Enhancing the Stationary State Prediction in Model Predictive Control systems to avoid Dross Defect in Heavy-section Foundries

Javier Nieves, Igor Santos, Pablo G. Bringas
S^{3}Lab, DeustoTech - Computing,
University of Deusto,
Bilbao, Spain
Email: {jnieves, isantos, pablo.garcia.bringas}@deusto.es

Argoitz Zabala, Jon Sertucha
Área de Ingeniería, I+D y Procesos Metalúrgicos
IK4-Azterlan,
Durango, Spain
Email: {azabala, jsertucha}@azterlan.es

***Abstract*—A Model Predictive Control (MPC) is a system designed to control a production plant. These systems are composed of several phases, being one of the most important ones the phase for the prediction of the plant situation in a given time. In a previous work, we presented a machine-learning approach for this prediction phase that replaced the need of developing a single mathematical function with a more generic classification approach. However, standalone classifiers had some drawbacks like to select the most adequate classification models for the learning data and task. In this paper we extend our previous work with a general method to foresee Dross defects building a meta-classification system through the combination of different methods and removing the need of selecting the best algorithm for each objective or dataset.**

I. Introduction

Manufacturing processes exist since ancient times. Over time, the manufacturing process has allowed transforming raw materials into finished products. Manufacturing has been improved due to the technological progress and the evolution of our society.

Thus, and with the help of multiple disciplines of knowledge, it is possible to anticipate several problems in the manufacturing process, avoiding them. The goals of these systems are: (i) minimise costs, (ii) maximise the production rate, (iii) minimise the stock, (iv) control the price fluctuation and (v) avoid service failures caused by hidden product defects. Typically, an exhaustive production control is carried out in order to ensure that the results achieved in the measurements of the aforementioned objectives are within normality. Unfortunately, many of them can only be measured once the products have already been finished.

As an improvement of the manufacturing process, the engineers try to determine the outcome of the final product before even producing it. The so-called Model Predictive Control systems (MPC) [1] are method designed for these goals. These algorithms are capable of controlling a specific manufacturing process. Besides, their objective is to try to keep the production of the plant under a normality, tuning the variables of the process to fulfil the constraints defined during the design of the system. Originally, these systems were developed to control the specific needs of a few industries, i.e., power plants and oil refineries. However, MPC technology is currently used in a wide range of industries such as chemical, food processing, automotive and aerospace applications [2].Despite its popularity, MPC systems have several limitations and some of them are related with the prediction step.

A way to reduce their drawbacks is the employment of machine learning methods: standalone classifiers are capable of obtaining good results in the prediction of several defects in the manufacturing process. However, there are several limitations of this approach: (i) we cannot be sure that the selected classifier is the best one to generalise the manufacturing process, (ii) the learning algorithms employed for creating some machine-learning classifiers only find a local maximum, hence, the final result may not be the optimal one and (iii) the nature of a standalone classifier is unique and, therefore, that classifier might not fit to the nature of the manufacturing process (linear or non-linear). A solution for these problems is the combination of standalone classifiers.

Against this background, we present here a meta-classification technique to enhance the prediction stages in an MPC system. These methods are able to learn from labelled data to build accurate classifiers that are going to share its knowledge under some rules. This approach allows removing some limitations that the common MPC systems have and avoids the problems that the employment of standalone classifiers produces in the stationary state prediction stage. Moreover, the paper explains the MPC controller work-flow and how this method can be applied.

In order to test this approach, we have selected the foundry process because it is considered as one of the main factors influencing the development of the world economy [3]. The selected use case is focused on heavy-section foundries, trying to avoid non-metallic inclusions with elongated and filamentary aspect usually surrounded by degenerated graphite (lamellar and/or vermicular shapes), also known as Dross [4]. Besides, the improvement of the prediction step for this parameter would lead foundries to lower reject rates and the subsequent cost and time saving.

The remainder of this paper is organised as follows. Section

II summarises the state-of-the-art related to Model Predictive Control systems. Section III details the casting production process and presents the prediction target, the Dross defect. Section IV describes different methods for combining classifiers and how they can be adopted for foreseeing the stationary state in the foundry plant. Section V describes the experiments and presents results. Finally, Section VI concludes the paper and outlines avenues for future work.

II. Model Predictive Control System

A. Definition

Model Predictive Control systems (MPC) are one of the few advanced methods that had a significant impact on manufacturing control engineering. They are applied to the manufacturing industry [5], [6] because they allow: (i) management of multivariable control problems in a natural way, (ii) fault management in the actuators, (iii) identification of problematic situations related to phases of the process, due to its ability to predict the impact of the changes to the current process and (iv) to take into account the uncertainty of the model (also known as the mismatch of plants/predictive models).

The methods for the prediction of the control plant were developed based on several common ideas [7], [8] such as the following ones:

- The employment of an explicit model to foresee the process outputs over a time horizon $t+1$, which is set in the near future.
- To retrieve a control sequence that minimises a cost function, which will be the objective that the MPC will try to optimise.
- To apply calculated control signals based on the results achieved for the prediction systems.

The MPC system control technology is responsible for driving the process from the current state to a different one that improves the current situation of the plant. This control process is the way to keep the plant producing within normality and correction. The main objectives to control, in order of importance, are as follows [1]:

1) Avoiding that inputs and outputs violate the constraints defined for the production process.
2) Monitoring variables that cannot be modified in order to know their state in every moment.
3) Directing the variables that can be modified in order to calculate the optimal parameters for the manufacturing process.
4) Avoiding sharp changes in the process.
5) Keeping a minimal control of the manufacturing process.

These objectives define the work flow that an MPC system must carry out, shown in Fig. 1. More accurately, the control steps are the following.

- **Reading values from the process.** The controller performs the reading of all the variables, both types, those it can modify and those it cannot.
- **Estimate the stationary state.** At this time, the controller is responsible for determining the state of the manufacturing process in an instant of time $t+1$, if the plant continues producing with the same configuration. In this paper we have focused on the optimisation of this task.
- **Determination of the target state.** In this step, the controller tries to establish the optimal state in which we want the manufacturing plant to work, avoiding major changes in the process when the redirection is done.
- **Determination of the optimisation path.** This is one of the most important stages. Its result will be the method to modify the plant. Thus, in this working step, firstly, the MPC has to select one of the possible trajectories to arrive the desired state (e.g., multiple movements, a single movement or an adaptation based on a function). Subsequently, the system avoids generating values that break the constraints of the plant through several techniques to calculate the penalisation of the proposed solution, such as the definition of restricted points, areas, reference trajectories or funnels.
- **Outputs for the variables to modify.** Finally, in this last stage, the feedback to the plant is carried out.

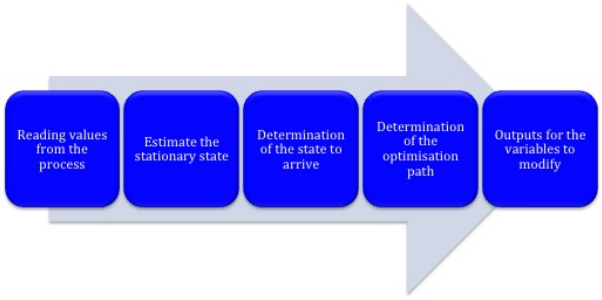

Fig. 1. General work-flow followed by MPC controllers for each of their executions.

B. Limitations

The majority of the current MPC suffers from limitations generated by the basic theory behind them and the guidelines created by their historical predecessors. Some of the most important limitations are listed below [9]:

- Limited options to choose the model.
- Inefficient feedback to the plant.
- The special attention of experienced engineers and domain experts is required for the design and tuning of MPC systems.
- There is no method to determine if the gathered data is fully representative of the process.
- No statistical efficiency is addressed.
- There is a lack of methods to validate the prediction systems.
- There is no systematic approach to the construction of the MPC.
- The formulation of the controller is difficult .
- There are major problems to adapt the models to the changes of the manufacturing process.
- The employment of non-linear predictive models is very poor.

- There are no major advances in the generation of hybrid MPC systems that are able to work with both discrete and continuous variables simultaneously.

The research presented in this paper avoids some of these limitations. Specifically, it increases the current options for building the predictive model, while we allow the use of linear and non-linear predictive models simultaneously. Furthermore, a systematic approach is proposed to generate these models through employing machine learning methods without involving domain experts.

III. Use case: Dross inclusions in the Manufacturing of Heavy-section Cast Parts

Foundry is a factory where metal is melted and poured into specially shaped containers to produce different cast parts with complex designs and close to their final shape. Although all of the foundry processes are not equal, the work-flow performed in foundries is very similar to the explained below [10].

1) *Pattern, Mould and Core making.* On one hand, moulding and core-making patters respectively provides the exterior and interior shape of the casting. The production of heavy-section castings requires big moulds, which are regularly created employing chemical-bonded sand mixtures. Currently, the preparation of the moulds is a handmade process.
2) *Melting, Magnesium Treatment and Inoculation.* In this step, the designed metallic charges are incorporated in a furnace. After melting, the chemical composition is checked and adjusted to the previously known requirements. Then, in order to obtain the correct or desirable graphite particles shape, it is added a *nodularising agent*. When the reaction is accomplished, the achieved batch is tested. Finally, due to it is necessary to guarantee the spherical shape of the graphite and obtain a high nodule counts, an inoculation process must be carried out just before pouring the mould.
3) *Pouring Process.* Before starting the mould filling, the treated and inoculated melt is usually placed in a basin located in the top of the mould. Once the mixture is ready, the molten material is poured onto the sand mould. Later, the metal begins to cool. This step is one of the most important because the majority of the defects can appear during this phase.
4) *Cast Part Separation.* Once the cast part is cool enough (e.g., for ferritic heavy-section castings, the temperature must be lower than $600^{\circ}\mathrm{C}$), the sand is removed by shaking the whole set in a grid. The removed sand is recovered for further uses.
5) *Quality Assurance and Finishing Step.* After the previous step, filling channels and feeding systems are removed from the cast part. Then, the residual sand is removed by shot-blasting. Finally, several controls are performed in order to determine the validity of the cast part according to the costumer requirements. If some defects appear in the casting, it will require additional cleaning operations to eliminate all affected areas and to fulfil the customer requirements. This tasks strongly increase the cost of each cast part. If the defects cannot be removed, the casting will be rejected. Finally, other finishing operations are carried out (machining, painting, among others), although this operations are not directly linked to the foundry process.

During the whole process, several defects can be produced in the castings. One of them is Dross inclusions. The term 'Dross' is usually referred to an internal slag inclusion, oxides formed by the reaction of different chemical elements involved in the nodularisation and inoculation treatments, with elongated and filamentary aspect surrounded by degenerated graphite particles.

The apparition of slag inclusion is usually produced by a bad cleaning of the Magnesium treatments before pouring. Although this fact, Dross defect can also appears even when correctly skimmed melts are used. In this case, it is produced in the internal mass of liquid irons before solidifying. Foundry plants have identified three causes that are related with this defect: (i) chemical composition of melt, (ii) turbulences when filling the moulds and (iii) pouring temperatures. To characterise the foundry process, we have selected more variables (variables extracted from the whole manufacturing process and related to the aforementioned steps) since monitoring only these factors may not be enough to detect Dross defects. To this end, we apply a simplification of the foundry process developed by domain experts, representing the castings with 120 different variables, as we did in previous work [10].

IV. Evolving the Prediction of the Stationary State through a Combination of Machine-Learning Classifiers

Classifiers by themselves are able to obtain good results, but as discussed above, it is difficult to assure that a specific classifier is particularly suitable for the prediction of one defect. In order to avoid this problem, several studies propose the combination of classifiers as a technique that was born with the aim of obtaining better classification decisions despite of incorporating a higher degree of complexity to the process [11].

From a statistical point of view [12], assuming a labelled dataset $\mathbf{Z}$ and a n number of different classifiers with relatively good performance in making predictions for $\mathbf{Z}$, we can select one of them to face a classification problem, but there is a risk of choosing a bad one. Therefore, the safest option is to use all of them and average their outputs. The resulting classifier is not necessarily better than the best classifier, but will decrease or eliminate the risk of the use of a non appropriate classifier for the task.

From a computational point of view [11], some supervised machine-learning algorithms, in their learning phase, generate models based on local maximum solutions. Thus, an aggregation of classifiers is much closer to the optimal classifier than only one of them.

Although there are different combination methods, we have discarded those methods that do not allow us to generate a

collective intelligence system for classification, which incorporates both linear and non-linear classifiers.

A. By Vote

The democracy for classifying elements is one of the oldest strategies for decision making. The methods employed for this research are:

- **Majority Voting Rule.** Defining that the labelled outputs of the classifiers are c-dimensional binary vectors $[d_{i,1}, ..., d_{i,c}]^T \in \{0,1\}^c, i = 1, ..., L$ where $d_{i,j} = 1$ if the classifier D_i determines $\mathbf{x}$ such as ω_j, or 0 otherwise, the combination of votes results in a set of classification for the class ω_k calculated as follows.

$$\sum_{i=1}^{L} d_{i,k} = \max_{j=1}^{c} \sum_{i=1}^{L} d_{i,j} \tag{1}$$

- **Product Rule.** Here, the probabilities are taking into account for the final calculation [13]. This rule represents the joint probability distribution of the measurements taken from the classifiers. Moreover, it is assumed that the representations are conditionally statistically independent. The product rule quantifies the probability of a hypothesis by combining the *a posteriori* probabilities generated by the classifiers. Therefore, we assign the label $Z \rightarrow \omega_j$ if

$$P^{-(R-1)}(\omega_j) \prod_{i=1}^{R} P(\omega_j|x_i) = \max_{k=1}^{m} P^{-(R-1)}(\omega_k) \prod_{i=1}^{R} P(\omega_k|x_i) \tag{2}$$

- **Average Rule.** This rule is derived from the Sum Rule (see equation 3) to, subsequently, make a division by the number of standalone classifiers employed, R, as denominator [13].

$$(1-R)P(\omega_j) + \sum_{i=1}^{R} P(\omega_j|x_i) = \max_{k=1}^{m} \left[(1-R)P(\omega_k) + \sum_{i=1}^{R} P(\omega_k|x_i) \right] \tag{3}$$

- **Max Rule.** For the Max Rule [13], we start with the Sum Rule (see equation 3) and obviating the product of *a posteriori* probabilities and assuming prior equalities, we will assign $Z \rightarrow \omega_j$ if

$$\max_{i=1}^{R} P(\omega_k|x_i) = \max_{k=1}^{m} \max_{i=1}^{R} P(\omega_k|x_i) \tag{4}$$

- **Min Rule.** For the Min Rule [13], we start with the Product Rule (see equation 2) and obviating the product of *a posteriori* probabilities and assuming prior equalities, we will assign $Z \rightarrow \omega_j$ if

$$med_{i=1}^{R}\ P(\omega_j|x_i) = \max_{k=1}^{m} med_{i=1}^{R}\ P(\omega_k|x_i) \tag{5}$$

B. Grading

In this method, the standalone classifiers are evaluated using the k-fold cross-validation [14]. *Grading* builds n_c training datasets, one for each standalone classifier k, adding the predictions g_{ik} to the original dataset as the new class. $prMeta_{ik}$ is the probability that the standalone classifier k is going to foresee the instance i and is calculated by the meta-classifier of k. In this way, the final probability for the class l and the instance i, if there is, at least, one meta-classifier which is going to foresee the result in a correct manner (i.e., $prMeta_{ik} > 0.5$), is calculated by the following formula:

$$prGrd_{il} = \sum \{prMeta_{ik} | c_{ik} = l \wedge prMeta_{ik} > 0.5\} \tag{6}$$

where $c_{ik} = argmax_l\{p_{ikl}\}$, in other words, the prediction of the standalone classifier k for i using the maximum likelihood.

The classification step is as follows [15]. First, each standalone classifier makes a prediction. Second, the obtained results are qualified. And finally, the classification is derived using only the positive results. Conflicts are solved using the *by vote* approach.

C. Stacking

The *stacking* method [16] is another generalisation based on the k-fold cross validation combination. For the learning phase of standalone classifiers, the training set θ is divided into k partitions and they are in turn split in 2 sets, usually disjoints $(\theta_{i1}, \theta_{i2})$. The first set of θ_{i1} define the space of level 0. Then, for each k_i partition of θ, it is generated a set of k transformation numbers that represents either (i) the assumptions made by standalone classifier or (ii) the input component θ_{i2} or (iii) the vector to connect both θ_{i1} and θ_{i2}. These points are the training set for the *level 1* meta-classifier.

For the classification process, firstly, we carry out a question to the classifiers in level 0. Secondly, we apply the transformations to produce the input dataset for the level 1. Subsequently, level 1 meta-classifier derive the solution. And finally, the response is transformed back into the level 0 space to provide the final result. The whole process is known as *stacked generalisation* and can be more complex adding multiple stacking levels.

D. MultiScheme

This method employs a combination rule based on the results obtained by the cross validation. MultiScheme method is performed through the calculation of the mean for each instance i and the error rate (i.e., the mean square error) of the standalone classifier G. Thus, using this measures, MultiScheme method is able to determine which classifier should be the most accurate. Specifically, the error estimation is made as follows.

$$CV(G, \theta) \equiv \frac{\sum_i [G(\theta_{i1}, \text{input of } \theta_{i2})(\text{output of } \theta_{i2})]^2}{m} \tag{7}$$

V. Experimental Results

In order to test our method, a dataset from a real foundry specialised in heavy-section nodular iron castings has been collected, accurately, in bushings for wind turbines. Cast parts that contain this defect must be rejected due to very restrictive quality standards. The gathered dataset contains attributes from the whole foundry process. In fact, it includes 120 different parameters related with raw materials, chemical composition or melt, thermal analysis data, inoculation, magnesium treatment, pouring time and pouring temperature, among others. We have worked with 89 different references.

This dataset was labelled with the volume affected by Dross defect measured in 2261 ultrasonic inspections. With the aim of evaluating the precision of the selected meta-classification methods, the final level of Dross was discretised in 5 different categories, applying a normal distribution-based discretisation. The ranges are the followings.

1) $Dross \leq 2.5$
2) $2.5 < Dross \leq 3.5$
3) $3.5 < Dross \leq 4.5$
4) $4.5 < Dross \leq 6.5$
5) $Dross > 6.5$

Specially, we have conducted the next methodology in order to evaluate properly the combination of classifiers:

- **Cross validation:** We have performed a *k-fold cross validation* [14] with $k = 10$. In this way, our dataset is 10 times split into 10 different sets of learning.
- **Learning the model:** We have made the learning phase of each algorithm with each training dataset, applying different parameters or learning algorithms depending on the model. More accurately, we have used the same classifiers that we presented in [10].
- **Learning the combination of the classifiers:** After the previous task, we employed the aforementioned combination methods. In this paper, we tested the following meta-classification methods:
  - *By vote:* For these experiments, we have used *the majority vote rule*, *the product rule* , *the average rule* , *the max rule* and *the min rule* [11] [13].
  - *Grading:* We have performed our experiments using the following first level classifiers: *a Naïve Bayes*, *Tree Augmented Naïve*, a *K-Nearest Neighbour* where $1 \leq k \leq 5$ and a *Support Vector Machine* (SVM) with Polynomial Kernel.
  - *Stacking:* To create the second space classifiers, we have tested the same classifiers used for creating the first level classifiers in the Grading method.
  - *MultiScheme:* In this research, we have used this meta-classifier as it is, combining the results using the cross validation outputs and the error rates.
- **Testing the model:** For each combination method, we have evaluated the percent of correctly classified instances and the area under the ROC curve (AUC), which represents the relation between false negatives and false positives [17].

TABLE I
Results of Dross Prediction in terms of Accuracy and Area Under the ROC Curve (AUC) using standalone classifiers [10].

Classifier	Accuracy(%)	AUC
Naïve Bayes	47.86±3.06	0.7690±0.05
BN: K2	62.63±2.94	0.8939±0.03
BN: TAN	74.92±2.81	0.9560±0.02
KNN K = 1	30.20±2.89	0.5888±0.05
KNN K = 2	30.67±2.78	0.6323±0.05
KNN K = 3	30.25±2.93	0.6602±0.05
KNN K = 4	31.22±2.89	0.6672±0.05
KNN K = 5	30.99±2.82	0.6753±0.05
SVM: Polynomial Kernel	57.07±2.76	0.8135±0.04
SVM: Normalised Polynomial Kernel	50.92±3.08	0.7779±0.04
SVM: RBF Kernel	44.78±2.79	0.7165±0.04
SVM: Pearson VII Kernel	62.59±2.72	0.8239±0.04
DT: J48	62.97±3.23	0.8157±0.06
DT: RandomForest N = 10	70.23±3.00	0.8999±0.03
DT: RandomForest N = 20	72.54±2.55	0.9223±0.02
DT: RandomForest N = 30	73.33±2.56	0.9296±0.02
DT: RandomForest N = 40	73.65±2.39	0.9334±0.02
DT: RandomForest N = 50	73.87±2.31	0.9356±0.02

On one hand, Table I shows the results achieved in our previous research [10] for Dross prediction in terms of accuracy and AUC. More accurately, we can realise that the best classifiers in that experiment were the Bayesian networks trained employing Tree Augmented Naïve (with an accuracy level of 74.92% and an AUC of 0.9560) and random forests (achieving an accuracy bigger than 70% and an AUC bigger than 0.92).

TABLE II
Results of Dross Prediction in terms of Accuracy and Area Under the ROC Curve (AUC) using meta-classifiers.

Classifier	Accuracy(%)	AUC
Stacking (TAN)	71.68±3.76	0.9422±0.02
Stacking (Naïve Bayes)	68.31±2.86	0.9275±0.03
Stacking (KNN k=1)	65.64±3.17	0.7665±0.05
Stacking (KNN k=2)	65.59±2.95	0.8273±0.04
Stacking (KNN k=3)	69.31±2.67	0.8501±0.04
Stacking (KNN k=4)	70.82±2.65	0.8662±0.04
Stacking (KNN k=5)	71.57±2.43	0.8774±0.04
Stacking (J48)	67.17±3.48	0.7991±0.08
Stacking (SVM with Polynomial Kernel)	75.61±2.73	0.9297±0.03
Grading (TAN)	26.16±3.77	0.4985±0.01
Grading (Naïve Bayes)	29.77±4.31	0.4881±0.02
Grading (KNN k=1)	24.50±3.77	0.4947±0.01
Grading (KNN k=2)	24.04±3.62	0.4939±0.01
Grading (KNN k=3)	24.73±3.78	0.4951±0.01
Grading (KNN k=4)	24.62±3.86	0.4964±0.01
Grading (KNN k=5)	25.89±3.73	0.4946±0.01
Grading (SVM with Polynomial Kernel)	25.22±2.19	0.5000±0.00
MultiScheme	74.60±2.66	0.9471±0.02
By Vote (Average Rule)	56.45±3.37	0.8645±0.04
By Vote (Product Rule)	20.52±5.62	0.4985±0.01
By Vote (Majority Voting Rule)	30.88±2.93	0.5372±0.03
By Vote (Min Rule)	20.44±5.56	0.4985±0.01
By Vote (Max Rule)	61.06±8.43	0.8422±0.05

On the other hand, Table II illustrates the results accomplished in our experiment applying meta-classification methods. In this case, we can realise that the best meta-classification

method, specifically, stacking with a SVM with polynomial kernel as second space classifier; enhances the best standalone classifier in terms of accuracy. Actually, the *stacking* method accomplish better predictions than other methods. Nevertheless, this meta-classification method is nearly followed by one of the simplest one, the MultiScheme. Furthermore, although the By Vote method does not obtain the same results, it is really close to the worst stacking method.

Regarding the area under the ROC curve, the meta-classifiers cannot reach the same results of the standalone classifiers. However, the calculated AUC of the best meta-classifier is really close to the previous ones'. Precisely, the stacking method that uses Tree Augmented Naïve gets 0.9422, the best method in terms of accuracy level achieves 0.9297 and MultiScheme reaches an AUC of 0.9471.

Due to the achieved results have a great degree of similarity, the employment of meta-classifiers are able to foresee the Dross defect in high precision foundries. In our case, the good results achieved by the *stacking* method that uses SVM with polynomial kernel show that this approach can handle the stationary state $t+1$ prediction as we did in our previous work with other defects (i.e., microshrinkages and mechanical properties) [18]. Therefore, combining several meta-classification methods, firstly, we can reduce the cost and the duration of the actual testing methods and, secondly, we can improve the prediction step of our new generation MPC system that employs machine learning techniques.

This research can be deployed, also, as a prediction tool that can work in the following way: (i) the operator carries out a prediction experiment to determine the final result of the casting regarding the Dross problem, (ii) if the volume of the Dross is bigger than the permissible threshold, the operator might change the manufacturing parameters to adjust the production or, even, he can produce another reference.

VI. Conclusions

The Dross defect is a non-metallic inclusion with elongated and filamentary aspect usually surrounded by a degenerated graphite. Unfortunately, this defect has to be tested once the production is finished, which implies a re-manufacturing of the casting when the Dross defect is found.

Hence, our previous research [10], which pioneers the application of supervised machine-learning methods to discover the apparition of this defect in advance, has been extended in this paper, presenting a new way for enhancing the stationary state prediction in MPC systems through the combination of several standalone classifiers.

Although some meta-classifiers do not achieve good results, there are a set of them that approximates the results achieved by the best standalone classifiers. In fact, the *stacking* built with a Support Vector Machine, which employs a Polynomial Kernel, outperforms the best standalone classifier in terms of accuracy. Regarding the error rates, the meta-classifier cannot improve the results achieved by the Bayesian networks, nevertheless, they are really close. These results illustrate the theory presented by Kuncheva [11].

Consequently, further work will be focused in three main directions. First, we plan to extend our tests to be able to predict other defects with the aim of generating a global system of incident analysis. Second, we plan to carry out a research to find a quick manner of updating the prediction models. And finally, we plan to test this prediction models in our optimisation algorithm [19], generating a complete MPC system that can be deployed in a real foundry plant.

References

[1] E. Camacho and C. Bordons, *Model predictive control*. Springer Verlag, 2004.

[2] S. Qin and T. Badgwell, "A survey of industrial model predictive control technology," *Control engineering practice*, vol. 11, no. 7, pp. 733–764, 2003.

[3] "46^{th} census of world casting production," Modern Casting, Tech. Rep., 2012, monthly report, edited annually with the data concerning the number of casting houses and the world casting production in the year preceding the issue.

[4] M. Gagne, M. Paquin, and P. Cabanne, "Dross in ductile iron: Source, formation and explanation," *Indian Foundry Journal*, vol. 57, no. 9, pp. 39–44, 2011.

[5] J. Bekker, I. Craig, and P. Pistorius, "Model predictive control of an electric arc furnace off-gas process," *Control Engineering Practice*, vol. 8, no. 4, pp. 445–455, 2000.

[6] R. Abou-Jeyab, Y. Gupta, J. Gervais, P. Branchi, and S. Woo, "Constrained multivariable control of a distillation column using a simplified model predictive control algorithm," *Journal of Process Control*, vol. 11, no. 5, pp. 509–517, 2001.

[7] J. Maciejowski, *Predictive control: with constraints*. Pearson education, 2002.

[8] G. Goodwin, M. Seron, and J. De Doná, *Constrained control and estimation: an optimisation approach*. Springer Verlag, 2005.

[9] D. Kassmann, T. Badgwell, and R. Hawkins, "Robust steady-state target calculation for model predictive control," *AIChE Journal*, vol. 46, no. 5, pp. 1007–1024, 2000.

[10] I. Santos, J. Nieves, P. Bringas, A. Zabala, and J. Sertucha, "Supervised learning classification for dross prediction in ductile iron casting production," in *Proceedings of 8th IEEE Conference in Industrial Electronics and Applications (ICIEA), Melbourne (Australia)*, 2013.

[11] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.

[12] T. Dietterich, "Ensemble methods in machine learning," *Multiple classifier systems*, pp. 1–15, 2000.

[13] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On combining classifiers," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 3, pp. 226–239, 1998.

[14] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence*, vol. 14, 1995, pp. 1137–1145.

[15] A. Seewald and J. Fürnkranz, "An evaluation of grading classifiers," *Advances in Intelligent Data Analysis*, pp. 115–124, 2001.

[16] D. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[17] Y. Singh, A. Kaur, and R. Malhotra, "Comparative analysis of regression and machine learning methods for predicting fault proneness models," *International Journal of Computer Applications in Technology*, vol. 35, no. 2, pp. 183–193, 2009.

[18] J. Nieves, I. Santos, , and P. G. Bringas, "Combination of machine-learning algorithms for fault prediction in high-precision foundries," in *Database and Expert Systems Applications - 23rd International Conference, DEXA 2012*, ser. Lecture Notes in Computer Science, S. W. L. al., Ed. Springer Berlin / Heidelberg, 2011, vol. 6861, pp. 56–70.

[19] J. Nieves, I. Santos, and P. Bringas, "Evolutionary programming to adjust the process in high precision foundries," in *Proceedings of 8th IEEE Conference in Industrial Electronics and Applications (ICIEA), Melbourne (Australia)*, 2013, pp. 912–917.