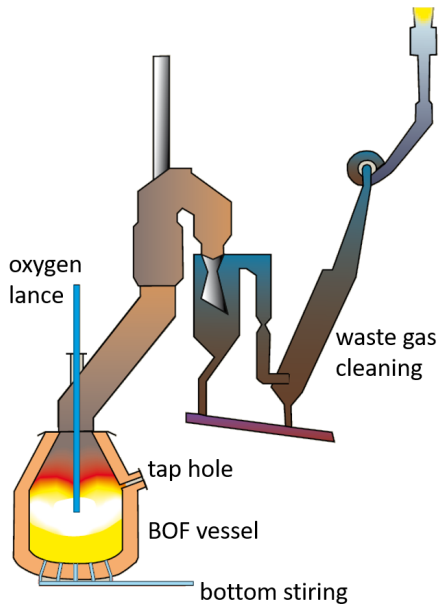


Fig. 1: The Basic Oxygen Furnace process where high speed oxygen is blown into a metal bath to reduce its carbon concentration.

elaborate our proposed approach, in which time-series and static data are combined to improve the prediction.

3.1 On the Steel Industry

In [9], a hybrid model based on Fuzzy C-Means clustering (FCM) and Support Vector Machine (SVM) is trained to predict the endpoint temperature and carbon content of the BOF steel making process. It was shown that the proposed method performs better than the conventional solutions such as multi-layer perceptron (MLP), classical SVM and Case-based reasoning (CBR). The experiments are conducted on a small and closed dataset that contains only static information from each batch, such as, amount of scrap and iron, carbon content and temperature after the first blow, coolant. More recently, a similar solution based on support vectors was presented in [10].

A multivariate solution is presented in [11], where a search algorithm is used to select relevant static features (from a total of 52 features) for predicting the endpoint concentrations of Fe, Mn, P and S within the BOF process for evaluating different recycled steel mixtures. The authors trained a distinct model for each of the seven possible steel mixes and endpoint target. The prediction performance of the models are reasonably good but a high variance is observed, which the authors explained by the high uncertainty in the input variables on the different mixes of recycled steel.

Also, a model for predicting the endpoint phosphorus content on the BOF process is presented in [12]. The study employs 21 hand-selected static input variables, which are used to cluster the training data using k-Means (k is empirically determined). Each cluster is then considered to develop a Polynomial Neural Network regressor. Their experiments show that clustering the data can have a positive impact on the results compared to a single regressor model, improvements were approximately 2% better when using clustering.

Different classical machine learning techniques such as SVM, Artificial Neural Networks (ANN), Regression Trees and Fuzzy inference are used in [13] to predict the yield in the steel-making process using 10 static relevant variables as input. The authors report SVMs as the best predictor. A different study [14] employs multivariate polynomial regression to predict the endpoint content of Fe, Mn, P and S. The authors trained the proposed model on a dataset collected over three years. Having used 56 static variables as the input to the model, the reported results show Fe as the most difficult target to predict, while Mn, P and S have a high correlation but with a significant variance.

Novel machine learning methodologies are usually applied on industrial use-cases only after being well established in academic research. Time-series have several applications in within industry sectors, such as quality control, soft sensing and predictive maintenance. Although several novel and complex techniques have been applied for predicting BOF endpoint targets, they mostly focus on static features that are only available prior blow.

In our previous study [5] we have shown that multivariate time series can be used to enhance the endpoint prediction of temperature and concentrations of C, S, P and Mn in the BOF phase of steel production. The TSFresh [15] framework was used to automatically extract the features

from the multivariate time series, which are then combined with the static data to form the input features to the model. Although such automatic feature extraction techniques are convenient, they usually lead to too many redundant and/or unnecessary features with negative influence on the performance of the model. In order to avoid this a feature selection step, based on statistical significance, was applied which only keeps the most relevant features. Regression techniques such as Gradient Boosted Regression Trees were used to fit a model on two years of data. Results show that using time series information can help to improve the prediction of temperature, P and Mn by 8%, 10% and 20%, respectively.

More recently, in [16] the authors combined 33 static variables with 81 engineered features from sensory data obtained during blow to predict the BOF endpoint using a support vector approach. Several machine learning methods were tested on a production-size dataset. Their results are competitive with the previous presented literature and reinforce the idea of introducing time series for a more accurate endpoint prediction.

Despite the significant growth of deep learning and its applications in many domains of expertise, the relevant studies in the field of steel industry are still rather scarce.

Karim et. al. [6] present a multivariate approach using Long-Short Term Memory-Fully Convolutional Networks (LSTM-FCNs) designed to process and classify multi-input data. The work presents better results than state of the art solutions on several datasets from the UCI repository, which contains MTS data from real world problems. The main contribution of this work is the addition of '*squeeze and excite*' blocks in the model, extracting more relevant information from the different data input channels. Such studies focus on classification problems, but the introduced techniques can often be used for prediction tasks, given the proper adaptations. The main gap, however, is that such models are designed to incorporate (multivariate) time series data only and not to process both static and time series data, which is commonly present in the industrial processes.

3.2 Novelty

The literature study reveals that the steel industry heavily relies on analytical models or data-driven approaches that do not take advantage of all the available sensor information, that is mostly available as multivariate time series. Despite the significant advances on multivariate time series analysis, the industrial sector tends to stick to classic and well established methodologies. One of the main reasons is the necessity for backlogging MTS data through long periods of time requiring much more data storage capacity when compared to only static data.

Fortunately, the advancements of Industry 4.0 have tackled some of these issues within the industry, with a more data oriented mindset. As a consequence, backlogging of sensor data has become more common and larger datasets are readily available. Recent literature [5, 16] focus on classic machine learning models that combine static process information and aggregates feature engineering or extraction methods for incorporating time-series information. We propose a novel approach based on the use of parallel CNNs and MLPs for handling both time-series and static information at once. Our model have several incremental advantages over feature extraction approaches. It is a more scalable solution of easier implementation that requires less pre-processing of the MTS data. Results have shown overall better prediction performance while also reducing the amount of outliers, which is highly desirable at the BOF. Also, Squeeze-and-excitation blocks are incorporated in our convolutional layers as an attention module, which might also be used later on for model explainability, to evaluate which convolutional filters and Input channels are more important.

In order to better understand the architecture of our framework, the next section briefly describes the concepts of neural networks and CNNs, further expanding on the use of Squeeze-and-excitation networks.

4 Neural Networks

Training of ANN is the procedure of finding the values of all weights and hyper-parameters such

that the desired output is generated to corresponding input. It can be viewed as the minimization of an error function computed between the output of the network and the desired output of a training observations set.

In this section we firstly discuss the Fully Connected Network (FCN), also known as Multi Layer Perceptron (MLP) [17], including how the data is processed, the architecture of the MLP and how the learning is conducted. Secondly we introduce the one dimensional Convolutional Neural Networks (CNN) and Squeeze-and-excitation (SE) [18] for temporal series. The SE block provides an attention-like mechanism and can be easily integrated into other neural blocks such as CNN (referred to as CNN-SE in this work).

4.1 Fully Connected Network

A fully connected neural network (FCN) consists of a series of fully connected layers, in which the neurons of each layer are connected to all activations of the previous layer. The activations can be computed with a matrix multiplication followed by a bias offset.

In FCN, information flows in a one-directional manner, among three types of matching layers: input, hidden, and output layers. Each layer is a function that maps an input $y_k \in \mathbb{R}^{N_k}$ to the an output $y_{k+1} \in \mathbb{R}^{N_{k+1}}$. The inner product and activation function from the k-th layer to $(k+1)$-th layers can be performed as follows:

$$S_{(k+1)} = f^*(\mathbf{W}_k \mathbf{y}_k + \mathbf{b}_k) \tag{1}$$

where f^* is an activation function which receives the product between the input vector $\mathbf{y}_k$ of $N_k \times 1$ and the weight matrix $\mathbf{W}_k$ of $N_{k+1} \times N_k$, plus the bias vector $\mathbf{b}_k$ of $N_{k+1} \times 1$. N_k and N_{k+1} are the number of neurons in the (k)-th and $k+1$-th layers respectively. Among the different possible activation functions, in this work we employ the Sigmoid function $\sigma(x) = \frac{1}{1+1^{e-x}}$ and Rectified Linear Unit (ReLU) $\delta(x) = \max(x;\ 0)$. The weights and biases are learned using the back-propagation algorithm [17] using gradient decent.

4.2 The CNN-SE network

In this section, we briefly review the concept of the Convolutional Neural Network (CNN) as well as the Squeeze-and-excitation (SE) blocks [18] adjusted for one dimensional convolutions, to match our time-series data.

The CNN block can be given by the convolution transformation $\mathbf{F}_{tr} : \mathbf{X} \rightarrow \mathbf{U}$ that maps an input $\mathbf{X} \in \mathbb{R}^{T' \times C'}$ to feature maps $\mathbf{U} \in \mathbb{R}^{T \times C}$, where T and T' are time-dimensions, while C and C' are the number of channels,

$$\mathbf{u}_c = \mathbf{v}_c * \mathbf{X} = \sum_{s=1}^{C'} \mathbf{v}_c^s * \mathbf{x}^s \tag{2}$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_C]$ results from the convolution between input vector $\mathbf{X} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{C'}]$ and spatial kernels $\mathbf{v}_c = [\mathbf{v}_c^1, \mathbf{v}_c^2, \ldots, \mathbf{v}_c^{C'}]$. $\mathbf{v}_c^s$ is a spatial kernel representing a single channel of $\mathbf{v}_c$, that acts on the corresponding channel of $\mathbf{X}$.

The SE block consists of a convolutional unit followed by the *squeeze* function, using global average pooling to generate channel-wise statistics (see Fig. 2). Statistic information $\mathbf{z} \in \mathbb{R}^C$ is provided by *squeezing* $\mathbf{U}$ through its temporal dimension T, the c-th element of $\mathbf{z}$ is then calculated as follows:

$$z_c = \mathbf{F}_{sq}(\mathbf{u}_c) = \frac{1}{T} \sum_{t=1}^{T} u_c(t) \tag{3}$$

The aggregated information obtained by the *squeeze* computation is followed by an *excite* operation, aiming to capture channel-wise dependencies,

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(\mathbf{W}_2 \delta(\mathbf{W}_1 \mathbf{z})) \tag{4}$$

Where the weights $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r} \times C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C \times \frac{C}{r}}$ are trainable parameters, σ refers to the activation function *sigmoid*, δ gating function ReLU. The dimensionality reduction factor r is used to limit the complexity of the model and can be optimized as a hyper-parameter. The experiments in [18] show that a $r = 16$ is a good trade-off between complexity and accuracy when using layers of 128 to 512 filters. The c-th channel of the output block is then rescaled as follows:

$$\tilde{\mathbf{x}}_c = \mathbf{F}_{scale}(\mathbf{u}_c, s_c) = s_c \mathbf{u}_c, \tag{5}$$

where $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_C]$ and $\mathbf{F}_{scale}(\mathbf{u}_c, s_c)$ refers to channel-wise multiplication between the scalar s_c and feature map $\mathbf{u}_c \in \mathbb{R}^T$.

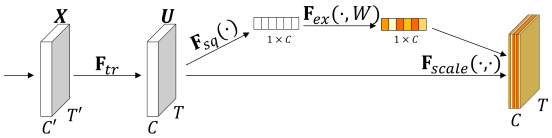


Fig. 2: Computation of the *squeeze-and-excitation* block following a CNN.

The SE block can be easily integrated into standard convolutional architectures which adaptively adjusts the input feature maps. This is comparable to a self-attention module, where the inputs values are used to calculate their own importance. It is shown that by adding SE blocks to ResNet-50 [18], a performance close to ResNet-101 can be achieved. This is impressive for a model requiring only half of the computational costs. The number of extra parameters P required to learn these SE maps can be computed as,

$$P = \frac{2}{r} \sum_{s=1}^{S} R_s \cdot G_s^2 \tag{6}$$

where S is the number of total stages, a stage is a group of consecutive layers with the same kernel size. R_s is the number of blocks repeated for stage s and G_s denoted the number of feature maps for stage s. For example, in a network with two stages, the first with two convolutional layers of 128 filters each and the second with three layers of 256, adding SE with a reduction ratio of 16 would require $P = \frac{2}{16}(2 \cdot 128^2 + 3 * 256^2) = 28672$ new trainable parameters. This means an increase of roughly 10% on the total number of parameters.

5 Methodology

In what follows, we describe the model architectures, dataset, and evaluation metrics used.

Two baselines are considered, namely, (1) the analytical model based on the physical behavior of the process which is currently being used in the factory and (2) an MLP trained only on static data. With regard to the FCN model, we define two input tensors, one for the time series and one for the static data. The former has a shape of (N, T, M) where N is the maximum number of samples (batches) in the dataset, T is the total number of time steps amongst all samples and M is the number of time signals of our MTS dataset, whereas the latter is shaped as (N, K) where K is the total number of static features used. Figure 3 depicts the proposed model, where a fully convolutional block processes the temporal data (N, T, M), while the FCN handles the static features (N, K). The output of both blocks are then concatenated and passed to a last activation layer.

In our proposed topology both time series and static data are processed in parallel by the CNN blocks and FCN block, respectively. The convolutional blocks contain three 1-dimension convolutional layers used as feature extractors, with kernel sizes of 8, 5 and 3 and number of filters are 128, 256, 128, respectively. Each layer is followed by batch normalization and a ReLU activation function. Initialization of convolution kernel's weights was made using *Uniform He* [19]. Type of padding used for the convolutional blocks was "*same*", padding with zeros evenly to both sides of the input such that output has the same width dimension as the input. Moreover, the first two blocks are followed by SE blocks in which the reduction ratio was set to $r = 16$, as suggested in the original paper [18]. This only increases the model complexity by $P = 10240$ parameters (roughly 5% relative increment in this case). The SE enhances the performance on multivariate data, as each feature map can impact the result on different degrees. This self-learned form of channel-wise attention incorporates the information of inter-correlation between multiple variables.

The FCN block is composed of a hidden layer with 64 neurons and a dropout rate of 50% to avoid overfitting, followed by a ReLU activation layer. The output from both blocks are then concatenated and are supplied to the final dense layer with a ReLU activation function. The model was trained in 250 epochs using *Mean squared logarithmic error* as loss function and *Stochastic gradient descent* as optimization function with initial *learning rate* of $1e^{-3}$.

5.1 Data Preparation

The desired quality of the steel product determines the specifications of the raw materials, such as weight and type of scrap, quantities of hot metal, iron ore and lime additions, blow time, chemical corrections and many other BOF controllable inputs and process variables. These are

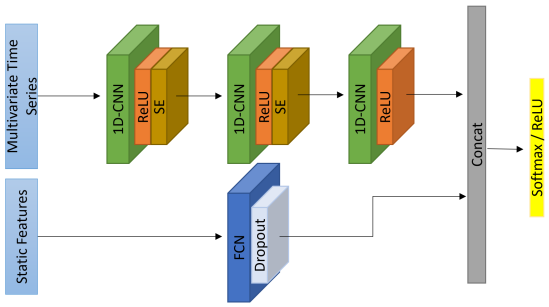


Fig. 3: The FCN-CNN-SE Model, static and time-series information are processed in parallel and then concatenated followed by one last activation layer.

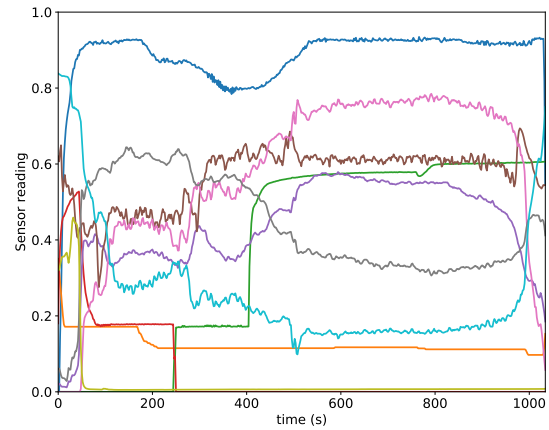


Fig. 4: Re-scaled time-series of 10 signals from one batch. The used signals in this paper are related to directly controllable variables such as oxygen flow, lance position and inert gas flow, and output ones such as amount of off-gases in the chimney, decarbonization curve and acoustic levels.

calculated using the models for carbon, phosphorus and temperature.

Once all the inputs are defined and before the blow begins, the analytical predictive models are used to estimate the endpoints of the blow, taking into account the chemical and physical characteristics and reactions for each endpoint target. Each predictive model in this phase is supplied with around 50 static variables (based on the physical and chemical behavior of the BOF process). Based on the prediction, further additions can be made during the blow to achieve the desired endpoint values.

The blowing phase lasts on average 16 minutes during which, several time-series signals are recorded. A univariate time series $\mathbf{x}$ is a one-dimensional signal (samples in a time domain) which can be defined as an ordered list of real values $[x_1, x_2, \ldots, x_{T'}]$ where T' is the total length of the signal. Usually this is the result of a sampled sensor output while monitoring a process. When a process is monitored by more than one sensor, it can be described as Multivariate Time Series (MTS), since it has more than one time-dependent variable. A MTS $\mathbf{X}$ consists of different univariate time series $[\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{C'}]$ with C' being the total amount of signals (or channels). The BOF process is highly instrumented, and several sensor readings and measurements are available during the process; usually more than 300 process-related time signals are available at this point, from which 10 are selected as most relevant by the process engineers. As data preparation the time signals are re-scaled to the range of $[0, 1]$ and normalizing the static features to have zero mean and standard deviation equal to one (training and test sets were normalized based on the values of the training set only). Fig. 4 shows the scaled signals of one heat (batch).

The available recorded data consists of the most recent 7600 heats. The usable dataset after removing corrupted or incomplete data, is composed of $N = 7158$ heats. Each sample contains $K = 54$ static features and $M = 10$ time signals with $T' = 700$ timesteps each. This data was split into 70% for training and validation, and 30% for testing. With the aim of keeping the temporal order, the samples are not shuffled before making the split. This way, we respect the realistic setup, in which a model would be deployed into production for predicting the next heat batches. It is important to note that the training set is shuffled in every epoch during training. Experiments were run on a 5-fold cross validation setting.

5.2 Evaluation setup

We consider two baselines based on static features to compare the proposed multivariate time series model with the more classic approach. The first baseline is the current mathematical model used in production (using only historical data) and the other one is an MLP block trained on the static features only. In order to study the impact of feature learning component (i.e., the

FCN-CNN-SE) we also trained a feature extraction method as used in [5], where features from multivariate time-series signals are extracted using TSFresh [15] and the 500 most important ones are joined with the static features. Then a Gradient Boosted Regression Tree (GBRT) is used to perform the regression task. Hyperparameters for this model were fine tuned using a *Grid Search* method on a 5-fold validation. The best results were obtained using 250 estimators, 3 maximum tree depth, 4 minimum sample leaf, 5 minimum sample split, learning rate of 0.1, and logarithmic loss as optimizer.

The evaluation metric is the standard Root Mean Squared Error, given by $RMSE = \sqrt{\frac{1}{q}\sum_{i=1}^{q} e_i^2}$, where q is the total number of test samples, i is the sample number and e_i is the difference between the predicted and desired values for the i-th sample. As a lower spread on prediction is the most desired outcome of this experiment, we will also present mean error and variance. Due to confidentiality reasons, all results presented here are normalized with respect to the measured values (training targets).

6 Experimental Results

In this section, we discuss the results obtained using models described in the previous section. Our deep learning *'FCN-CNN-SE'* approach and a more standard approach using feature extraction and GBRT for regression (referred to as *'Feat. Extr.'*). The analytical and MLP baseline models are referred to as *'modelled'* and *'static'*, respectively.

Figure 5 compares the prediction of the data-driven models with the analytical model for final blow temperature. The batch samples are sorted based on their measured values (training targets). Considering the fact that the target values are normally distributed, all the data-driven models perform better for the samples with average final blow temperature, whereas a negative bias is observed for the analytical model current in production. The reference model temperature is the one calculated in the beginning of the process, using the target additions. During the process, the target temperature can still be adapted, typically towards higher temperatures, leading to the

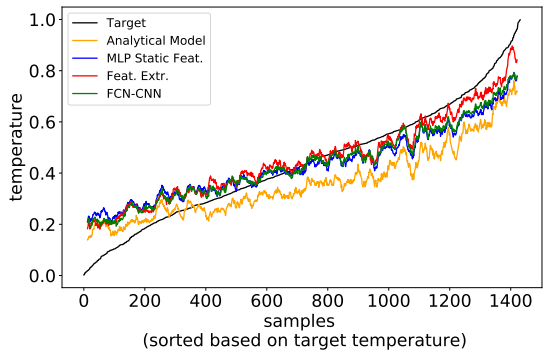


Fig. 5: Prediction results for the endpoint temperature. The black line is the measured value and the yellow, blue, red and green curves are the moving average of the results on the test set for the Analytical, Static (FCN), Feature Extraction and FCN-CNN-SE prediction models, respectively. The batch samples are ordered according to the crescent order of the samples target temperature.

observed bias. Alternative reference model temperatures can also not be considered as a fair reference due to non-causal influences on these calculations. The most fair comparison for these methods is either the standard deviation (σ) or the static model (FCN). This is due to the fact that the analytical model is tuned to be more sensitive to a low final temperature rather than high, because a final temperature that is too low results in higher costs compared to a high temperature. We can also observe that all the models struggle with the samples at both ends, while the Feature Extraction model performs slightly better on the high-end of the curve (above $1675°C$).

Figure 6 shows the absolute prediction error (°C) for the temperature, while maintaining the temporal order of the samples (batches realized in the span of approximately two months). Here we can easily notice the bias on the analytical model. We can also observe that all models approximately follow the same trend, which could point to some other external parameters that are not considered during modeling or feature engineering, *e.g.* humidity of the scrap, composition of additions and lance wear, which are inherently extremely difficult to grasp in any model. A root-cause analysis on these intervals could be an interesting research track for further model optimization but is outside of the scope of current work. It indicates

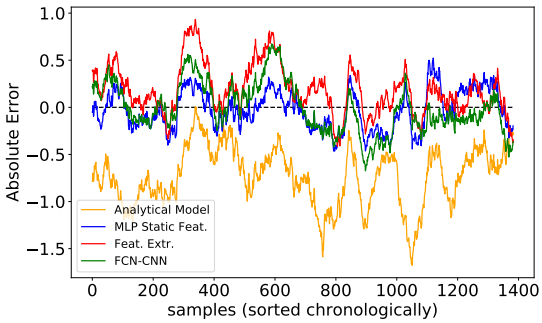


Fig. 6: Prediction error for endpoint temperature. Yellow, blue, red and green curves are the moving average of the results on the test set for the Analytical, Static (FCN), Feature Extraction and FCN-CNN-SE prediction models, respectively. The samples are arranged maintaining their temporal order

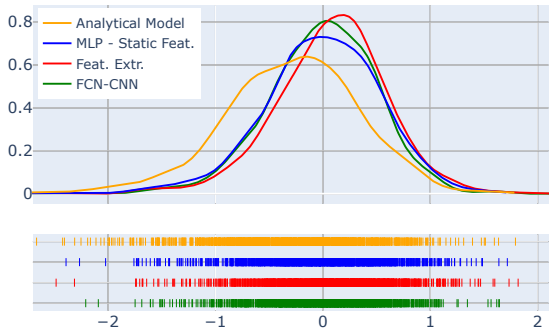


Fig. 7: Distribution curves of the error on temperature prediction.

that retraining of these models will be an essential in the deployment strategy, in order to cope with the concept drift not taken into account in the model.

The histogram presented in Fig. 7 presents the distribution of the prediction error between each model and the measured temperature. A significant gain is noticed when using more static features (*MLP Static Features*) than only the ones employed in the analytical model. Moreover, at the bottom part of the figure we can also observe that our multivariate time series approaches show lower spread. The FCN-CNN method has approximately 30% less outliers on the $1st$ and $4th$ quartiles than the Feature Extraction method.

Table 1 lists the RMSE and standard deviation for the prediction error of Temperature, concentrations of Carbon, and Phosphorus of the three regression-based models along with the analytical model that is in use at the factory. Both time-series models show an improvement over the static approaches. The FCN-CNN-SE overall has the best performance, showing a slightly smaller RMSE than the Feature Extraction approach. The deep learning approach also presented lower bias (mean), with 2.63% relative improvement on Temperature and 1.95% on Phosphorus predictions. On the other hand, the performance of the proposed model for Carbon is less appealing. This is mainly due to the fact that the in-use analytical model is specifically designed to optimize the carbon concentration by looking at some related time series data. Therefore it is not surprising that analytical and proposed methods show similar performances on this task.

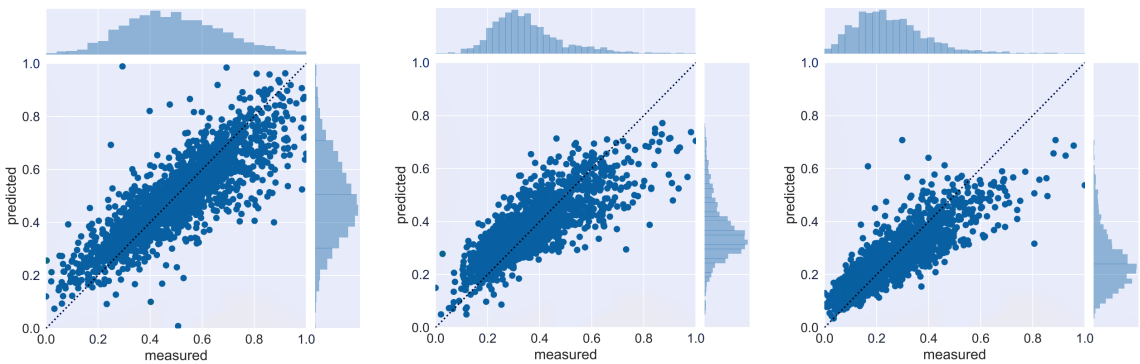
Finally, Fig. 8 shows the scattered distributions of the FCN-CNN-SE predictions and measured values for the endpoint temperature, concentrations of carbon and phosphorus. The predictions present a higher variance at high target values in all cases (cone shaped). Also in all three plots, a negative bias at this region is observed which is inline with the behaviors shown in Fig. 5 where the models struggle to predict values at the higher-end.

Processing time

Processing time is an important factor on industrial systems. Once turning point has been reached (i.e. measured CO2 gas hit a certain threshold), an updated prediction of all endpoint variables is needed for any corrections to be made in time. These calculations are desired to happen in a fraction of a second since the system response time is 1Hz (sampling time). We compared the preprocessing and inference time of Feat. Extraction and FCN-CNN-SE on an *AMD Ryzen 5 six-core processor*. While the inference times of both models were quite similar (approx. $100ms$), the preprocessing step of GBRT took $2250ms$ compared to only $270ms$ for FCN-CNN-SE. Because the signals only need to be normalized for the deep learning approach while 500 features have to be calculated for the GBRT.

Table 1: Experimental Results comparing the different modelling approaches. (0) Analytical: current models in use, based on historical data. (1) Static: FCN model trained on 57 static features. (2) Feature extraction: features are extracted from the time series – individually – and used in a data-driven model (GBRT). (3) FCN-CNN-SE: Deep learning approach using the raw dataset as input, without required additional selection methods on the extracted features. Values are normalized in respect to the Analytical values.

	Temperature		Carbon		Phosphorus	
	RMSE	σ	RMSE	σ	RMSE	σ
Analytical	1	1	1	1	1	1
Static (FCN)	0.789	0.867	0.947	0.918	0.796	0.801
Feature Extraction	0.759	0.828	0.932	0.931	0.742	0.740
FCN-CNN-SE	**0.739**	0.814	**0.926**	0.924	**0.727**	0.728



(a) Temperature (b) Carbon (c) Phosphorus

Fig. 8: Scatter distribution of FCN-CNN-SE prediction vs measured values for endpoints of (a) temperature, (b) carbon and (c) phosphorus concentrations. Graphs at the top and right sides of each image present the distribution of each variable (Predicted and Measured).

To evaluate the robustness of our model over time, training and test data were split respecting the chronological order of produced batches. We have calculated the linear trend using the prediction error over time (showed in Fig. 6) and found no significant decay in performance for any of the models, with the linear coefficient a being $\|a\| < 0.0003$ in all cases (temperature, carbon and phosphorus), over 1400 samples. Additionally, the dataset contains over 60 types of casted steel with different chemical compositions, no significant difference in error was found between different grades.

7 Discussion and Conclusion

In this paper we presented a deep learning approach for endpoint prediction on basic oxygen furnace (BOF). The goal was to predict the final temperature of the liquid steel as well as the endpoint concentrations of phosphorus and carbon, using multivariate time series and static features incorporated in a deep learning framework.

We first introduced two baseline models, (1) an FCN trained only on 54 static features and (2) a GBRT trained on 54 static features and the 500 most relevant features extracted from the 10 time series. Our proposed approach, the FCN-CNN-SE model was trained on the same 54 static features and the 10 time series. The experimental results show a clear improvement when multivariate time series are aggregated into the models; by incorporating sensor information in our models we have improved performance by 10%, relative. Our model also outperforms the more classic approach that employs feature engineering by 2.6%. On

large scale production, this improvement can significantly reduce the costs at second metallurgy and help to avoid delays on a production chain that is highly optimized.

From an industrial point of view, the proposed model do not require exceptional computational power for inference, and can be easily deployed on a real-time system. A homologous approach as the one used here for the static features (FCN) has been recently deployed and is being tested in production. In our experiments, the pipeline for extracting the features and inference for one sample was six times slower, when comparing to our method that only requires data normalization. Given the time constrains of the process control system, the proposed model would already comply with the time requirements, while the feature extraction approach would require further optimization.

While Feature Extraction and Feature engineering are well established methods, model performance is limited to the chosen features. Even if an expert agent with process knowledge could select the most relevant features to be calculated or an exhaustive list of features is used, important information might still be neglected during the feature engineering. The advantage of our method is that the entirety of the time series is used as input, hence, the model has the opportunity to learn the best representations of the data, resulting in higher performance by increasing the input space.

Although outperforming by a small margin, the FCN-CNN-SE has some advantages over the feature extraction method. It is quite easy to be employed since it requires minimal pre-processing and the only data preparation necessary is normalizing the signals. Moreover, the proposed model showed 30% less diversion (outliers in the $1st$ and $4th$ quartiles) in predicting the endpoint parameters compared to the alternative methods.

Since the *Squeeze-and-excite* blocks act as an attention module on the network, by analyzing its scalar output we can infer signal importance[18]. Furthermore, if a FCN is used for prior-blow prediction, the same network can be used on the FCN-CNN-SE (transfer learning) for a more accurate late-blow prediction.

As future work, this framework will be further tested on other use cases inside the company, as it can be easily adapted to solve classification problems and is scalable to handle more input information if necessary.

Acknowledgments. We would like to thank Carina Vercruyssen from ArcelorMittal for her valuable and constructive suggestions during the planning and development of this research work. This study was supported by Flanders Innovation & Entrepreneurship (VLAIO), in cooperation with ArcelorMittal and Ghent University-imec under grant HBC.2019.2173. A. Jalalvand would like to thank the Special Research Fund of Ghent University for funding his research (BOF19/PDO/134).

References

[1] Tuncel, K.S., Baydogan, M.G.: Autoregressive forests for multivariate time series modeling. Pattern Recognition **73**, 202–215 (2018). https://doi.org/10.1016/j.patcog.2017.08.016

[2] Fawaz, H.I., Forestier, G., Weber, J., Idoumghar, L., Muller, P.-A.: Deep learning for time series classification: a review. Data Mining and Knowledge Discovery **33**(4), 917–963 (2019). https://doi.org/10.1007/s10618-019-00619-1

[3] Bagnall, A., Lines, J., Bostrom, A., Large, J., Keogh, E.: The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data mining and knowledge discovery **31**(3), 606–660 (2017)

[4] Ruiz, A.P., Flynn, M., Large, J., Middlehurst, M., Bagnall, A.: The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **35**(2), 401–449 (2021)

[5] Sala, D.A., Jalalvand, A., Van Yperen-De Deyne, A., Mannens, E.: Multivariate time series for data-driven endpoint prediction in the basic oxygen furnace. In: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 1419–1426 (2018). https://doi.org/10.1109/ICMLA.2018.00231

[6] Karim, F., Majumdar, S., Darabi, H., Harford, S.: Multivariate lstm-fcns for time series classification. Neural Networks **116**, 237–245 (2019)

[7] Yang, S., Gao, T., Wang, J., Deng, B., Azghadi, M.R., Lei, T., Linares-Barranco, B.: Sam: A unified self-adaptive multicompartmental spiking neuron model for learning with working memory. Frontiers in Neuroscience **16** (2022). https://doi.org/10.3389/fnins.2022.850945

[8] Fauvel, K., Lin, T., Masson, V., Fromont, E., Termier, A.: Xcm: An explainable convolutional neural network for multivariate time series classification. Mathematics **9**(23) (2021). https://doi.org/10.3390/math9233137

[9] Han, M., Cao, Z.J.: An improved case-based reasoning method and its application in endpoint prediction of basic oxygen furnace. Neurocomputing **149**(PC), 1245–1252 (2015). https://doi.org/10.1016/j.neucom.2014.09.003

[10] Liu, L., Li, P., Chu, M., Gao, C.: End-point prediction of 260 tons basic oxygen furnace (bof) steelmaking based on wnpsvr and woa. Journal of Intelligent & Fuzzy Systems (Preprint), 1–15 (2021)

[11] Ruuska, J., Sorsa, A., Lilja, J., Leiviskä, K.: Mass-balance Based Multivariate Modelling of Basic Oxygen Furnace Used in Steel Industry. International Federation of Automatic Control **50**(1), 13784–13789 (2017). https://doi.org/10.1016/j.ifacol.2017.08.2065

[12] bing Wang, H., jun Xu, A., xiang Ai, L., yuan Tian, N.: Prediction of Endpoint Phosphorus Content of Molten Steel in BOF Using Weighted K-Means and GMDH Neural Network. Journal of Iron and Steel Research International **19**(1), 11–16 (2012). https://doi.org/10.1016/S1006-706X(12)60040-5

[13] Laha, D., Ren, Y., Suganthan, P.N.: Modeling of steelmaking process with effective machine learning techniques. Expert Systems with Applications **42**(10), 4687–4696 (2015). https://doi.org/10.1016/j.eswa.2015.01.030

[14] Sorsa, A., Ruuska, J., Lilja, J., Leiviskä, K.: Data-driven multivariate analysis of basic oxygen furnace used in steel industry. IFAC-PapersOnLine **28**(17), 177–182 (2015). https://doi.org/10.1016/j.ifacol.2015.10.099

[15] Christ, M., Braun, N., Neuffer, J., Kempa-Liehr, A.W.: Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). Neurocomputing **307**, 72–77 (2018). https://doi.org/10.1016/j.neucom.2018.03.067

[16] Bae, J., Li, Y., Ståhl, N., Mathiason, G., Kojola, N.: Using machine learning for robust target prediction in a basic oxygen furnace system. Metallurgical and Materials Transactions B **51**, 1632–1645 (2020)

[17] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)

[18] Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7132–7141 (2017)

[19] He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2015)