Broiler Weight Forecasting using Dynamic Neural Network Models with Input Variable Selection

Simon V. Johansen[a,b,*], Jan D. Bendtsen[b], Martin R.-Jensen[a], Jesper Mogensen[a]

[a]*SKOV A/S, Hedelund 4, Glyngøre, Denmark*
[b]*Department of Control and Automation, Aalborg University, Denmark*

Abstract

The dynamic influence of environmental broiler house conditions on broiler growth is investigated in this work. 18 forecasting trials have been carried out on ensemble dynamic neural network models trained on farm-scale broiler batch production data with automatic mutual information based input variable selection. The model forecasts future broiler behavior comprised of broiler weight, feed consumption and water consumption based on environmental condition inputs such as heating, ventilation, and temperature from a previous batch. We present a case study on a recent forecasting trial, where the training and evaluation data are analyzed along with the forecasts. We analyze trends in the input variable selection and evaluate the forecasting performance of the 18 trials. It was found that useful forecasts can be obtained by using input data from the previous batch as a substitute for unknown future input data.

Keywords: agricultural engineering, biological system modeling, recurrent neural networks

1. Introduction

Due to the growing middle class in many developing countries, the increase in meat consumption is predicted to continue rising (OECD-FAO, 2016, pp. 107). The highest growth in meat production is foreseen in poultry meat, of which we expect broiler meat to represent the majority. The reason being that broiler production gives the highest yield per feed unit among land animals, making chicken meat a relatively inexpensive source of animal protein. The poultry industry on a world scale is predicted to steadily increase, from an average of 110.3 billion kg of meat in 2013-2015 to 131.3 in 2025 (OECD-FAO, 2016, pp. 127) – a significant predicted increase of 19%.

Available scientific literature on dynamic broiler models focuses mainly on active broiler weight control by regulating feed uptake and composition, which traditionally favors simplistic models (Wathes et al., 2008; Aerts et al., 2003). Feed is generally considered the biggest expense in broiler production, and correct climate control is known to yield superior feed utilization for broiler growth. However, no prior scientific modeling literature studying the complex dynamic interconnection between broiler weight and broiler house environmental conditions has been found, and will thus be the subject of this work. Such a model has the potential of allowing active broiler weight control by regulating the broiler environment, which is nonexistent in industry today.

Traditionally, empirically motivated nonlinear growth curve models have been used to determine evolution of broiler weight. It has been extensively studied in scientific literature such as (Aggrey, 2002), (Ahmadi & Mottaghitalab, 2007) and (Hasan Elerolu & Duman, 2014). A growth curve in this context is a static curve fitted to old broiler weight data. It has a fixed structure with few parameters

[*]Corresponding author
Email addresses: `sjo@skov.dk` (Simon V. Johansen), `dimon@es.aau.dk` (Jan D. Bendtsen), `mrj@skov.dk` (Martin R.-Jensen), `jmo@skov.dk` (Jesper Mogensen)

that offers biologically intuitive interpretations. Common models include the Richerds Model and Gompertz-Laird Model that are described by 4 and 3 parameters respectively, where the parameters have biologically intuitive interpretations – such as time and size of maximum growth rate (Aggrey, 2002). In (Lopes et al., 2008), the relation between broiler house environment and production performance was investigated using a neural network. However, it is not clear how to extract temporal performance (i.e. growth prediction) from this type of model.

Dynamic broiler growth models are an extension of growth curves and has primarily been developed for control synthesis in scientific literature. In (Aerts et al., 2003) a time-variant online parameter estimation of a dynamic model was successfully applied to predict future weight up to 7 days without a priori information. The model uses feed intake as input and weight as output and was used for model predictive control (MPC) in (Cangar et al., 2007). This control algorithm was tested in a laboratory setting with a stocking density of 5.3 birds/m^2 and 20 birds/m^2, the later to emulate farm scale density. The mean relative weight control error was 2.7% and 7.3% for the low and high-density experiments respectively – suggesting that farm scale broiler production is harder to both predict and control.

A similar result was obtained in (Demmers et al., 2010), where a small differential recurrent neural network was used to model the feed quantity and control the broiler weight using nonlinear MPC. In (Stacey et al., 2004) a dynamic broiler weight model was developed and used information about feed uptake and composition of two feed types with known nutritional value. The model was successfully used to control broiler weight, it was tested on farm scale with 30,000-40,000 broilers per house and achieved results comparable to that of a stockman. However, these studies did not take environmental conditions into account.

Input variable selection (IVS) is a critical step in data driven modeling, where appropriate input variables are selected from available data. The IVS discipline is diverse and has many perspectives and applications, for an overview of time series IVS see May et al. (2011), and for feature selection see Brown et al. (2012). Given our application, we narrow our scope to model free filter type IVS algorithms – also known as filter type IVS.

One class of statistical IVS is based on mutual information(MI). By definition, mutual information $I(X;Y)$ is the reduction in uncertainty with respect to the random variable Y due to the observation of the random variable X May et al. (2008b). MI is often calculated by estimating the underlying marginal and joint probability density functions (PDFs) through kernel density estimation (KDE). Such an IVS algorithm was formulated in Sharma (2000), and modified in May et al. (2008b) among others. It uses successive regression to remove selected information from the input candidates and target output and estimates partial mutual information (PMI), while relying on computationally heavy bootstrapping to determine a stopping criteria. Alternatively, the Copula entropy can be used to estimate MIChen et al. (2014).

In this paper we use neural network class models for forecasting and mutual information based IVS to select inputs and their respective lag. More specifically, we use dynamic neural networks, which have been successfully applied to model complex biological processes. Recent applications include algae growth prediction in a laboratory setting (Wang et al., 2015), prediction of bioethanol production in a bioreactor (Grahovac et al., 2016), bioreactor prediction (Nair et al., 2016), yeast fermentation modeling in a bioreactor (Nasimi & Irani, 2014) and state estimation in a continuous bio reactor (Hernandez et al., 2013). Likewise, mutual information based IVS has been applied extensively to environmental modeling, such as IVS for prediction of rainfall Sharma (2000), salinity Bowden et al. (2005), Water Quality May et al. (2008a), storm water runoff He et al. (2011), flood forecasting Chen et al. (2014), and rainfall-runoff Li et al. (2015).

In the present effort, we present a data-driven dynamic neural network broiler batch forecasting model with IVS. In particular, we show that dynamic interactions between environmental conditions and broiler behavior in ad libitum feed broilers are present and can be captured by such a model. We do this in a data driven framework on real farm scale production data from a state-of-the-art broiler house. Preliminary results were presented in Johansen et al. (2017).

The remainder of the article is structured as follows. In Section 2 the model, training and validation method is described, and in Section 3 the MI based IVS is described. In Section 4 the experimental verification setup is described, consisting of a single forecasting trial in Section 5, and an analysis of all trials in Section 6. We provide concluding remarks in Section 7.

2. Model, Training and Validation Method

2.1. Model

Since the multilayer perceptron (MLP) model and its variations have been extensively researched in scientific literature we will only give a brief introduction and describe how we apply it to our specific problem. For a thorough introduction to the subject we redirect the reader to (Du & Swamy, 2014) and (Haykin, 1994).

The particular type of Dynamic Neural Network (DNN) model we use in this work can be classified as a discrete-time nonlinear ARX model of the form

$$\hat{y}[k+1 \mid \mathcal{W}] = \mathcal{N}\big(\hat{Y}[k], U[k] \mid \mathcal{W}\big) \tag{1}$$

with

$$U[k] = \Big[u[k-\bar{n}_1]^T \ \cdots \ u[k-\bar{n}_i]^T\Big]^T \text{ and} \tag{2a}$$

$$\hat{Y}[k] = \Big[\hat{y}[k-\bar{m}_1 \mid \mathcal{W}]^T \ \cdots \ \hat{y}[k-\bar{m}_j \mid \mathcal{W}]^T\Big]^T, \tag{2b}$$

where $\mathcal{N}$ is a MLP model, $U[k]$ is delayed values of the input vector $u[k] \in \mathbb{R}^{N_u}$ corresponding to the $N_{\bar{n}}$ elements of $\bar{n} = \{\bar{n}_1, \cdots, \bar{n}_{N_{\bar{n}}}\}$, $\hat{Y}[k]$ is delayed values of the previous output vector $\hat{y}[k] \in \mathbb{R}^{N_y}$ corresponding to the $N_{\bar{m}}$ elements of $\bar{m} = \{\bar{m}_1, \cdots, \bar{m}_{N_{\bar{m}}}\}$ and $\mathcal{W} \in \mathbb{R}^{N_{\mathcal{W}}}$ is an abstract representation of all the $N_{\mathcal{W}}$ model weights. This particular structure has been adopted to accommodate potentially long propagation delays, while still aiming to keep the number of weights relatively low.

The DNN model $\mathcal{N}$ is selected with one hidden layer with hyperbolic tangent activation function in the hidden layer and linear activation function in the output layer. It can be shown that $\mathcal{N}$ is an universal function approximator, meaning that it has the capacity for approximating any system to any accuracy (Du & Swamy, 2014, chap. 4.2). In matrix-vector representation, (1) is written explicitly as

$$\hat{y}[k+1 \mid \mathcal{W}] = W^{\mathrm{o}} \tanh(\mathcal{X}) + \theta^{\mathrm{o}} \text{ with} \tag{3}$$
$$\mathcal{X} = \sum_{i=1}^{N_{\bar{m}}} W^{\mathrm{h}}_{y,i}\hat{y}[k-\bar{m}_i \mid \mathcal{W}] + \sum_{j=1}^{N_{\bar{n}}} W^{\mathrm{h}}_{u,j} u[k-\bar{n}_j] + \theta^{\mathrm{h}},$$

where N_h is the number of neurons in the hidden layer, $\mathcal{X} \in \mathbb{R}^{N_h}$, $W^{\mathrm{o}} \in \mathbb{R}^{N_y \times N_h}$ is the output weights, $W^{\mathrm{h}}_{y,i} \in \mathbb{R}^{N_h \times N_y}$ is the delayed output weights, $W^{\mathrm{h}}_{u,j} \in \mathbb{R}^{N_h \times N_u}$ is the delayed input weights, $\theta^{\mathrm{h}} \in \mathbb{R}^{N_h}$ is the hidden layer bias and $\theta^{\mathrm{o}} \in \mathbb{R}^{N_y}$ is the output bias. A visual representation of (3) is depicted on Figure 2.

Input and output specific delays selected by the IVS are accommodated by modifying the structure of $W^{\mathrm{h}}_{u,j}$ and $W^{\mathrm{h}}_{y,i}$. For example, if the inputs indexed 1 and 3 are selected with delay of $j = 2$, $N_u = 4$ inputs, $N_h = 3$ hidden neurons, then $W^{\mathrm{h}}_{u,j}$ equals

$$W^{\mathrm{h}}_{u,2} = \begin{bmatrix} \mathcal{W}_1 & 0 & \mathcal{W}_2 & 0 \\ \mathcal{W}_3 & 0 & \mathcal{W}_4 & 0 \\ \mathcal{W}_5 & 0 & \mathcal{W}_6 & 0 \end{bmatrix}. \tag{4}$$

2.2. Forecasting Trial Setup

Each forecasting trial aims at forecasting future outputs throughout a batch, only based on previous batches. We partition the available batches into four categories, namely training, testing, prediction, and evaluation as depicted on Figure 1. In ascending order from the latest batch is the evaluation batch, prediction batch, and the remaining batches are training batches. The training batches are used to find the network weights $\mathcal{W}$ through training. The latest training batch is used in an early stopping setting to avoid over-fitting by selecting the model that generalizes best to this batch. In order to forecast future outputs on the evaluation batch, unknown future input values are required of which we use the inputs from the prediction batch as substitute.

2.3. Model Training

The dynamic neural network is trained by minimizing the cost function

$$E(\mathcal{W}) = \overbrace{\sum_{b=1}^{N_B}\sum_{p=1}^{N_P}\sum_{k=P_p}^{N_{s,b}} \frac{\|y[k \mid b] - \hat{y}[k \mid P_p, b, \mathcal{W}]\|_2^2}{N_r}}^{E_r} + \bar{\alpha}\overbrace{\|\mathcal{W}\|_2^2}^{E_{\mathcal{W}}} \text{ with}$$

$$N_r = \sum_{b=1}^{N_B}\sum_{p=1}^{N_P}\sum_{k=P_p}^{N_{s,b}} N_y = N_B N_P N_y (N_{s,b}+1) - N_B N_y \sum_{p=1}^{N_P} P_p, \quad (5)$$

where N_B is the number of training batches, $N_{s,b}$ is the number of samples in batch b and N_r is the total number of residuals in E_r. $\|\cdot\|_2$ is the standard Euclidean 2-norm. Both the inputs and outputs are normalized to a mean of 0 and standard deviation of 1 during training. The network weights $\mathcal{W}$ are initialized using the Nguyen-Widrow algorithm as explained in (Nguyen & Widrow, 1990).

Each batch is divided into N_P Sub-Batches with initial starting times denoted by $P = \{P_1, \cdots, P_{N_P}\}$, which we find speeds up training and decreases the risk of converging to a poor local minima. Consequently, for each batch we get

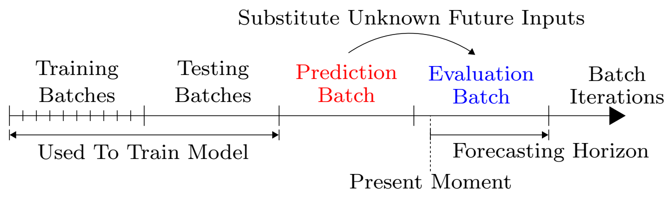


Figure 1: Visual representation of how batches are selected for a forecasting trial.

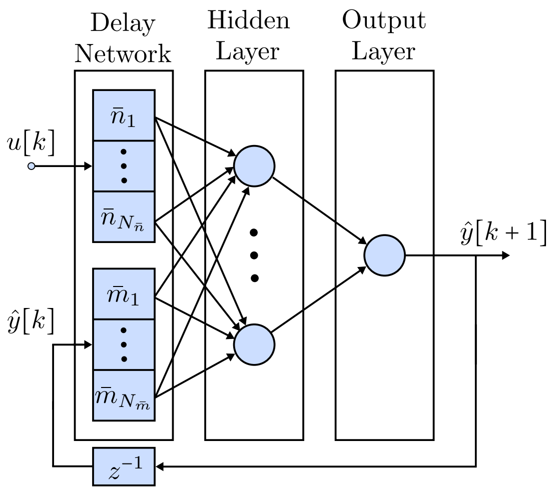


Figure 2: Visual representation of (3) with one input and one output for simplicity. Neurons are represented with blue circles, which contain an activation function and bias each. Note that all incoming signals to a neuron are multiplied by a weight. The operator z^{-a} produces a delay of a samples.

N_P sets of trajectories with different initial conditions to minimize, as illustrated on Figure 3, with equal weight.

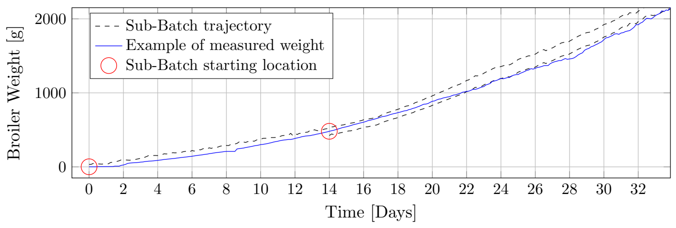


Figure 3: Example of the Sub-Batch trajectories with Sub-Batch starting time at $P = \{0, 14\}$ days.

The target output at sample k given batch b is denoted $y[k \mid b]$, while the predicted network output at sample k, initialized at Sub-Batch starting time P_p, using future input values from batch b with model weights $\mathcal{W}$ is denoted $\hat{y}[k \mid P_p, b, \mathcal{W}]$. In this context, initialization refers to the initial values of (2).

The last term of the cost function is a scalar regularization term punishing the size of the $N_{\mathcal{W}}$ system weights $\mathcal{W}$, where $\bar{\alpha}$ is the regularization weight. The regularization weight $\bar{\alpha}$ is determined iteratively through Bayesian Regulation as described in (Burden & Winkler, 2008). We normalize the regularization weight to make the cost functions comparable between trainings, and is calculated according to $\bar{\alpha} = \alpha/\beta$ with:

$$\alpha = \frac{\gamma}{2E_{\mathcal{W}}} \quad \beta = \frac{N_r - \gamma}{2E_r} \quad \gamma = N_{\mathcal{W}} - \alpha\,\mathrm{trace}\big(G^{-1}\big) \tag{6}$$

Where $E_{\mathcal{W}}$ and E_r originate from (5) and γ, $0 \leq \gamma \leq N_{\mathcal{W}}$, is a measure of how many of the $N_{\mathcal{W}}$ parameters are used. Lastly, $G \in \mathbb{R}^{N_{\mathcal{W}} \times N_{\mathcal{W}}}$ is the Hessian matrix of the joint cost $\beta E_r + \alpha E_{\mathcal{W}}$ as seen in to (7a), where I is the identity matrix of appropriate dimensions. It is partially approximated through the Jacobian J_r of E_r in (7b), similar to the Levenberg Marquard algorithm. The scalar constants α and β are iteratively updated between training iterations, with initial conditions $\alpha = 0$ and $\beta = 1$.

$$G = \frac{d^2(\beta E_r + \alpha E_{\mathcal{W}})}{d\mathcal{W}\, d\mathcal{W}^T} = \frac{d^2(\beta E_r)}{d\mathcal{W}\, d\mathcal{W}^T} + 2\alpha I \tag{7a}$$

$$\approx J_r^T J_r + 2\alpha I \text{ with } J_r = \beta \frac{dE_r}{d\mathcal{W}^T} \tag{7b}$$

The cost function (5) is minimized using the Levenberg-Marquardt (LM) optimization algorithm by means of the Ceres Solver library (Sameer Agarwal & Others, 2015). Ceres implements an exact step LM variant based on Madsen et al. (2004). It is an algorithm that efficiently solves large scale least square problems on the form

$$\mathcal{W}^* = \underset{\mathcal{W}}{\arg\min}\left[\frac{1}{2}E(\mathcal{W}) = \frac{1}{2}\|F(\mathcal{W})\|_2^2\right], \tag{8}$$

where $E(\mathcal{W}) \in \mathbb{R}$ is the cost function to be minimized, $F(\mathcal{W}) \in \mathbb{R}^{N_r + N_{\mathcal{W}}}$ is the residuals of $E(\mathcal{W})$, $\mathcal{W} \in \mathbb{R}^{N_{\mathcal{W}}}$ is the weights and $\mathcal{W}^* \in \mathbb{R}^{N_{\mathcal{W}}}$ is the optimal weights found by the algorithm. Note that the problem is not guaranteed to be convex, and consequently $\mathcal{W}^*$ is only guaranteed to be a local minimum. In the following the dependence on $\mathcal{W}$ is implicit for the residuals $F(\mathcal{W})$ and its Jacobian $J_F(\mathcal{W}) = \frac{dF(\mathcal{W})}{d\mathcal{W}^T} \in \mathbb{R}^{(N_r + N_{\mathcal{W}}) \times N_{\mathcal{W}}}$. LM is an iterative procedure and requires an initial weight guess. The weights are updated according to $\mathcal{W} \leftarrow \mathcal{W} + \Delta\mathcal{W}$ by solving for the weight change $\Delta\mathcal{W} \in \mathbb{R}^{N_{\mathcal{W}}}$ in

$$\big(J_F^T J_F + \mu I\big)\Delta\mathcal{W} = -J_F^T F, \tag{9}$$

where $0 \leq \mu \in \mathbb{R}$ is the damping parameter, using the Cholesky Factorization. The LM algorithm is a trust region method and controls the step size with the damping parameter μ, of which a larger value will reduce the step size. This is controlled by the ratio between the actual and predicted cost change given by

$$\rho = \frac{E(\mathcal{W}) - E(\mathcal{W} + \Delta\mathcal{W})}{\hat{E}(0) - \hat{E}(\Delta\mathcal{W})}, \tag{10}$$

where 0 is the zero-vector, and $\hat{E}(\Delta\mathcal{W}) \simeq E(\mathcal{W} + \Delta\mathcal{W})$ is an approximated model of $E(\mathcal{W} + \Delta\mathcal{W})$ in $\mathcal{W}$ given by

$$\hat{E}(\Delta\mathcal{W}) = E(\mathcal{W}) + \Delta\mathcal{W}^T J_F^T F + \Delta\mathcal{W}^T J_F^T J_F \Delta\mathcal{W}. \tag{11}$$

If the cost function is successfully decreased by $\Delta\mathcal{W}$, then the step is accepted and the damping parameter μ is updated according to

$$\mu \leftarrow \mu \cdot \max\Big\{\tfrac{1}{3},\, 1 - (2\rho - 1)^3\Big\}, \tag{12}$$

otherwise the step is rejected and the damping parameter is increased according to $\mu \leftarrow \mu \cdot 2^{\eta}$ with $\eta \in \mathbb{Z}^+$ being the number of consecutive rejected updates. Note that if $\frac{1}{2} < \rho$ then μ is decreased, which increases the trust region, as it indicates that J_F is a good approximation. This process is repeated until $\mathcal{W}$ converges, or 1000 training iterations have elapsed.

2.4. Ensemble Model

We generate an ensemble model with 64 sub-models on the same training data, and each sub-model is trained with different initial weights. We use the ensemble model mean and standard deviation to represent the "true" model output, as it is expected to be robust against initialization effects due to the high number of models. To diminish the effect of "outlier" models, we apply Grubbs' outlier test to the separate output samples to remove deviating model outputs at a 5% significance level.

3. Mutual Information-based Input Selection

3.1. Information Theory

The information entropy of the random variable X is given by

$$H(X) = -\int_X p(x) \log p(x) dx, \tag{13}$$

and is a measure of how much information is contained by X. Note that this measure extends to multivariate inputs as well, in which case it is a measure of the collective information of all variables. The mutual information $I(X;Y)$ is the reduction in uncertainty with respect to the random variable Y due to the observation of the random variable X, and is given by

$$I(X;Y) = -\int_Y \int_X p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dx\, dy. \tag{14}$$

Alternatively, it can be regarded as the amount of information X provides about Y, and vice versa. It can be shown to be a function of the linear correlation coefficient R_{XY} in case of linear and Gaussian distributed inputs given by

$$I(X;Y) = -\frac{1}{2} \log\left(1 - R_{XY}^2\right). \tag{15}$$

However, mutual information is more robust due to its insensitivity to noise and data transformations (i.e. nonlinearity) May et al. (2011). Through resubstitution (13) and (15) can be estimated by

$$\hat{H}(X) = -\frac{1}{n} \sum_{k=1}^{n} \log \hat{f}_X(X[k]) \text{ and} \tag{16a}$$

$$\hat{I}(X;Y) = \frac{1}{n} \sum_{k=1}^{n} \log \frac{\hat{f}_{X,Y}(X[k], Y[k])}{\hat{f}_X(X[k]) \hat{f}_Y(Y[k])}, \tag{16b}$$

where $\hat{f}_X(\cdot)$ and $\hat{f}_Y(\cdot)$ are the estimated marginal density function of X and Y, respectively, and $\hat{f}_{X,Y}(\cdot,\cdot)$ is the estimated joint density function of X and Y Beirlant et al. (1997). Intuitively, the nominator of (16b) equals the denominator if X and Y are uncorrelated, as $\hat{f}_X(X[k])\hat{f}_Y(Y[k]) = \hat{f}_{X,Y}(X[k], Y[k])$, and consequently does not add any new information, as $\log 1 = 0$. With

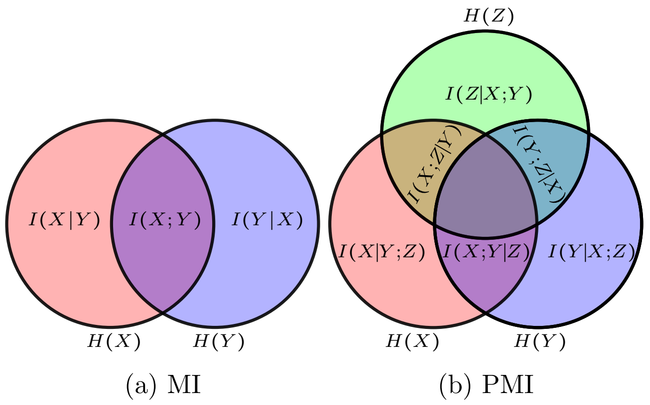


Figure 4: Venn diagram representation of MI and PMI.

$X = \{X_1, \cdots, X_i\}$, $Y = \{Y_1, \cdots, Y_j\}$ and $x \in \mathbb{R}^i$ the notation for multivariate mutual information and i-variate joint density estimation is

$$I(X;Y) \triangleq I(X_1, \cdots, X_i; Y_1, \cdots, Y_j) \text{ and}$$
$$\hat{f}_X(x) \triangleq \hat{f}_{X_1,\cdots,X_i}(x_1, \cdots, x_i).$$

The partial mutual information (PMI) is a measure of the information between two variables X and Y, excluding the information from a third variable Z. It can be regarded as a measure of how much mutual information is not explained by Z, which is useful for determining if an input variable provides additional information about an output variable. In terms of mutual information and information entropy it is given by:

$$\begin{aligned} I(X;Y \mid Z) &= I(X;Y,Z) - I(X;Z) \\ &= H(X,Z) + H(Y,Z) - H(X,Y,Z) - H(Z) \end{aligned} \tag{17}$$

The relationship between information entropy, mutual information and partial mutual information is visualized on Figure 4. Applying (20) on a dataset is illustrated on Figure 5.

3.2. Kernel Density Estimation (KDE)

To calculate the joint PDFs in (16), we employ kernel density estimation. The d-variate Gaussian kernel density estimator (KDE) can be formulated by

$$\hat{f}_X(x) = \frac{1}{n}\sum_{k=1}^{n} K_{\bar{H}}(x - X[k]) \text{ with} \tag{18a}$$

$$K_{\bar{H}}(y) = \frac{K\left(\bar{H}^{-1}y\right)}{\det \bar{H}} \text{ and} \tag{18b}$$

$$K(z) = \frac{1}{\sqrt{(2\pi)^d}} \exp\left(-\frac{1}{2}z^T z\right). \tag{18c}$$

Where (18a) and (18b) are the multivariate parzen window with $x, y, z \in \mathbb{R}^d$, $\bar{H} \in \mathbb{R}^{d\times d}$ being the kernel bandwidth or smoothing matrix, n is the number of samples in the known data $X \in \mathbb{R}^{d\times n}$, and $X[k] \in \mathbb{R}^d$ denotes the k'th sample of X. For our purposes we use the generalized Scott's rule (Wolfgang Hrdle & Werwatz, 2004, pp. 73) given by

$$\bar{H} = \Sigma^{1/2}\,\bar{h}_S \text{ with } \bar{h}_S = n^{-1/(d+4)}, \tag{19}$$

where Σ is the covariance matrix of the known data $X \in \mathbb{R}^{d\times n}$ and $\bar{h}_S \in \mathbb{R}$. Furthermore, (18c) is the kernel function, where we use the multivariate Gaussian density function. The full expression for (18) equals

$$\hat{f}_X(x) = \frac{\sum_{k=1}^{n} \exp\left(-\frac{1}{2}(x - X[k])^T \left(\bar{h}_S^2 \Sigma\right)^{-1} (x - X[k])\right)}{n\sqrt{(2\pi)^d \det\left(\bar{h}_S^2\Sigma\right)}}. \tag{20}$$

Li et al. (2015) investigates different kernel bandwidth selection algorithms for the IVS algorithms in May et al. (2008b) under non-normal conditions, as May et al. (2008b) assumes normality through the Gaussian reference rule. Using alternative bandwidth selection algorithms lead to increased selection accuracy in IVS for rainfall and runoff prediction. However, we have not found it necessary to investigate alternative bandwidth selection algorithms for our purposes – suggesting that the data might be close to Gaussian.

3.3. Input Variable Selection (IVS)

In the context of IVS for the model described in Section 2, the task of progressively selecting input variables and their respective lags can be formulated as follows.

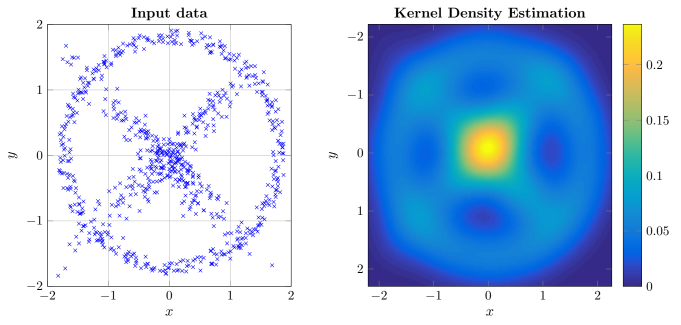


Figure 5: The estimated probability density distribution $\hat{f}_{X,Y}(x, y)$ given by (20) of two correlated variables x and y. The linear correlation coefficient is -0.005 in this case, suggesting that x and y are very close to being uncorrelated. The KDE represents the underlying correlated distribution significantly better in this case. The Mutual Information $\hat{I}(x; y) = 0.204$ given by (16b), suggesting that information of y can be obtained from x, as the 95% critical value for Gaussian noise with $n = 400$ samples equals 0.0567 May et al. (2008a).

Let X denote the set of lagged input candidates, $S = \{S_1, \cdots, S_{m-1}\}$ the $(m-1)$ previously selected lagged inputs with $S \subseteq X$ and Y being the target output. The m'th lagged input candidate with *most new information* is selected according to

$$S_m = \underset{C\in X\setminus S}{\arg\max}\, I(C, S; Y) = \underset{C\in X\setminus S}{\arg\max}\, I(C; Y \mid S). \tag{21}$$

This is repeated until S_m provides *no significant* new information about Y, and S_m is not added to S. Originally, the 95 percentile critical bootstrap value was used to estimate a termination threshold, but since it is computationally heavy May et al. (2008b) compared alternatives. Termination criterias based on the Akaike Information Criterion, modified Hampel outlier test and PMI lookup table based on Monte Carlo simulations of normally distributed noise for different sample sizes were compared. We have good experience with using the 95-percentile tabulated PMI as depicted on Figure 6, but it should be noted that May et al. (2008b) found that this method gives biased values for non-Gaussian distributed inputs. Realizing that this table is approximately log-log linear in the number of samples n, the following approximation is found (See

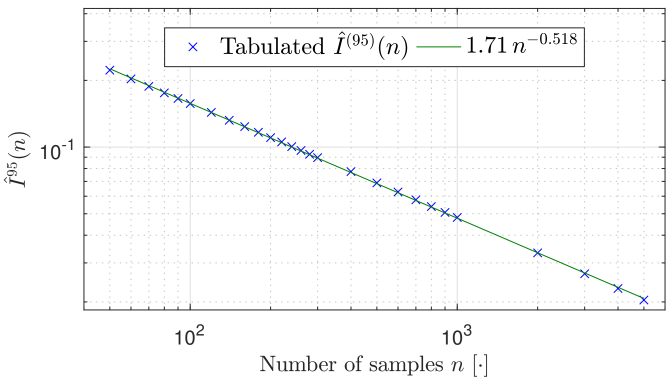


Figure 6: Tabulated PMI and an approximation May et al. (2008b).

Figure 6):

$$\hat{I}^{(95)}(n) = 1.7134 \cdot n^{-0.518} \tag{22}$$

Calculation of (21) requires estimation of an $(m+1)$ dimensional probability density function using (20), and is therefore increasingly dependent on the data quality for larger m – both in terms of number of samples and increasing data sparsity in higher dimensional spaces. This phenomenon is called *the curse of dimensionality* and is illustrated in Table 1. Hence, higher order MI is not preferred to be calculated, as we cannot make any guarantees regarding data quality for broiler production.

3.4. Regression based PMI

This method is taken from May et al. (2008b), and use the KDE based conditional expectation known as the Nadaraya-Watson estimator for regression. It is the regression of Y given the observed X, denoted by $E[Y|X]$, where the j'th sample is calculated by

$$E[Y \mid X = X[j]] = \frac{1}{n}\frac{\sum_{k=1}^{n} Y[k] K_{\bar{H}}(X[j]-X[k])}{\sum_{k=1}^{n} K_{\bar{H}}(X[j]-X[k])}. \tag{23}$$

For simplicity, the removal of known "information" through regression of the elements of B from the variable A is written as

$$M(A|B) = A - E[A|B].$$

The key idea behind this method is to estimate the partial mutual information by removing the known mutual information from the selected inputs through successive regression. Let $(\cdot)^i$ denote the i'th successive regression of $(\cdot)$, then the m'th iteration of the selected inputs $S = \{S_1^1, \cdots, S_{m-1}^{m-1}\}$ of the dataset X equals

Dimension	Sample Size
1	4
2	19
3	67
4	223
5	768
6	2790
7	10700
8	43700
9	180700
10	842000

Table 1: Sample size required to maintain a constant standard error in a general regression neural network May et al. (2011).

$$X^m = \begin{cases} M\left(X^{m-1} | S_{m-1}^{m-1}\right) & m > 1 \\ X & m = 1 \end{cases}. \tag{24}$$

Notice that X^m depends on up to two subsequent regressions through X^{m-1} and S_{m-1}^{m-1}. To demonstrate, X^4 is obtained from the following successive regressions:

$$\begin{array}{llll} X^1 = X & S_3^1 = S_3 & S_2^1 = S_2 & S_1^1 = S_1 \\ X^2 = M\left(X^1|S_1^1\right) & S_3^2 = M\left(S_3^1|S_1^1\right) & S_2^2 = M\left(S_2|S_1^1\right) & \\ X^3 = M\left(X^2|S_2^2\right) & S_3^3 = M\left(S_3^2|S_2^2\right) & & \\ X^4 = M\left(X^3|S_3^3\right) & & & \end{array}$$

Note that the first row contains the original values, and each subsequent row is a regression based only on the previous row.

Using (24), the IVS problem formulated in (21) can be approximated by

$$S_m = \underset{C \in X \setminus S}{\arg\max} \left[I(C; Y \mid S) \approx I\left(C^{m-1}; Y^{m-1}\right) \right], \tag{25}$$

which has the benefit of only requiring 2-dimensional density function estimates compared to previous $(m-1)$ from

S. When implementing this algorithm, the values of $(\cdot)^m$ are calculated from $(\cdot)^{m-1}$ after S_m has been selected to avoid unnecessary recalculation.

4. Data Selection for Forecasting

This work is based on data gathered from a ≈20 year old state-of-the-art broiler house located in the northern Denmark – not the same as used in Johansen et al. (2017). We use data from 29 batches, collected over a period of 3 years and 4 months. Each batch contains roughly 40,000 broilers.

The potential inputs to the model consist of the available environmental variables for this broiler house: The measured inside temperature, outside temperature and humidity, light intensity reference, ventilation demand, and heating demand. Note that measured CO_2 was not included, as it was only present in 8/28 batches. The model outputs consist of the available broiler behavior indicators: the measured weight along with feed and water consumption per bird.

We intentionally distinguish between reference, demand and measured variables. Reference variables are independent, demand variables are determined by a deterministic entity like a controller outside the model, and measured variables are dependent on the model process.

To elaborate, humidity and temperature measurements are measured during closed loop operation and are controlled through both the ventilation and heating demands. For this reason, these variables are not independent, which has been known to cause problems in closed loop identification (Rajamani Doraiswami & Stevenson, 2014). We do not consider this an issue as all closed loop controlled variables are considered inputs to the model and therefore no deterministic output feedback is present. Keep in mind that reference inputs for controlled variables is determined by the broiler farmer, whom we consider a non-deterministic entity, making it an open loop identification problem with restricted input space. We give an example of this in Section 5.1.1.

We use a sample interval of 3 hours with the first sample at midnight (00:00) on the first day regardless of the actual start time of the batch due to technical limitations, which permits the time of day to be partially captured. The mean value of the data within each sample period is used.

4.1. Evaluation Procedure

From the 29 available batches 18 forecasting trials have been carried out. Each trial aims at predicting the broiler weight of the unfinished evaluation batch by using the preceding batches as described in Section 2.2. In addition to this, we restrict each trial to at most contain $N_{B,\max} = 10$ training batches to introduce a forgetting factor, as the broiler house changes over time. As all inputs and outputs are not guaranteed to be present in all the batches, we select the batches to maximize

$$N_u \cdot N_y \cdot \min\{N_{B,\max}, N_B\}, \tag{26}$$

which selects up to $N_{B,\max}$ batches with the highest number of inputs N_u and outputs N_y – the most recent batches take priority to use the most up to date data.

The IVS algorithm described in Section 3 is applied to the training and testing batches to select relevant inputs, input lags and output lags. As we cannot guarantee the data-quality in broiler production, we deliberately limit the amount of selected variables and number of lags. With 10 training batches, each roughly contains 250 samples, Table 1 suggests that no more than 5 inputs should be selected. We restrict the search space to 3 input classes (e.g. measured temperature) per output where each input class can have up to 2 lags. As the IVS is independently carried out for each of the outputs, a maximum of 18 lagged inputs can be selected. Furthermore, we consider up to 16 sample lags for the inputs and outputs for the model – equivalent of 2 days.

We evaluate the 18 forecasting trials in Section 6, where we provide an overview of the IVS and identify trends. To evaluate the ensemble models' ability to forecast across all 18 trials, we perform three types of forecasts on each trial: 1) forecast on prediction batch with known future inputs, 2) forecast on evaluation batch with known future inputs, and 3) forecast on evaluation batch with substitute future inputs from the prediction batch. We calculate the *trial 1-Norm forecasting error*[1] of each output according to

$$\sum_{i=1}^{N_t-1} \sum_{k=i+1}^{N_t} \frac{|e[k \mid i]|}{\frac{1}{2}N_t\,(N_t-1)}, \tag{27}$$

where $e[k \mid i]$ is the residual error at sample k forecasted from sample i, N_t is the number of samples in the forecasting trial, $|\cdot|$ is the absolute value, and the factor $\frac{1}{2}N_t\,(N_t-1)$ is a normalization constant. To give an intuition about the *day-specific 1-Norm forecasting error* of each output from day $\kappa \in \{2,\ 7,\ 14,\ 21,\ 28\}$ to the end of the batch, we use

$$\sum_{k=\bar{\kappa}+1}^{N_t} \frac{|e[k \mid \bar{\kappa}]|}{N_t - \bar{\kappa}} \text{ with } \bar{\kappa} \in \kappa. \tag{28}$$

Based on this we investigate the performance impact of using substitute future inputs in 3), by comparing with 1) and 2).

4.2. Model Configuration

For this work we have good experience with $N_h = 7$ hidden neurons in the hidden layer of the model. On average this results in 230 free model parameters across all trials. According to the Gaussian Regularization, this results in an average of $\gamma = 21$ (8.5% of the model weights) unused weights across all trials. We find this reasonable, as the model has room to handle additional complexity. We have good experience with $N_P = 5$ sub-batch training start times at $P \in$ day $\{2, 7, 14, 21, 28\}$ in (5).

[1]We use the 1-Norm as performance metric, as it is a measure of the mean absolute error and therefore offers a more intuitive interpretation compared to the 2-norm used for training.

5. Experimental Forecasting: Trial Case Study

In this section a forecasting case study of the most recent forecasting trial is presented. We present and analyze the trial input- and output data, and give an example of a forecast from day 14 to the batch end. To demonstrate the model's ability to forecast broiler weight throughout a batch, we forecast the weight from all preceding samples to day 7, 14, 21, 28 and batch end on the same batch – denoted *weight on day forecasting.* We show the ensemble standard deviation of the ensemble model to give an indication of the underlying models. In both cases we show the forecast on the evaluation batch with known future outputs, and on the evaluation batch with substitute future inputs from the prediction batch as described in Section 2.2.

5.1. Trial Data Analysis

To understand the trial results we give our interpretation of the input and output variables, which are depicted on Figure 7. We emphasize the difference between what the model "knows" through training data, and the "unknown" validation and prediction data, which can help explain if the model has difficulties at generalizing. We also point out known sources of error related to broiler batch production that can influence the model's forecasting ability. The IVS results for this trial are depicted on Figure 2 with rounded rectangles, which we will not comment on in the following.

5.1.1. Input variables

First we give an example of the aforementioned correlated input variables, which are most clearly illustrated on the prediction batch on Figure 7. Note that the ventilation demand is negatively correlated with the temperature and humidity. The intuition is that the broilers introduce both heat and water into the air in the broiler house, increasing the temperature and humidity, which are removed through ventilation.

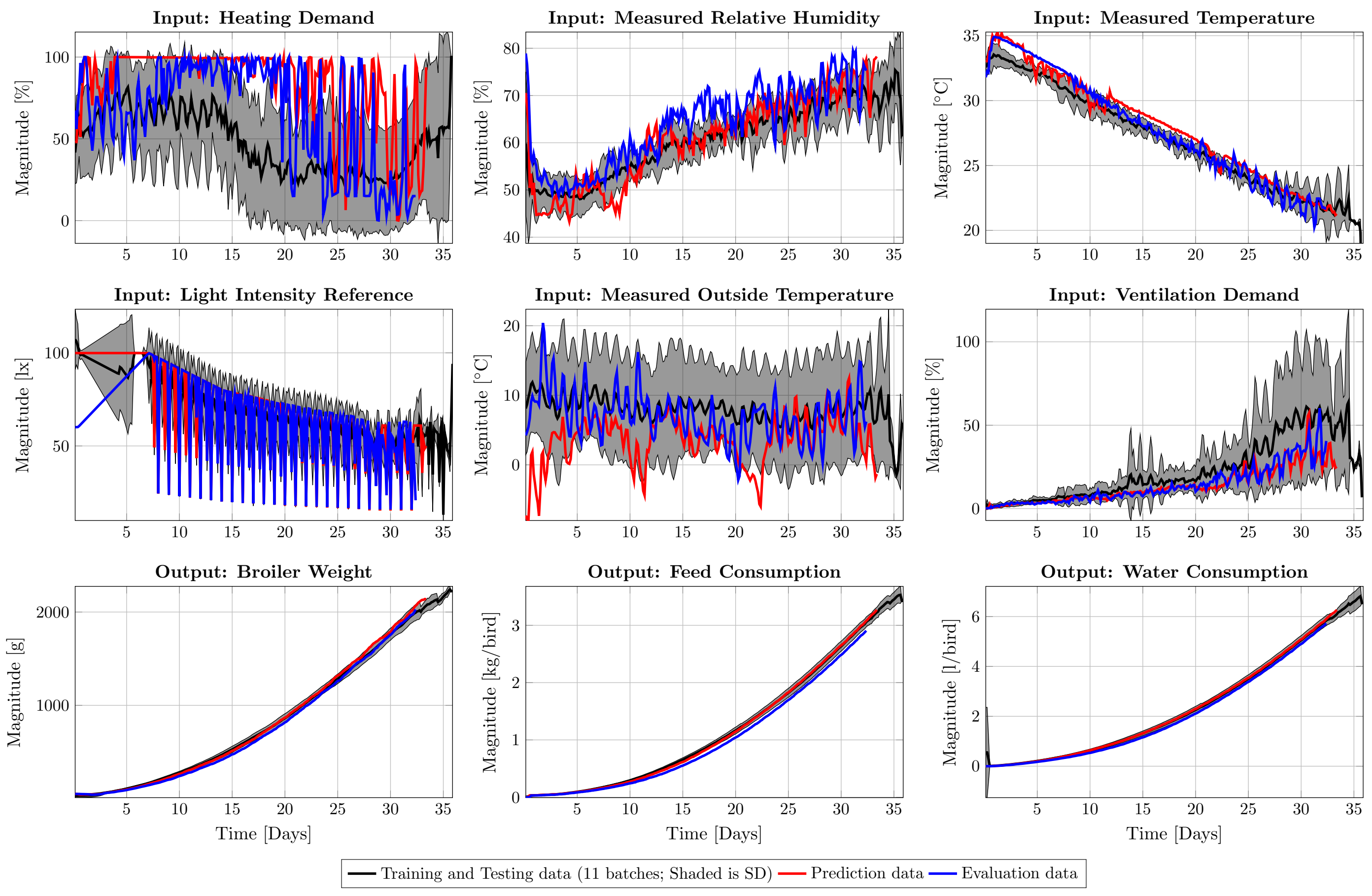


Figure 7: The inputs and output data from the batches used in the most recent trial. The black shaded area denotes one standard deviation from the mean value.

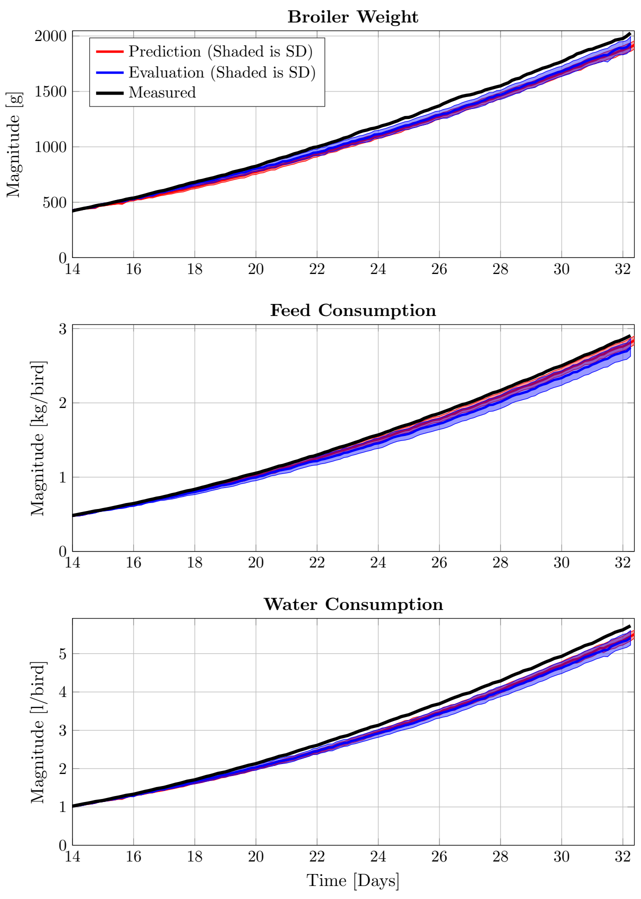
A rule of thumb by the hatchery supplying the day old broilers is to keep the temperature at ≈30 °C in the beginning of the batch and decrement it by 1 °C every 3 days until 20 °C. The beginning temperature appears to be ≈4 °C offset in this case. We note a low degree of variation in temperature across all batches, compared to e.g. humidity. Overall the measured temperature of the prediction- and evaluation batch appear to have similar behavior as the training and testing data. However, the prediction batch is on average ≈1 °C higher onwards of day ≈11.

The heating demand for the prediction and evaluation batch have significantly higher fluctuations onwards of day 15, which are caused by an below average outside temperature. Lastly, we note that the light intensity reference data are quite spiky, which is caused by a light program emulating the day and night cycle.

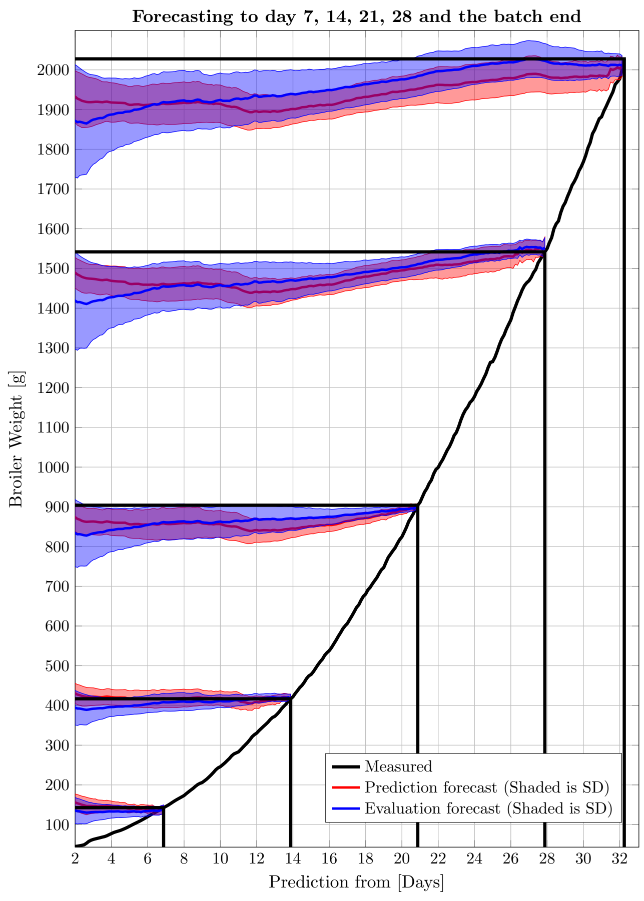
5.1.2. Output variables

Broiler weight is highly positively correlated with cumulative feed and water consumption, as the two are necessary for both survival and growth of the broilers. Bird weight has been reported by customers to be highly predictable from feed and water consumption – supported by the fact that the three have similar variations and behavior. Note that it is unknown if and when the farmer has changed feed type for any of the batches.

We note that the feed consumption is distinctly lower for the evaluation batch compared to the training, testing and prediction data, but the measured broiler weight and water consumption are within one standard deviation of the training and testing data. For this reason, we expect



(a) Example of forecasting from day 14 to the batch end.



(b) Weight on day forecasting to day 7, 14, 21, 28 and the batch end.

Figure 8: The solid blue evaluation forecasts are generated with models initialized with evaluation batch data and future inputs from the evaluation batch. The solid red prediction forecasts are initialized with evaluation batch data, but with substitute future inputs from the prediction batch. To be specific, the blue traces use future input data from the current batch, and the red traces use substitute future input data from the previous batch as described in Section 2.2. The blue and red bold lines are the ensemble model mean, while the shaded areas denote the ensemble standard deviation. Figure 8a depicts a single forecast from day 14 to the batch end. Figure 8b aims at showing the weight forecast of Figure 8a for all start samples on specific days marked with vertical solid black lines. The horizontal solid black lines show the measured weight for that day.

the prediction forecast to be higher than the evaluation forecast.

It is a common problem within broiler production that the measured weight is negatively biased from day 15 and onwards. This has been corrected by multiplying a manually configured and time varying "behavior factor" with the measured weight. We only have the corrected weight which naturally leads to some uncertainty.

5.2. Discussion of Figure 8a

We see that both the prediction and evaluation forecasts show good overall forecasting capability, in the sense that it tracks the measured output shape throughout the batch – however, the model has a tendency to underestimate for all outputs. Comparing the prediction and evaluation forecasts we see that the evaluation forecast on day 34 has an ensemble mean of 38 g higher broiler weight, 67 g/bird lower feed consumption and 38 g/bird lower water consumption for the evaluation forecast – a higher broiler

			#Selected /								**Lags (% of selected)**								
Type		**Variable**	**#Availiable**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Broiler Weight	Input	Measured Relative Humidity	17/18 (94%)	29	24	12	0	18	0	6	24	29	12	18	18	6	6	0	0
		Measured Temperature	4/18 (22%)	0	0	25	25	50	25	25	25	0	0	0	0	0	25	0	0
		Light Intensity Reference	5/12 (42%)	80	0	0	0	0	20	100	0	0	0	0	0	0	0	0	0
		Measured Outside Temperature	12/18 (67%)	8	17	0	17	17	17	25	8	8	17	8	8	0	17	17	17
		Ventilation Demand	16/18 (89%)	19	6	13	25	13	0	13	25	19	0	6	19	6	0	25	13
	Output	Broiler Weight	18/18 (100%)	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		Feed Consumption	18/18 (100%)	39	11	6	0	0	6	0	11	11	0	22	11	39	11	17	17
		Water Consumption	18/18 (100%)	44	6	6	0	0	0	11	28	6	28	17	0	22	11	6	17
Feed Consumption	Input	Heating Demand	8/18 (44%)	0	0	50	13	0	0	13	13	13	0	0	38	0	0	0	63
		Measured Relative Humidity	14/18 (78%)	14	14	29	14	7	43	14	21	14	0	14	14	0	0	0	0
		Measured Temperature	4/18 (22%)	50	25	100	25	0	0	0	0	0	0	0	0	0	0	0	0
		Light Intensity Reference	7/12 (58%)	29	0	0	0	0	0	0	29	14	0	0	14	29	29	14	43
		Measured Outside Temperature	13/18 (72%)	0	15	31	15	0	8	15	8	0	8	31	8	15	0	15	31
		Ventilation Demand	8/18 (44%)	38	0	50	0	13	13	25	0	0	0	13	13	25	0	13	0
	Output	Broiler Weight	18/18 (100%)	39	28	17	0	22	0	33	0	6	0	0	0	11	11	17	17
		Feed Consumption	18/18 (100%)	100	61	0	39	0	0	0	0	0	0	0	0	0	0	0	0
		Water Consumption	18/18 (100%)	100	50	0	39	0	0	6	0	6	0	0	0	0	0	0	0
Water Consumption	Input	Heating Demand	1/18 (6%)	0	0	100	0	0	0	100	0	0	0	0	0	0	0	0	0
		Measured Relative Humidity	10/18 (56%)	10	30	10	10	0	20	20	10	0	0	10	10	10	20	0	10
		Measured Temperature	3/18 (17%)	67	0	33	33	0	0	0	0	0	0	0	0	0	0	33	0
		Light Intensity Reference	10/12 (83%)	80	0	10	0	0	0	0	80	20	0	0	0	0	0	0	10
		Measured Outside Temperature	8/18 (44%)	0	13	13	0	0	13	13	0	0	0	13	38	13	38	38	0
		Ventilation Demand	13/18 (72%)	54	0	23	0	0	15	15	15	0	8	38	8	0	0	0	8
	Output	Broiler Weight	18/18 (100%)	0	6	11	0	0	28	17	17	6	22	6	6	0	22	0	39
		Feed Consumption	18/18 (100%)	61	0	22	0	0	0	17	11	0	6	22	0	6	0	0	0
		Water Consumption	18/18 (100%)	100	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2: Overview of the selected inputs, input lags and output lags for each of the 18 trials. A separate IVS is presented for each output types. The lags used in the most recent trial is marked with a solid black rectangle, while our interpretation of trends are marked with shaded gray rectangles.

weight for less feed and water. Comparing the raw data of the prediction- and evaluation batch on Figure 7 at the end of the evaluation batch, we see that the evaluation batch produces the same amount of meat with 3.8% less feed and 1.1% less water – supporting the previous observation while assuming unbiased weight measurements.

5.3. Discussion of Figure 8b

We see that the under-estimation trend is present when forecasting weight after day 14, as the weight on day 7 and 14 forecasts show good agreement with the measured weight. The weight on day 21 and 28 forecasts are underestimated when forecasting from before day ≈22, which is caused by a higher than average growth in this period. To elaborate, between day ≈7 to ≈24 the broiler weight is more than one standard deviation lower than the training and testing data, but it is close to the mean broiler weight after day ≈24 – the ensemble model cannot explain this from the input data. This could be caused by biased broiler weight measurements in this interval.

Comparing the evaluation and prediction forecasts, we see that they resemble each other well in terms of ensemble model standard deviation as the shaded area is similar for the two forecasts, indicating that substitute input data from the previous batch can be used to represent the ensemble standard deviation of the current batch despite different ensemble model means.

6. Experimental Forecasting: Trial Evaluations

6.1. Input Selection Discussion

On Table 2 a summary of all selected inputs, input lags and output lags is depicted. In the following, we discuss the selection trends for each output.

Broiler weight

The primarily selected inputs for the broiler weight are the measured relative humidity, measured outside temperature and ventilation demand. The measured relative humidity appears to have a short and long horizon with lags at 0-2 and 7-11 – equivalent of “now” and a day ago (8 samples per 24 hour). The measured outside temperature appears to have no clear trends in the selected lags, which suggest a seasonal dependence rather than a dependence of outside temperature. The ventilation demand also appears to have no clear trends, but an argument for small

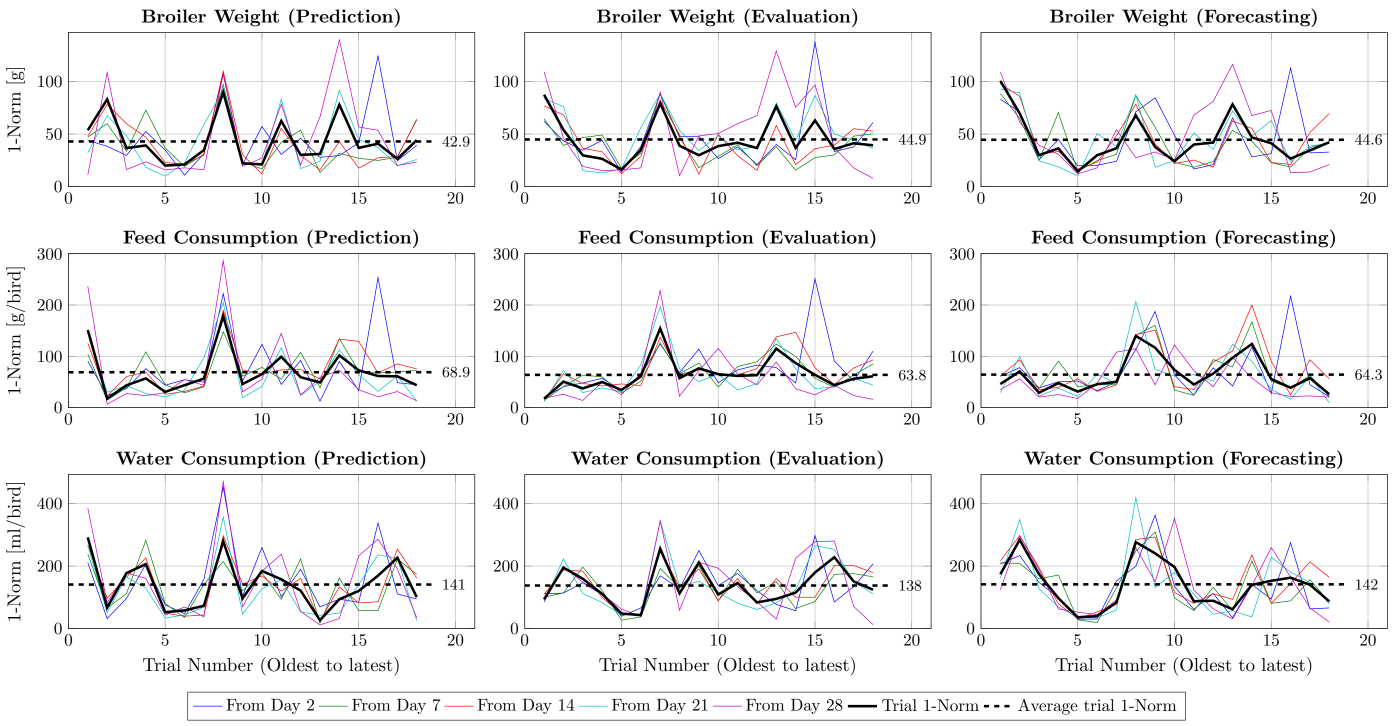


Figure 9: Forecasting performance overview of all trials, where the first column is the prediction batch with known future inputs, the second column is the evaluation batch with known future inputs, and the third column is the evaluation batch with substitute future inputs from the prediction batch. See Section 2.2 for additional details. To give an idea of the forecasting performance throughout the batch, we present the day specific 1-Norm from day 2, 7, 14, 21 and 28 to the batch end calculated using (28). The trial 1-Norm is calculated using (27).

clusters at around lag 0, 3, 7, 11 and 14 can be made with a periodicity of ≈4 lags (12 hours).

The selected outputs for the broiler weight appear to be strongly correlated with lag 0 and 1 of itself. It also appears to be dependent on the two other outputs, either very recently at lag 0 and 1 or after lag 5. A similar, but entirely heuristic claim was made in Johansen et al. (2017).

Feed Consumption

The primary selected inputs for the feed consumption are the measured relative humidity and measured outside temperature. The measured relative humidity appears to be selected within the first 8 lags. Like the broiler weight input selection, the measured outside temperature appears to have no clear trends.

The selected outputs for the feed consumption are primarily between lag 0 and 3 for both the feed consumption and water consumption. Broiler weight appears to have a short term component between lag 0 and 6 and a long term component between lag 12-15.

Water Consumption

The primary selected inputs for the feed consumption are the ventilation demand, light intensity reference and measured relative humidity. The ventilation demand appears to have a weak trend between lag 0 and 11 – an argument can be made that there are small clusters at lag 0, 2, 6 and 10. The light intensity reference has a strong trend at lag 0 and 7, indicating a day rhythm as they are spaced ≈24 hours apart. The measured relative humidity appears to have no strong trends, but it is likely that there are small clusters at lag 1, 5, 12 and 15.

The selected outputs for the water consumption appear to be strongly correlated with lag 0 and 1 of itself, like the broiler weight output. Broiler weight appears to have a long term component between lag 5 and 15, possibly with a small cluster at lag 7, 13 and 15. Feed consumption appears to have two clusters, one between lag 0 and 2, and lag 6 to 10.

Other Trends

The measured temperature was not primarily selected for any of the outputs – but it appears to have great influence on the ensemble model presented in Section 5. This might suggest that the temperature is in an optimum range for the majority of the batches, as we know that temperature has a strong influence on broiler growth. Alternatively, this might be due to it being tightly controlled.

Heating demand was not primarily used for any of the outputs, but comes quite close in feed consumption. Hence, heating demand might contribute little to broiler forecasting.

6.2. Trial Performance Evaluations

In this section we comment on the 1-Norm error trial overview depicted on Figure 9. We see small error differences between the overall 1-Norm for all outputs, suggesting that substituting unknown future inputs with inputs from a previous batch does not impact performance significantly. In summary, the ensemble model mean forecasts have an overall 1-Norm broiler weight forecasting error of 42.3g for these trials.

In particular, we see that the average trial 1-Norm of the forecasting trials (column 3) are closer to the evaluation trials (column 2) than the prediction trials (column 1) in terms of broiler weight and feed consumption. This indicates that the past evaluation data contains more predictive power than the future prediction data. Overall, this suggests that using unknown future input substitution produces similar forecasts, as observed in Section 5.

7. Concluding Remarks

In this present work 18 forecasting trials have been carried out on ensemble DNN models that are trained on farm scale broiler batch production data with automatic IVS based on mutual information. We conducted a case study where we presented and interpreted the training and forecasting data, along with a forecast from day 14 to the batch end, and finally a weight on day forecast on day 7, 14, 21, 28 and the batch end from all preceding days. Most importantly, we found empirical evidence suggesting that a dynamic interconnection between environmental broiler house conditions and broiler weight can be captured at least partially by the proposed method. Additionally, we analyzed the 1-Norm forecasting error across all trials, and found that using old inputs from a previous batch as a substitute for unknown future inputs produce similar results in this case. Overall, the proposed method has an average 1-Norm broiler weight forecasting error of 42.3g.

Future work includes: Investigating state estimation as an alternative to reinitializing the model with old output data at every time step when forecasting, as well as establishing a measure of how much the model has to generalize from known training data to forecasting data, and determine the optimal number of neurons.

Acknowledgments

The authors would like to thank broiler manager Erik Hjorth for permitting access to the data used in this work. This work was supported by Innovationsfonden [grant number 5016-00133B].

References

Aerts, J. M., Lippens, M., De Groote, G., Buyse, J., Decuypere, E., Vranken, E., & Berckmans, D. (2003). Recursive prediction of broiler growth response to feed intake by using a time-variant parameter estimation method. *Poultry Science*, *82*, 40–49.

Aggrey, S. (2002). Comparison of three nonlinear and spline regression models for describing chicken growth curves. *Poultry Science*, *81*, 1782–1788.

Ahmadi, H., & Mottaghitalab, M. (2007). Hyperbolastic models as a new powerful tool to describe broiler growth kinetics. *Poultry Science*, *86*, 2461–2465.

Beirlant, J., Dudewicz, E. J., Györfi, L., & Van Der Meulen, E. C. (1997). Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, *6*, 17–39.

Bowden, G. J., Maier, H. R., & Dandy, G. C. (2005). Input determination for neural network models in water resources applications. part 2. case study: forecasting salinity in a river. *Journal of Hydrology*, *301*, 93–107.

Brown, G., Pocock, A., jie Zhao, M., Lujn, M., & Guyon, I. (2012). Conditional likelihood maximisation: A unifying framework for information theoretic feature selection. *Journal of Machine Learning Research*, *13*.

Burden, F., & Winkler, D. (2008). Bayesian regularization of neural networks. In *Methods in Molecular Biology* (pp. 23–42). Springer Science + Business Media.

Cangar, O., Aerts, J.-M., Vranken, E., & Berckmans, D. (2007). Online growth control as an advance in broiler farm management. *Poultry Science*, *86*, 439–443.

Chen, L., Ye, L., Singh, V., Zhou, J., & Guo, S. (2014). Determination of input for artificial neural networks for flood forecasting using the copula entropy method. *Journal of Hydrologic Engineering*, *19*, 04014021.

Demmers, T., Cao, Y., Gauss, S., Lowe, J., Parsons, D., & Wathes, C. (2010). Neural predictive control of broiler chicken growth. *IFAC Proceedings Volumes*, *43*, 311–316.

Du, K.-L., & Swamy, M. N. S. (2014). *Neural Networks and Statistical Learning*. Springer Science + Business Media.

Grahovac, J., Jokić, A., Dodić, J., Vučurović, D., & Dodić, S. (2016). Modelling and prediction of bioethanol production from intermediates and byproduct of sugar beet processing using neural networks. *Renewable Energy*, *85*, 953–958.

Hasan Elerolu, A. e. F. N. o., Arda Yldrm, & Duman, M. (2014). Comparison of growth curves by growth models in slowgrowing chicken genotypes raised the organic system. *International Journal of Agriculture and Biology*, *16*, 529–535.

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. MacMillan Publishing Company.

He, J., Valeo, C., Chu, A., & Neumann, N. F. (2011). Prediction of event-based stormwater runoff quantity and quality by ANNs developed using PMI-based input selection. *Journal of Hydrology*, *400*, 10–23.

Hernandez, S. C., Bueno, J. A., Sanchez, E. N., & Diaz-Jimenez, L. (2013). State estimation by artificial neural networks in a continuous bioreactor. *IFAC Proceedings Volumes*, *46*, 215–220.

Johansen, S., Bendtsen, J. D., Riisgaard-Jensen, M., & Mogensen, J. (2017). Data driven broiler weight forecasting using dynamic neural network models. *IFAC*, *2017*.

Li, X., Maier, H. R., & Zecchin, A. C. (2015). Improved PMI-based input variable selection approach for artificial neural network and other data driven environmental and water resource models. *Environmental Modelling & Software*, *65*, 15–29.

Lopes, A. Z., Ferreira, L., Junior, T. Y., & Lacerda, W. S. (2008).

Modeling productive performance of broiler chickens with artificial neural network. In *Livestock Environment VIII, 31 August - 4 September 2008, Iguassu Falls, Brazil*. American Society of Agricultural and Biological Engineers (ASABE).

Madsen, K., Nielsen, H. B., & Tingleff, O. (2004). Methods for non-linear least squares problems (2nd ed.). Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby.

May, R., Dandy, G., & Maier, H. (2011). Review of input variable selection methods for artificial neural networks. In *Artificial Neural Networks - Methodological Advances and Biomedical Applications*. InTech.

May, R. J., Dandy, G. C., Maier, H. R., & Nixon, J. B. (2008a). Application of partial mutual information variable selection to ANN forecasting of water quality in water distribution systems. *Environmental Modelling & Software*, *23*, 1289–1299.

May, R. J., Maier, H. R., Dandy, G. C., & Fernando, T. G. (2008b). Non-linear variable selection for artificial neural networks using partial mutual information. *Environmental Modelling & Software*, *23*, 1312–1326.

Nair, V. V., Dhar, H., Kumar, S., Thalla, A. K., Mukherjee, S., & Wong, J. W. (2016). Artificial neural network based modeling to evaluate methane yield from biogas in a laboratory-scale anaerobic bioreactor. *Bioresource Technology*, *217*, 90–99.

Nasimi, R., & Irani, R. (2014). Identification and modeling of a yeast fermentation bioreactor using hybrid particle swarm optimization-artificial neural networks. *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, *36*, 1604–1611.

Nguyen, D., & Widrow, B. (1990). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In *1990 IJCNN International Joint Conference on Neural Networks*. Institute of Electrical & Electronics Engineers (IEEE).

OECD-FAO (2016). *OECD-FAO Agricultural Outlook 2016-2025*. OECD Publishing.

Rajamani Doraiswami, C. D., & Stevenson, M. (2014). Closed loop identification. In *Identification of Physical Systems* (pp. 357–378). Wiley-Blackwell.

Sameer Agarwal, K. M., & Others (2015). Ceres solver. `http://ceres-solver.org`.

Sharma, A. (2000). Seasonal to interannual rainfall probabilistic forecasts for improved water supply management: Part 1 — a strategy for system predictor identification. *Journal of Hydrology*, *239*, 232–239.

Stacey, K., Parsons, D., Frost, A., Fisher, C., Filmer, D., & Fothergill, A. (2004). An automatic growth and nutrition control system for broiler production. *Biosystems Engineering*, *89*, 363–371.

Wang, H., Yan, X., Chen, H., Chen, C., & Guo, M. (2015). Chlorophyll-a predicting model based on dynamic neural network. *Applied Artificial Intelligence*, *29*, 962–978.

Wathes, C., Kristensen, H., Aerts, J.-M., & Berckmans, D. (2008). Is precision livestock farming an engineer's daydream or nightmare, an animal's friend or foe, and a farmer's panacea or pitfall? *Computers and Electronics in Agriculture*, *64*, 2–10.

Wolfgang Hrdle, S. S., Marlene Mller, & Werwatz, A. (2004). *Nonparametric and Semiparametric Models*. Springer.