AAAS
Plant Phenomics
Volume 2020, Article ID 4261965, 16 pages
https://doi.org/10.34133/2020/4261965

Plant Phenomics
A SCIENCE PARTNER JOURNAL

Research Article

Semisupervised Deep State-Space Model for Plant Growth Modeling

S. Shibata,[1] R. Mizuno,[1] and H. Mineno[2,3]

[1]*Graduate School of Integrated Science and Technology, Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu, Shizuoka 432-8011, Japan*
[2]*College of Informatics, Academic Institute, Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu, Shizuoka 432-8011, Japan*
[3]*JST, PRESTO, 4-1-8 Honcho, Kawaguchi, Saitama 332-0012, Japan*

Correspondence should be addressed to H. Mineno; mineno@inf.shizuoka.ac.jp

Received 12 February 2019; Accepted 15 March 2020; Published 25 May 2020

The optimal control of sugar content and its associated technology is important for producing high-quality crops more stably and efficiently. Model-based reinforcement learning (RL) indicates a desirable action depending on the type of situation based on trial-and-error calculations conducted by an environmental model. In this paper, we address plant growth modeling as an environmental model for the optimal control of sugar content. In the growth process, fruiting plants generate sugar depending on their state and evolve via various external stimuli; however, sugar content data are sparse because appropriate remote sensing technology is yet to be developed, and thus, sugar content is measured manually. We propose a semisupervised deep state-space model (SDSSM) where semisupervised learning is introduced into a sequential deep generative model. SDSSM achieves a high generalization performance by optimizing the parameters while inferring unobserved data and using training data efficiently, even if some categories of training data are sparse. We designed an appropriate model combined with model-based RL for the optimal control of sugar content using SDSSM for plant growth modeling. We evaluated the performance of SDSSM using tomato greenhouse cultivation data and applied cross-validation to the comparative evaluation method. The SDSSM was trained using approximately 500 sugar content data of appropriately inferred plant states and reduced the mean absolute error by approximately 38% compared with other supervised learning algorithms. The results demonstrate that SDSSM has good potential to estimate time-series sugar content variation and validate uncertainty for the optimal control of high-quality fruit cultivation using model-based RL.

1. Introduction

Several studies have been performed to evaluate advanced cultivation techniques for stable and efficient production of high-quality crops based on farmers' experience and intuition [1–4]. For example, water stress cultivation of tomato plants is a technique that increases their sugar content by reducing irrigation. The technique requires sensitive irrigation control to provide the appropriate water stress throughout the cultivation period. A fine balance must be achieved because insufficient water stress does not improve sugar content while excessive water stress causes permanent withering. Such a technique is currently limited to expert farmers, and there have been some studies conducted to estimate water stress indirectly from soil moisture or climatic environmental factors such as temperature, humidity, and sunlight [5–10]. Recent studies have attempted to assess water stress with deep neural networks by monitoring plant motion caused by withering [11, 12]. Those studies contributed to the quantification of water stress to improve water stress cultivation to some extent. However, the purpose of water stress cultivation is to raise the sugar content, and a technique to directly control the sugar content flexibly is of interest. In this regard, our final goal is to develop a method to determine the optimal action to achieve the desired sugar content of greenhouse tomato plants at harvest stably and efficiently. In this study, we aim to develop a plant growth model to estimate time-series sugar content variation employing reinforcement learning, as the first step toward the final goal.

Reinforcement learning (RL) [13–15] acquires an optimal strategy through the experience of an agent performing an action in an environment and has demonstrated high

flexibility and nontask-dependent representation capabilities. There are two types of RL: model-free RL [16, 17] and model-based RL [18, 19]. Model-free RL does not use the environmental information (state transition of the environment) to predict how the environment changes and what type of reward is obtainable when the environment is modified. By contrast, model-based RL uses the information of the state transition of the environment. Therefore, model-based RL is better than model-free RL at judging behavior based on a long-term plan with regard to a future state. Namely, model-based RL is expected to perform well in determining optimal actions to achieve the desired sugar content at harvesting.

The training of model-based RL involves two steps: (1) modeling an environment and (2) planning to learn the optimal policy for the model. In this paper, we focus on modeling an environment and developing a plant growth model for the optimal control of water stress cultivation. The model-based RL's environmental model is often a probabilistic model evaluated based on the standard deviation, because the plant states and the surrounding environment data are time-series data and generally contain a significant amount of noise [20]. The noise is affected not only by external factors but also by internal factors such as nonlinearly distributed plant growth. For example, nondestructively measured sugar content data based on spectroscopy varies depending on the location of the measurement point on a fruit because of the uneven internal structure of a fruit. Thus, we need to select a robust model that can properly handle such noisy data.

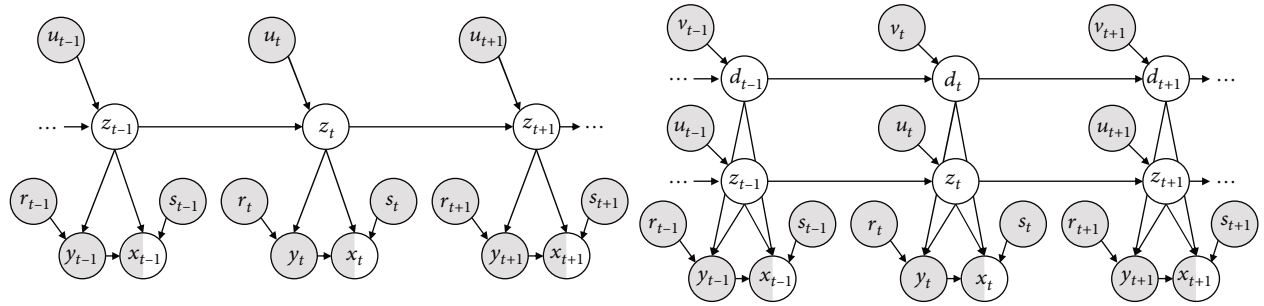
Among probabilistic models, the generative model is known as the most suitable for plant growth modeling, because generative models assign low probability to outliers. By contrast, discriminative models process the training dataset without considering the effects of noise. The generative model not only is robust to noise but also has good characteristics for predicting future states [20] and for generalization performance [21].

When model-based RL is used for the optimal cultivation of high sugar content fruits, it is necessary to predict future plant conditions from the cultivation environment based on present conditions. Moreover, it is important that the modeling method can be applied to different plant varieties and specific environments as well as to various environments. Therefore, we use the generative model to achieve high generalization performance of plant growth modeling, which requires predictability of the future states. In particular, we try to make the generative models much more robust and flexible by using sequential deep generative models combined with a state-space model (SSM, a typical generative model for time-series data) and because deep neural networks have fewer restrictions.

The variational autoencoder (VAE) [22] is a deep generative model for nonsequential data, and the parameters are optimized via stochastic gradient variational Bayes (SGVB) [23]. The stochastic recurrent networks (STORN) [23] are highly structured generative processes that are difficult to fit to deterministic by combining the elements of the VAE. Additionally, STORN is able to generate high-dimensional sequences such as music by including recurrent neural networks (RNN) in the structure of VAE and represents stochastic sequential modeling by inputting a sequence independently sampled from the posterior to a standard RNN. The variational RNN (VRNN) was proposed by Chung et al. [24] as a model similar to STORN. The main difference is that the prior of the latent variable depends on all previous information via a hidden state in the RNN. The introduction of temporal information has been shown to help in modeling highly structured sequences. VAE is also applied for optimal control. Watter et al. [25] addressed the local optimal control of high-dimensional nonlinear dynamic systems. Considering optimal control as the identification of the low-dimensional latent space, their proposed model Embed to Control (E2C) is trained while compressing high-dimensional observations such as images. Their results showed that E2C exhibits strong performance in various control tasks. Krishnan et al. [26] modeled the change of a patient's state over time using temporal generative models called Deep Kalman Filters (DKF). Unlike previous methods, DKF incorporates action variables to express factors external to patients, such as prescribing medication or performing surgery. In particular, structured variational inference is introduced in DKF to cater to the unique problems of living organisms, such as considering that a patient's states vary slowly and that external factors may have long-term effects on patients. These phenomena are similar to plant growth modeling.

The methods described above require a comparatively large-scale dataset to model complex generative processes. On the other hand, creating a large-scale dataset for the time-series sugar content of fruits is not temporally or financially easy because it is necessary to use a sensor to make gentle contact with the fruit manually. In addition, because measurements are performed manually, it is necessary to establish the methods based on various considerations such as measurement time, position, repetition of measurements, and measurement of the surrounding tomatoes to reduce the variance of the measurement value. By contrast, it is common to measure the fruit juice of representative fruits at the time of harvest (destructive measurement). Once the destructive measurement is performed, it is not possible to measure the same fruit over time. For these reasons, the data collection interval required for sugar content is longer than that of automatically sensed data such as temperature, humidity, and solar radiation. In particular, creating a large-scale dataset for time-series crop condition and quality is also not easy using manual measurements because of the workload and cost factors. Thus, it is important to develop a suitable method even though the amount of available data is not large.

In this study, we propose a novel sequential deep generative model called a semisupervised deep state-space model (SDSSM) to evaluate such defective data. SDSSM is similar to DKF in that a deep neural network is used for enforcement of the representation capability of SSM. On the other hand, the major difference is that SDSSM is trained by semisupervised learning to achieve a balance between high generalization performance and high representation power, even if some types of training data are sparse.



(a) Cont-SDSSM and Disc-SDSSM (b) 2L-SDSSM

Figure 1: Graphical representations of proposed models.

2. Materials and Methods

2.1. Plant Growth Model

2.1.1. Overview of SDSSM. Based on the general SSM, we assume the following generative processes for plant growth modeling:

$$p_\theta(z_t \mid z_{t-1}, u_t) = N(z; \mu_z(z_{t-1}, u_t), \sigma_z(z_{t-1}, u_t)), \text{(system model)}, \tag{1}$$

$$p_\theta(x_t \mid z_t, s_t) = N\left(x; \mu_y(z_t, s_t), \sigma_y(z_t, s_t)\right), \text{(observation model)}, \tag{2}$$

where z_t and x_t are latent variables and observed variables, respectively, at time step t. The probabilistic models are shown in Figure 1. In our task, the latent variable z_t denotes the plant states. We assume that the water content and the duration of plant growth, which are particularly strongly related to sugar content, are plant states.

Regarding these two states as single-type states with continuous variation, we set a normal distribution to the latent variables z_t according to the previous studies of deep generative models, where a normal distribution was adopted for continuous values. The observed variables x_t indicate the sugar content and are assumed to follow a normal distribution considering the continuous variation of sugar content. Moreover, we introduce the action variables u_t, r_t, and s_t to the system model and observation model, respectively.

The action variable u_t is added to the process of the state transition considering that plant states vary according not only to previous states but also with external factors such as temperature. In fact, the accumulated temperature is well-known in the agricultural domain as the growth indicator for plants. The action variable s_t is added to the process of the emission because sugar is produced via photosynthesis based on a plant's state and its surrounding environmental variables such as carbon dioxide (CO_2) concentration. The detailed settings of each random variable are discussed in Section 2.2.2.

Training the generative model based on Equations (1) and (2) using SGVB requires a large amount of data owing to the assumption of the complex generative process: there are implicitly two types of states in the single latent space, and the state transition and emission of the observation are strongly nonlinear. Deep generative models trained by semi-supervised learning have recently demonstrated significantly improved generalization capabilities in comparison to previous methods and perform very well even for very small datasets [21, 27, 28].

In particular, conditional VAE (CVAE) is a typical deep generative model trained by semisupervised learning, and the generative model and learning algorithm are based on VAE. VAE learns the parameters simultaneously with the inference of the latent states using only observations. On the other hand, CVAE introduces labels for observations as latent variables to improve the quality of prediction by exploring information in the data density. However, CVAE does not assume missing observations. To explore missing observations efficiently, we take a different approach to CVAE by applying a probabilistic model of SDSSM, as shown in Figure 1(a). Formally, we assume the following generative model:

$$\begin{aligned} p_\theta(z_t \mid z_{t-1}, u_t) &= N(z; \mu_z(z_{t-1}, u_t), \sigma_z(z_{t-1}, u_t)), \text{(system model)}, \\ p_\theta(x_t \mid z_t, s_t) &= N(x; \mu_x(z_t, s_t), \sigma_x(z_t, s_t)), \text{(observation model)}, \\ p_\theta(y_t \mid z_t, r_t) &= N\left(y; \mu_y(z_t, r_t), \sigma_y(z_t, r_t)\right), \text{(observation model)}, \end{aligned} \tag{3}$$

where y_t is an additional observed variable that follows a normal distribution and is generated from the same latent variable z_t as is x_t, and r_t is the action variable added to the emission of the observation y_t. Thus, we assume that observation y_t is a generative process similar to observation x_t. The difference between observations appears through the nonlinear functions having different forms and inputs. In particular, sharing latent variables allows one to infer the latent states complementarily to the other observations, even when one observation is missing. Therefore, SDSSM learns the complex latent space as efficiently as other deep generative models, even when the training dataset includes few observations. Here, the functions μ_z, σ_z, μ_y, σ_y, μ_x, and σ_x are

arbitrary nonlinear functions parameterized by deep neural networks (DNNs) as follows:

$$\begin{aligned}\mu_z &= \mathrm{DNN}_z(z_{t-1}, u_t), \log \sigma_z = \mathrm{DNN}_z(z_{t-1}, u_t),\\ \mu_x &= \mathrm{DNN}_x(z_t, s_t), \log \sigma_x = \mathrm{DNN}_x(z_t, s_t),\\ \mu_y &= \mathrm{DNN}_y(z_t, r_t), \log \sigma_y = \mathrm{DNN}_y(z_t, r_t),\end{aligned} \tag{4}$$

where DNN_z, DNN_x, and DNN_y are deep neural networks that have weight matrices w_z, w_x, and w_y, respectively. Thus, the parameters of the generative model are $\theta = \{w_z, w_x, w_y\}$. According to Kingma and Welling [29], we assume that μ_z, μ_x, and μ_y denote the mean and σ_z, σ_x, and σ_y indicate a diagonal covariance matrix. To ensure definite positivity, the outputs from deep neural networks for the diagonal covariance matrix are taken using their logarithm.

2.1.2. Learning SDSSM Using SGVB. We maximize the marginal log-likelihood based on the labeled dataset (D_l) and the unlabeled dataset (D_u) to optimize the parameters θ and φ in the generative model. The labeled dataset $(D_l = \{(x_{t_1^{(l)}}, y_{t_1^{(l)}}, u_{t_1^{(l)}}, s_{t_1^{(l)}}, r_{t_1^{(l)}}) \cdots (x_{t_m^{(l)}}, y_{t_m^{(l)}}, u_{t_m^{(l)}}, s_{t_m^{(l)}}, r_{t_m^{(l)}})\})$ does not include missing values, whereas the unlabeled dataset $(D_u = \{(y_{t_1^{(u)}}, u_{t_1^{(u)}}, s_{t_1^{(u)}}, r_{t_1^{(u)}}) \cdots (y_{t_n^{(u)}}, u_{t_n^{(u)}}, s_{t_n^{(u)}}, r_{t_n^{(u)}})\})$ includes missing values of observations x_t. Note that the labeled data is x_t. Here, the superscript l represents labeled and the superscript u represents unlabeled. We treat x_t in the labeled dataset as observed variables and x_t in the unlabeled dataset as latent variables. In the following, we omit the dependence of p and q on u_t, v_t, s_t, and r_t. The marginal log-likelihood on the labeled dataset is as follows:

$$\mathrm{Log}p_\theta(x_{1:T}, y_{1:T}) = \log \int p_\theta(x_{1:T}, y_{1:T} \mid z_{1:T}) dz_{1:T}. \tag{5}$$

Note that we describe $x_1, x_2, \cdots, x_T$ at each time step t ($t = 1, 2, \ldots, T$) as $x_{1:T}$. Following the principle of SGVB, we maximize the evidence lower bound (ELBO) with respect to parameters θ and φ instead of maximizing the marginal log-likelihood directly. We derive a labeled ELBO $\mathscr{L}^l(x, y; \theta, \varphi)$ by introducing the recognition model into Equation (14) and using Jensen's inequality:

$$\begin{aligned}\mathrm{Log}p_\theta(x_{1:T}, y_{1:T}) &\geq \log \int q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T}) \frac{p_\theta(x_{1:T}, y_{1:T} z_{1:T})}{q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T})} dz_{1:T}\\ &= -\mathscr{L}^l(\mathrm{x}, \mathrm{y}; \theta, \varphi),\end{aligned} \tag{6}$$

where q_φ is the posterior approximation of latent variable z_t, called the recognition model. In general, SGVB approximates the true posterior without factorization. There is an assumption that the true posterior distribution is factorized to a simpler form using a mean-field approximation in the framework of variational inference. Relaxing the constraint contributes to the improvement of the representation capability.In the case of sequential data, the Kullback–Leibler (KL) divergence terms in ELBO often have no analytic form. The gradients of the KL terms are derived by sampling estimation so that insufficient sampling leads to high-variance estimations [26]. Fraccaro et al. [30] derived low-variance estimators of the gradients using true factorization of the posterior distribution according to the Markov property. On the basis of their work, we factorize the recognition model as follows:

$$p(z_{1:T} \mid x_{1:T}, y_{1:T}) = \prod_{t=1}^{T} p(z_t \mid z_{t-1}, x_{t:T}, y_{t:T}). \tag{7}$$

We set the initial latent state to zero: $z_0 = 0$. The studies mentioned above derived a similar form, such that a latent state at time step t is conditioned by previous latent states and by the sequential observations and action variables from time step t to T. In our case, the form (including the future sequence of observations x_t) cannot be calculated owing to the assumption that the observations are missing. However, Krishnan et al. [26] demonstrated that the sequence from the initial time step 0 to time step t contains sufficient information. Drawing on their work, we factorize the recognition model as follows:

$$q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T}) = \prod_{t=1}^{T} q_\varphi(z_t \mid z_{t-1}, x_{1:t}, y_{1:t}). \tag{8}$$

Based on the decomposed recognition models, the labeled ELBO $\mathscr{L}^l(x, y; \theta, \varphi)$ is defined as follows:

$$\begin{aligned}-\mathscr{L}^l(\mathrm{x}, \mathrm{y}; \theta, \varphi) = &\sum_{\mathrm{t}=1}^{T} E_{q_\varphi(z_t)}[\log p_\theta(y_t \mid z_t) + \log p_\theta(x_t \mid y_t, z_t)]\\ &- \beta \mathrm{KL}\Big(q_\varphi(z_1) \| p_\theta(z_1)\Big)\\ &- \sum_{t=2}^{T} E_{q_\varphi(z_{t-1})}\Big[\beta \mathrm{KL}\Big(q_\varphi(z_t) \| p_\theta(z_t \mid z_{t-1})\Big)\Big],\end{aligned} \tag{9}$$

where $q_\varphi(z_t) = q_\varphi(z_t \mid z_{t-1}, x_t, y_t)$. The expectations with respect to $q_\varphi(z_t)$ and $q_\varphi(z_{t-1})$ in Equation (9) are estimated via Monte Carlo sampling after applying the reparameterization trick. KL denotes a Kullback–Leibler divergence. All KL terms in Equation (9) can be computed analytically. Additionally, we add the weight coefficient β for the KL divergence and gradually increase it from a small number during training to facilitate flexible modeling by the explicit avoidance of restrictions.In the unlabeled dataset, we are interested in the marginal log-likelihood $\log p_\theta(y_{1:T})$, which is derived by marginalizing out not only z_t but also x_t. We obtain an unlabeled ELBO from the marginal log-likelihood in the same way as we obtained the labeled ELBO. The unlabeled ELBO is decomposed as follows

by applying d-separation to the graphical model:

$$\begin{aligned}\mathrm{Log}p_\theta(y_{1:T}) &\geq \iint q_\varphi(z_{1:T}, x_{1:T} \mid y_{1:T}) \log \frac{p_\theta(x_{1:T}, y_{1:T}, z_{1:T})}{q_\varphi(z_{1:T}, x_{1:T} \mid y_{1:T})} dz_{1:T} \mathrm{d}x_{1:T} \\ &= \iint q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T}) q_\varphi(x_{1:T} \mid y_{1:T}) \\ &\quad \cdot \log \frac{p_\theta(x_{1:T}, y_{1:T}, z_{1:T})}{q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T}) q_\varphi(x_{1:T} \mid y_{1:T})} dz_{1:T} \mathrm{d}x_{1:T}.\end{aligned} \tag{10}$$

Unlike the labeled dataset, an unlabeled ELBO has two recognition models, $q_\varphi(z_{1:T} \mid x_{1:T}, y_{1:T})$ and $q_\varphi(x_{1:T} \mid y_{1:T})$, because the two latent variables, z_t and x_t, are included. The former has the same form as the recognition model of the labeled ELBO. On the other hand, the latter is factorized as follows, following a similar approach to the labeled dataset:

$$q_\varphi(x_{1:T} \mid y_{1:T}) = \prod_{t=1}^{T} q_\varphi(x_t \mid y_{t:T}). \tag{11}$$

Using this decomposed recognition model, an unlabeled ELBO is eventually defined as

$$-\mathscr{L}^u(\mathrm{x}, \mathrm{y}; \theta, \varphi) = \sum_{\mathrm{t}=1}^{T} E_{q_\varphi(x_t \mid y_t)} \left[-\mathscr{L}^l(\mathrm{x}, \mathrm{y}; \theta, \varphi) + H\left[q_\varphi(x_t \mid y_t)\right] \right], \tag{12}$$

where $H[q_\varphi(x_t \mid y_t)]$ denotes the entropy of the recognition model $q_\varphi(x_t \mid y_t)$. The unlabeled ELBO $\mathscr{L}^u(x, y; \theta, \varphi)$ includes the labeled ELBO $\mathscr{L}^l(x, y; \theta, \varphi)$, and all probability models except for the recognition model $q_\varphi(x_t \mid y_t)$ are shared by the labeled ELBO and unlabeled ELBO. An expectation in the unlabeled ELBO is estimated via Monte Carlo sampling from the recognition model $q_\varphi(x_t \mid y_t)$. We derive an objective function by summing the labeled and unlabeled ELBOs as follows:

$$J = \sum_{D_l} \mathscr{L}^l(\mathrm{x}, \mathrm{y}; \theta, \varphi) + \sum_{D_u} \mathscr{L}^u(\mathrm{x}, \mathrm{y}; \theta, \varphi) + \alpha E_{D_u} \left[-\log q_\varphi(x_t \mid y_t) \right], \tag{13}$$

where α denotes a small positive constant. Because the recognition model $q_\varphi(x_t \mid y_t)$ has only an unlabeled ELBO, it does not acquire label information during training. In accordance with Krishnan et al. [26], we add a regression term to the objective function to train the recognition model using both the labeled dataset and the unlabeled dataset. The objective function is differentiable owing to the differentiable labeled and unlabeled ELBOs, and the parameters θ and φ are optimized simultaneously by stochastic gradient descent via back-propagation.

2.1.3. Extension of SDSSM. We propose two additional types of extensions to SDSSM depending on some assumptions of the latent space. We call the SDSSM described above a continuous SDSSM (Cont-SDSSM) to distinguish it from the other two models. There are two main plant states that are strongly related to sugar content.

The first is the growth stage [31]. The products of photosynthesis, such as organic acid and sugar content, are accumulated in the fruit, and the ratio of accumulated components varies depending on the growth stage with the transition in metabolism. Therefore, the growth stage is considered essential for plant growth modeling. Second, the sugar content also varies depending on the water content in the plant because insufficient water content often suppresses photosynthesis. Regarding these complicated state transitions with continuous variation, Cont-SDSSM uses single latent variables that follow a normal distribution. We have to consider the appropriate distribution to model the complex transitions and highly structured generative processes based on a plant's growth and surrounding environmental data.

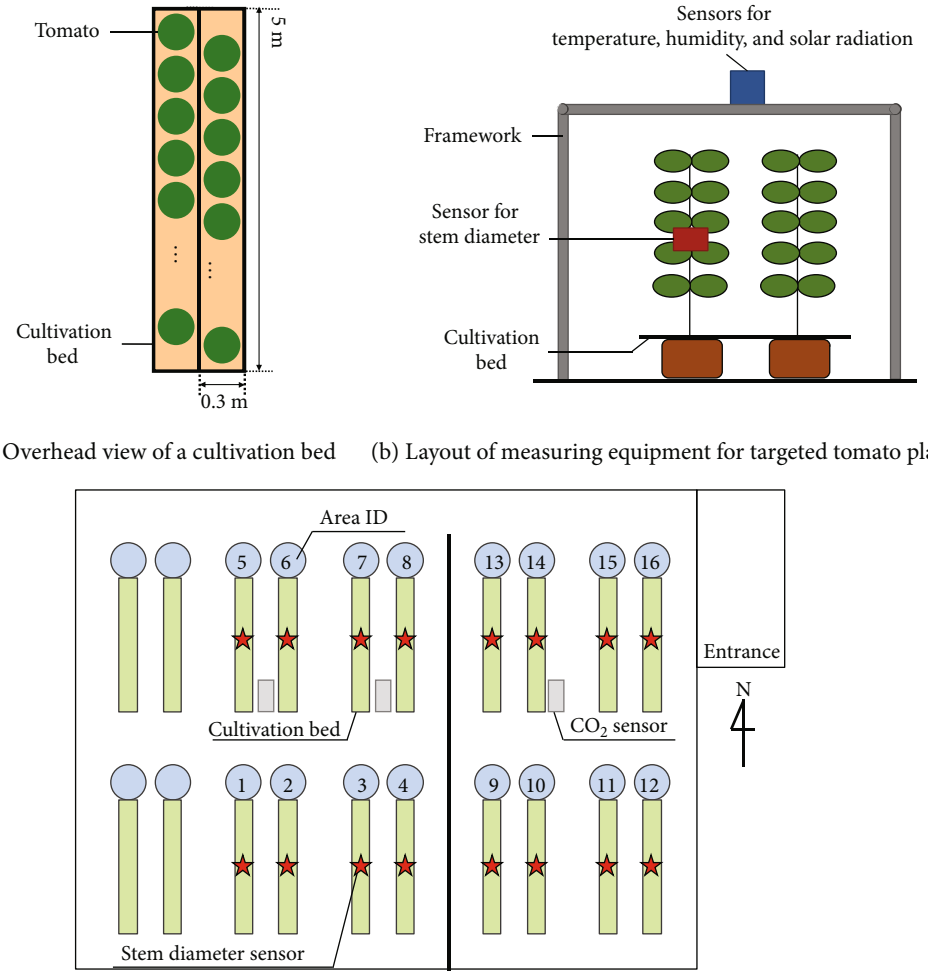
As another approach for modeling the plant states, we design a simpler model called discrete SDSSM (Disc-SDSSM), which represents only the growth stage without considering the water content. In Disc-SDSSM, the forms of generative model, recognition model, and objective function are the same as those in Cont-SDSSM. The major difference is that the latent states follow a categorical distribution because the growth stage is defined as the plant growth class before the harvest, e.g., the flowering stage and ripening stage, which implies discrete growth. We use a Gumbel-Softmax distribution as the posterior of the latent states z_t (instead of a normal distribution) to obtain differentiable categorical samples via the categorical reparameterization trick [32]. Formally, the generative model is defined as follows:

$$\begin{aligned}&p_\theta(z_t \mid z_{t-1}, u_t) = \mathrm{Cat}(z; \pi_z(z_{t-1}, u_t)), \text{(system model)}, \\ &p_\theta(x_t \mid z_t, \mathrm{s}_t) = N(x; \mu_\mathrm{x}(z_t, s_t), \sigma_\mathrm{x}(z_t, s_t)), \text{(observation model)}, \\ &p_\theta(y_t \mid z_t, r_t) = N\left(y; \mu_y(z_t, r_t), \sigma_y(z_t, r_t)\right), \text{(observation model)},\end{aligned} \tag{14}$$

where π_z is an arbitrary nonlinear function parameterized by deep neural networks as follows: $\pi_\mathrm{z} = \mathrm{NN}_z(z_{t-1}, u_t)$.As a different approach to representing plant states, we design a generative model called two latent SDSSM (2L-SDSSM), as shown in Figure 1(b). 2L-SDSSM uses two types of latent variables, z_t and d_t, to clearly separate the two plant states. The generative model is defined as follows:

$$\begin{aligned}&p_\theta(z_t \mid z_{t-1}, u_t) = N(z; \mu_\mathrm{z}(z_{t-1}, u_t), \sigma_z(z_{t-1}, u_t)), \text{(system model)}, \\ &p_\theta(d_t \mid d_{t-1}, v_t) = \mathrm{Cat}(d; \pi_\mathrm{d}(d_{t-1}, v_t)), \text{(system model)}, \\ &p_\theta(x_t \mid z_t, \mathrm{s}_t) = N(x; \mu_\mathrm{x}(z_t, s_t), \sigma_\mathrm{x}(z_t, s_t)), \text{(observation model)}, \\ &p_\theta(y_t \mid z_t, r_t) = N\left(y; \mu_y(z_t, r_t), \sigma_y(z_t, r_t)\right), \text{(observation model)},\end{aligned} \tag{15}$$

where z_t denotes the water content in the plants, d_t denotes the growth stage, and v_t is an action variable



(a) Overhead view of a cultivation bed (b) Layout of measuring equipment for targeted tomato plants

(c) Overhead view of horticulture beds

Figure 2: Experimental environment.

playing the same role as the action variable u_t in Cont-SDSSM. The random variables z_t and d_t are mutually independent latent variables that follow a normal distribution and category distribution, respectively. We sample random latent variables d_t from a Gumbel-Softmax distribution via the categorical reparameterization trick, which is similar to our handling of Disc-SDSSM. The labeled and unlabeled ELBOs have different forms owing to differences in the generative model as follows:

$$\begin{aligned}-\mathscr{L}^l(\mathrm{x},\mathrm{y};\theta,\varphi) = \sum_{t=1}^{T} E_{q_\varphi(z_t),q_\varphi(d_t)}[\log p_\theta(x_t \mid z_t) + \log p_\theta(y_t \mid z_t)] \\ - E_{q_\varphi(z_{t-1}),q_\varphi(d_{t-1})}\left[\beta \mathrm{KL}\left(q_\varphi(z_t)\|p_\theta(z_t \mid z_{t-1})\right)\right. \\ \left. + \beta \mathrm{KL}\left(q_\varphi(d_t)\|p_\theta(d_t \mid d_{t-1})\right)\right], \\ \mathscr{L}^u(\mathrm{x},\mathrm{y};\theta,\varphi) = \sum_{t=1}^{T} E_{q_\varphi(x_t)}\left[-\mathscr{L}^l(\mathrm{x},\mathrm{y};\theta,\varphi) + H\left[q_\varphi(x_t)\right]\right],\end{aligned} \tag{16}$$

where $q_\varphi(z_t) = q_\varphi(z_t \mid z_{t-1}, x_t, y_t)$, $q_\varphi(d_t) = q_\varphi(d_t \mid d_{t-1}, x_t, y_t)$, and $q_\varphi(x_t) = q_\varphi(x_t \mid y_t)$.

2.2. Experiments

2.2.1. Experimental Dataset. We grew tomato plants (*Solanum lycopersicum L.*) in a greenhouse at the Shizuoka Prefectural Research Institute of Agriculture and Forestry in Japan from August 28 to November 18, 2017. There were 16 cultivation beds in the greenhouse, as shown in Figure 2(c), and we cultivated 24 individual tomato plants in each cultivation bed, as shown in Figure 2(a). The tomatoes were grown by three-step dense-planting hydroponic cultivation. Under the three-step dense-planting hydroponic cultivation (Figure 2(b)), the quantity and timing of irrigation greatly affects the quality (sugar content) of tomatoes.

To collect data, we installed sensors in the greenhouse to measure the temperature, humidity, solar radiation, CO_2 concentration, and stem diameter. The stem diameter was measured by laser displacement sensors (HL-T1010A, Panasonic Corporation) of the target tomato plants. The stem

Table 1: Variables in the proposed models.

Variable	Cont-SDSSM	Disc-SDSSM	2L-SDSSM
x_t	Sugar content	Sugar content	Sugar content
y_t	Stem diameter	Stem diameter	Stem diameter
$\boldsymbol{u_t}$	Temperature, solar radiation, VPD, elapsed date[1], accumulated temperature[2]	Elapsed date[1], accumulated temperature[2]	Temperature, solar radiation, VPD
$\boldsymbol{v_t}$	—	—	Elapsed date[1], accumulated temperature[2]
$\boldsymbol{s_t}$	CO_2 concentration, solar radiation, step ID	CO_2 concentration, solar radiation, step ID	CO_2 concentration, solar radiation, step ID

[1]Number of days after flowering. [2]Summation of daily temperatures from the flowering date to the present.

diameter decreases with exposure to high solar radiation in the daytime and increases with decreased exposure to solar radiation in the evening. In addition, the maximum daily stem-shrinkage of the stem diameter is susceptible to a vapor-pressure deficit [33]. All sensor data were collected every minute, and the daily averaged values were used for the model inputs. In addition, we measured the sugar content of the fruits of the target tomato plants on which the stem diameter sensors were installed. The sugar content was measured every three days by a hand-type near-infrared spectrometer (CD-H100, Chiyoda Corporation), and the measurements were conducted in each step of the three-step dense-planting hydroponic cultivation. We used the average sugar content value of 10 time measurements for each fruit in which stem diameter sensors were installed.

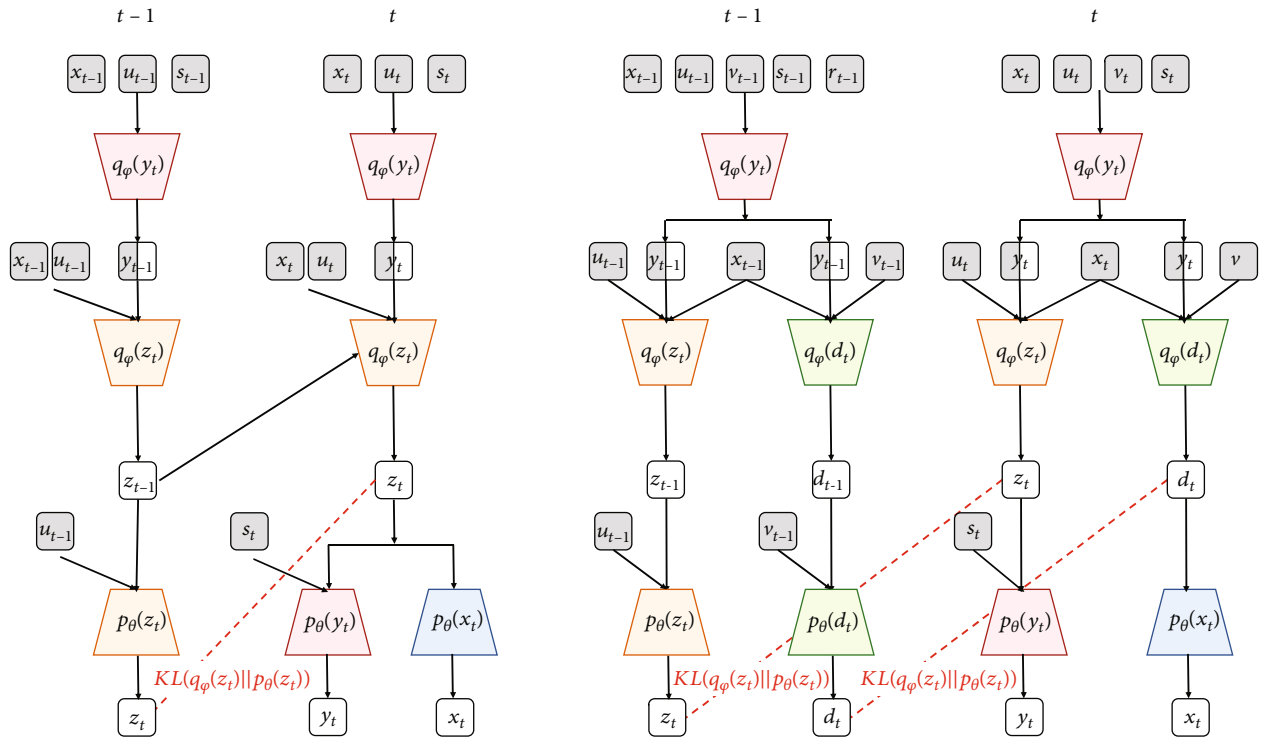
2.2.2. SDSSM Variable Settings. The three models (Cont-SDSSM, Disc-SDSSM, and 2L-SDSSM) used for the estimation of sugar content are analyzed through comparative evaluations. This is to reveal the performance of the proposed SDSSM. The variables for the proposed models are listed in Table 1. In this paper, each variable of the proposed three models is set as listed in Table 1; x_t is sugar content, y_t is stem diameter; vector u_t is temperature, solar radiation, vapor-pressure deficit (VPD), the elapsed date from flowering, and accumulated temperature (which is the total temperature from the flowering date to the present); and vector s_t is the CO_2 concentration, solar radiation, and step ID of three-step dense-planting hydroponic cultivation to indicate to which step of the tomato the data belong. The action variables r_t are not used in any models in this experiment. The settings of each variable in Disc-SDSSM are similar to those of Cont-SDSSM. The difference is that v_t includes the elapsed date from flowering and the accumulated temperature. 2L-SDSSM has similar settings to Cont-SDSSM, and the differences are that u_t includes temperature, solar radiation, and VPD, while v_t includes the elapsed date from flowering and the accumulated temperature.

An overview of the information flow at time step t in the graphical model (Figure 1) of the proposed method is shown in Figure 3 by using probability distributions. In addition, Figure 4 shows the inputs and outputs of each probability distribution for the neural network architectures. Cont-SDSSM and Disc-SDSSM include five types of neural networks corresponding to five types of probability distributions: $p_\theta(x_t)$, $p_\theta(y_t)$, $p_\theta(z_t)$, $q_\varphi(x_t)$, and $q_\varphi(z_t)$. 2L-SDSSM has seven types of neural networks corresponding to its seven types of probability distributions: $p_\theta(x_t)$, $p_\theta(y_t)$, $p_\theta(z_t)$, $p_\theta(d_t)$, $q_\varphi(x_t)$, $q_\varphi(z_t)$, and $q_\varphi(d_t)$.

These seven types of neural networks have the same basic architecture: a hidden layer converts the input nonlinearly, and then the outputs are converted to a mean vector and diagonal covariance log-parameterization matrix through a single separate hidden layer. The neural networks have hidden layers structured as follows: fully connected layers, rectified linear unit (ReLU) layers [34], and batch normalization layers [35]. The neural networks that emit latent states z_t or d_t use a long short-term memory (LSTM) [36, 37] as the first hidden layer instead of a fully connected layer. In this case, LSTM has a forget gate, input gate, and output gate, which makes it possible to consider long-term time series. Therefore, LSTM is used as the first hidden layer. All hidden layers have 128 units, and all weights are initialized using He et al.'s initialization [38] to accelerate convergence. All random variables of Cont-SDSSM follow a normal distribution. On the other hand, the random latent variable$d_t$is categorically distributed. This is because categorical distribution has only one parameter, π; Disc-SDSSM and 2L-SDSSM have two consistent hidden layers without the branch structure used for Cont-SDSSM.

2.2.3. Experimental Conditions. We verified the performance of the proposed methods through two types of evaluations. In the first evaluation, we compared semisupervised SDSSMs to supervised SDSSMs trained using only labeled data via supervised learning to verify the effectiveness of our semi-supervised learning approach. In this experiment, the supervised Cont-SDSSM, Disc-SDSSM, and 2L-SDSSM are called Cont-SV (SV denotes supervised), Disc-SV, and 2L-SV, respectively. In addition, the semisupervised Cont-SDSSM, Disc-SDSSM, and 2L-SDSSM are called Cont-SSV (SSV denotes semisupervised), Disc-SSV, and 2L-SSV, respectively.

In the second evaluation, we compared semisupervised SDSSMs with typical deep neural networks, the multilayer perceptron (MLP) and stacked LSTM (sLSTM). This is to investigate the performance of the proposed models for the optimal estimation of sugar content. We expected that the proposed models fit all of the observed data rather than local



(a) Cont-SDSSM and Disc-SDSSM

(b) 2L-SDSSM

Figure 3: Overview of information flow at time step t of the proposed models.

data because the KL terms in ELBO perform a regularization function. On the other hand, both MLP and sLSTM which were trained on a small dataset can easily result in overfitting. Therefore, we applied dropout [39] to each layer in the MLP and sLSTM to reduce overfitting.

We trained each model using the collected tomato dataset. We divided the dataset into training data, validation data, and test data to train and evaluate the models appropriately. In addition, to validating the robustness of the proposed methods, cross-validation was conducted using data sets of 16 cultivation beds divided into four patterns, as shown in Table 2. The hyperparameters were tuned by using random sampling. The hyperparameter such as learning rate, optimization algorithms, dimension size of latent variables for SDSSMs, sequence length, and dropout rate were the same for all the compared models.

When training SDSSMs, we gradually increased the weight coefficient for the KL terms in ELBO by 0.0001 after each epoch (starting from 0 to 1). The mean absolute error (MAE), root mean squared error (RMSE), relative absolute error (RAE), and relative squared error (RSE) were used as the error indicators. In this study, all the models were tuned using the validation data, and the models which showed the lowest MAE were selected as the best models with the most optimal hyperparameters. In this experiment, we implemented all source codes using Python. We used Chainer [40] to implement a deep neural network architecture and scikit-learn [41] to preprocess the dataset. This evaluation was performed on a PC with an Intel Core i7-5820K Processor, GeForce GTX 1080, and 64 GB of memory. The training process time depends largely on the values of the hyperparameters, the tuning method, and the number of epochs. For example, when we used the training data for pattern C in Table 2, which had the largest number of test labeled contents for training, it took approximately 4.6 s to complete 1 epoch, so it took approximately 1 h to create one model converge. Regarding the inference time, it took approximately 5.33 s to infer the test data in 2L-SDSSM when it was repeatedly measured 100 times using the same test data (pattern C in Table 2).

3. Results and Discussion

Figure 5 shows the average errors of each supervised SDSSM (Cont-SV, Disc-SV, and 2L-SV), the semisupervised SDSSMs (Cont-SSV, Disc-SSV, and 2L-SSV), MLP, and sLSTM for the four types of test data. The results demonstrate that all semisupervised SDSSMs reduced the estimation errors for all error indicators. In particular, the MAE of Cont-SV is 1.25, whereas that of Cont-SSV is 0.78; the MAE reduction rate of Cont-SSV versus Cont-SV is approximately 38%. Similarly, the MAE reduction rates of the semisupervised SDSSMs versus the supervised SDSSMs are approximately 30% for Disc-SSV and 9% for 2L-SSV. Moreover, both the RAE and RSE of all semisupervised SDSSMs are less than 1. Therefore, all semisupervised SDSSMs perform better than the naive models which output the average of the true values as the estimation values. The MAEs of typical deep neural networks, such as MLP and LSTM, are larger than those of Cont-SSV and less than those of Disc-SSV and 2L-SSV.

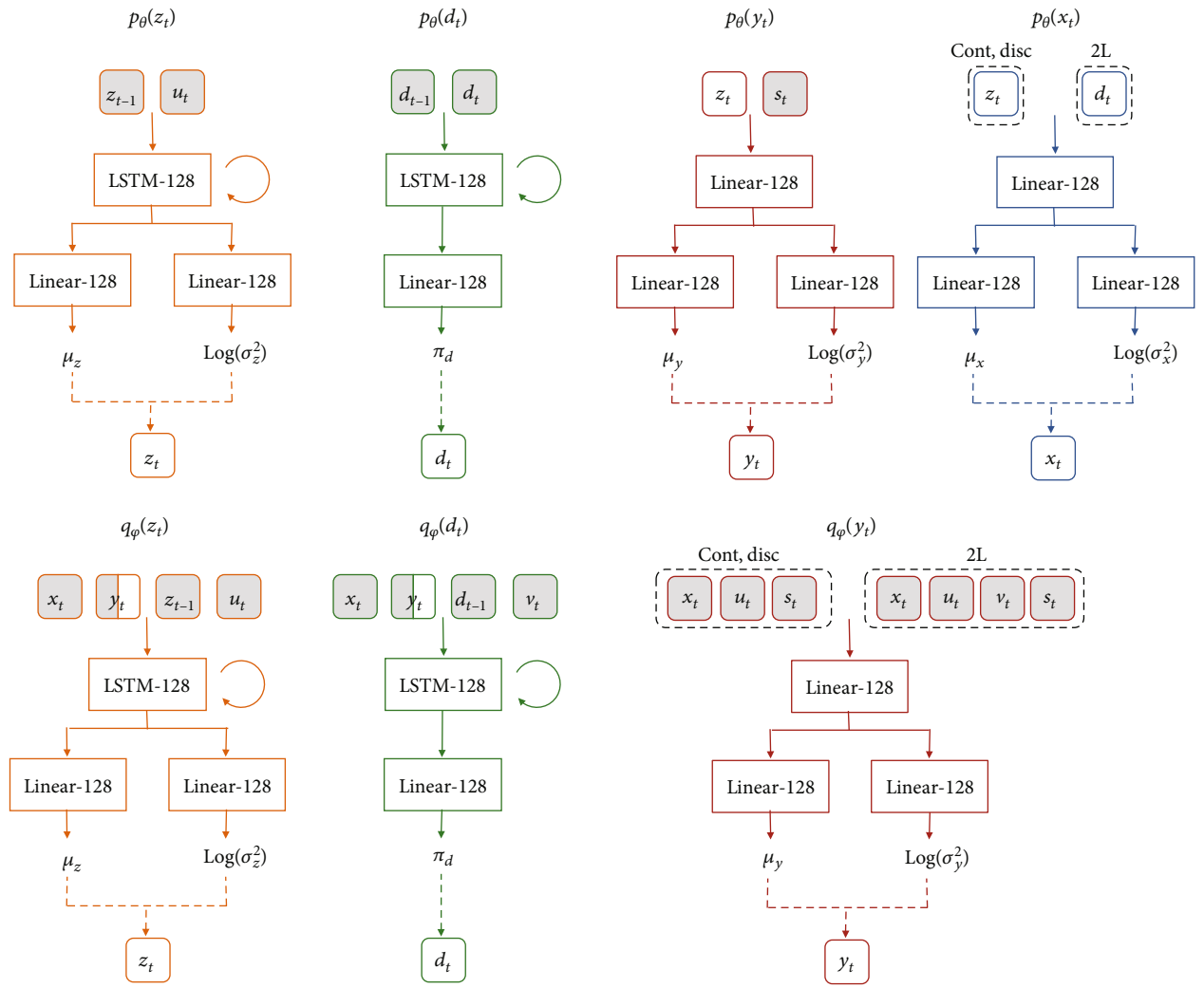


Figure 4: Network architectures showing each neural network in the proposed models.

Table 2: Data used for cross-validation.

Dataset pattern	Training (data size (labeled size) (cultivation bed no.))	Validation (data size (labeled size) (cultivation bed no.))	Test (data size (labeled size) (cultivation bed no.))
A	2,241 (382) (3, 4, 5, 6, 7, 11, 13, 14, 15)	747 (123) (8, 12, 16)	996 (167) (1, 2, 9, 10)
B	2,241 (345) (1, 5, 7, 8, 9, 13, 14, 15, 16)	747 (131) (2, 6, 10)	996 (154) (3, 4, 11, 12)
C	2,241 (361) (1, 2, 3, 4, 7, 9, 10, 11, 15)	747 (123) (8, 12, 16)	996 (189) (5, 6, 13, 14)
D	2,241 (381) (1, 3, 4, 5, 9, 11, 12, 13, 14)	747 (131) (2, 6, 10)	996 (161) (7, 8, 15, 16)

Although the error indicators of MLP and sLSTM appear to better than other models, further analysis is needed.

Figure 6 shows the true values, estimated values, and standard deviations of estimated values of time-series sugar content in the area 10 output by supervised SDSSMs, semisupervised SDSSMs, MLP, and sLSTM trained on dataset pattern A. The standard deviations of Cont-SDSSM and 2L-SDSSM are so low that they are not clearly visible. The results show that our semisupervised approach (Cont-SSV, Disc-SSV, and 2L-SSV) estimates the sugar contents better because of the improved generalization performance. Although there are few sugar content data before October (the other dataset patterns show a similar tendency because we collected sugar content data in the same manner over the entire area), the semisupervised SDSSMs clearly identify the variation patterns in other periods with their effective use of unlabeled data. From middle October to late October in Figure 6, although each supervised SDSSM (Cont-SV, Disc-SV, and 2L-SV) appears to be able to estimate better compared to the semisupervised SDSSMs, each supervised SDSSM's RMSE in Figure 5 is higher than that of each semisupervised SDSSM.

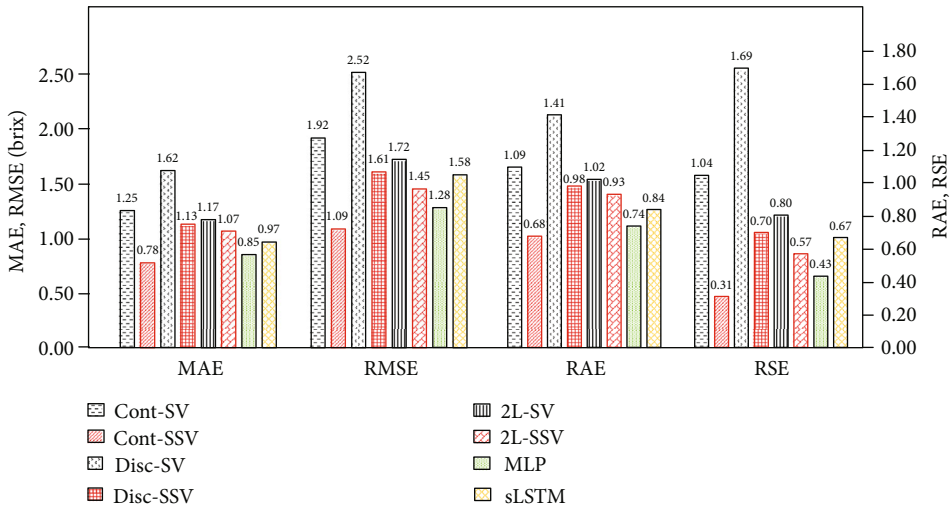


Figure 5: Error indicators of Cont-SV, Cont-SSV, Disc-SV, Disc-SSV, 2L-SV, 2L-SSV, MLP, and sLSTM.

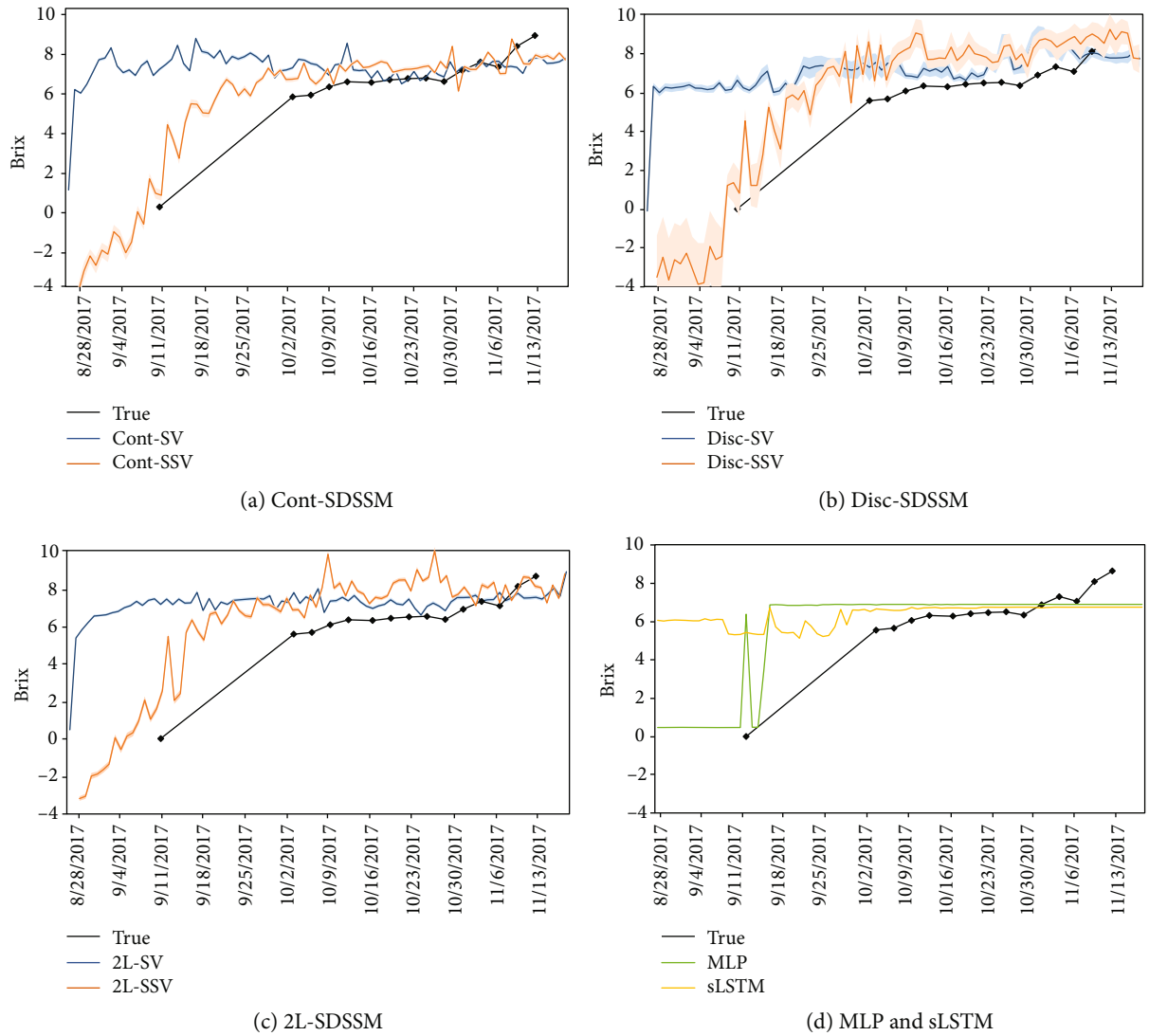


Figure 6: True and estimated values of the sugar content (brix) with the standard deviations for supervised SDSSMs, semisupervised SDSSMs, MLP, and sLSTM.

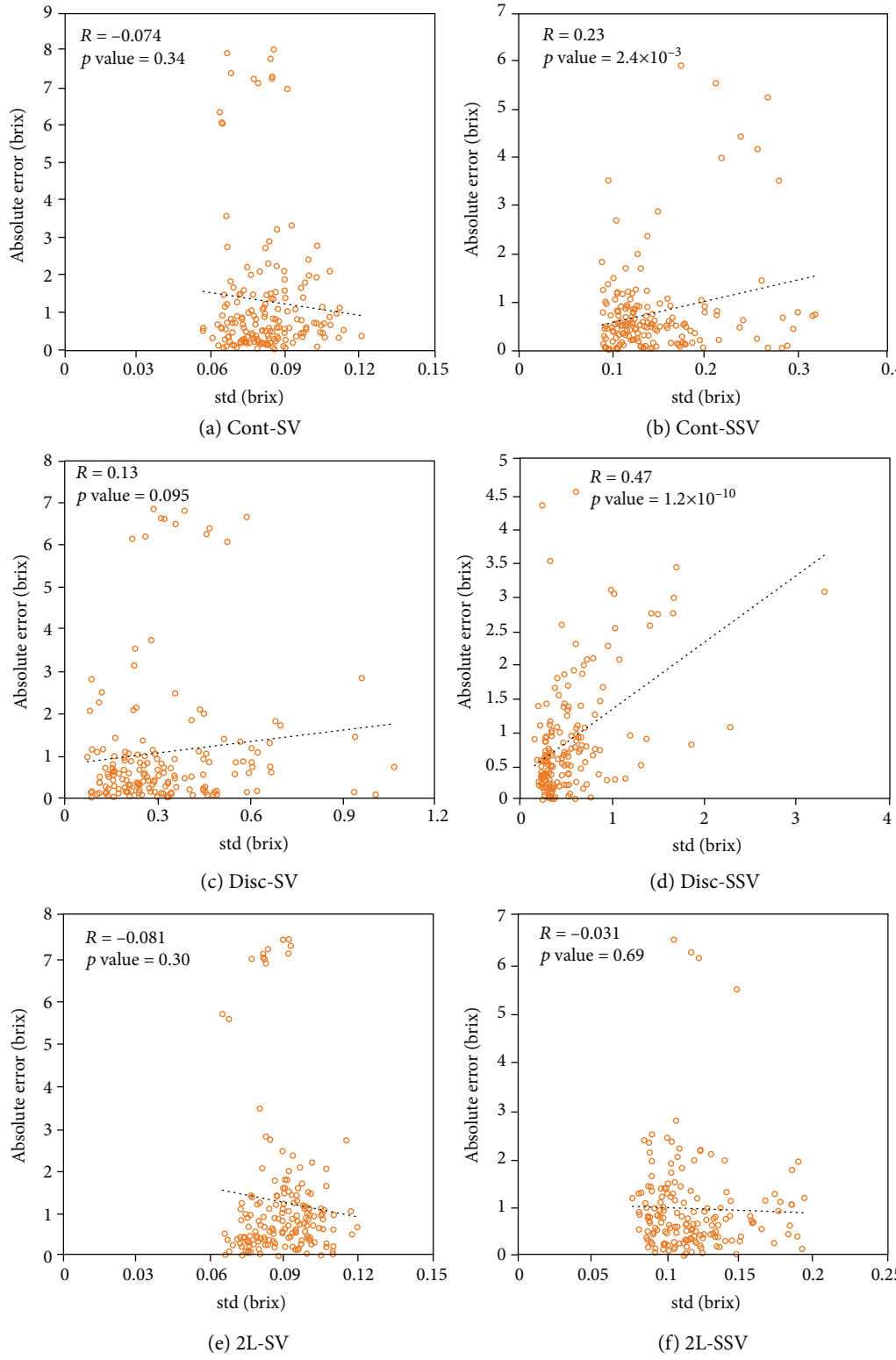


Figure 7: Scatter plots of standard deviations and absolute errors of supervised SDSSMs and semisupervised SDSSMs.

SDSSMs also output the variance and the estimation values. The averages of the standard deviations of the estimated values for all test plots in Table 2 are approximately 0.076 for Cont-SV, 0.28 for Disc-SV, 0.082 for 2L-SV, 0.17 for Cont-SSV, 1.04 for Disc-SSV, and 0.12 for 2L-SSV. In particular, the standard deviation of Disc-SSV is larger than those of the others, as clearly shown in Figure 6. A large standard deviation indicates output instability in the estimation for sugar content. The instability, however, is not a significant problem in a model for model-based RL when the standard

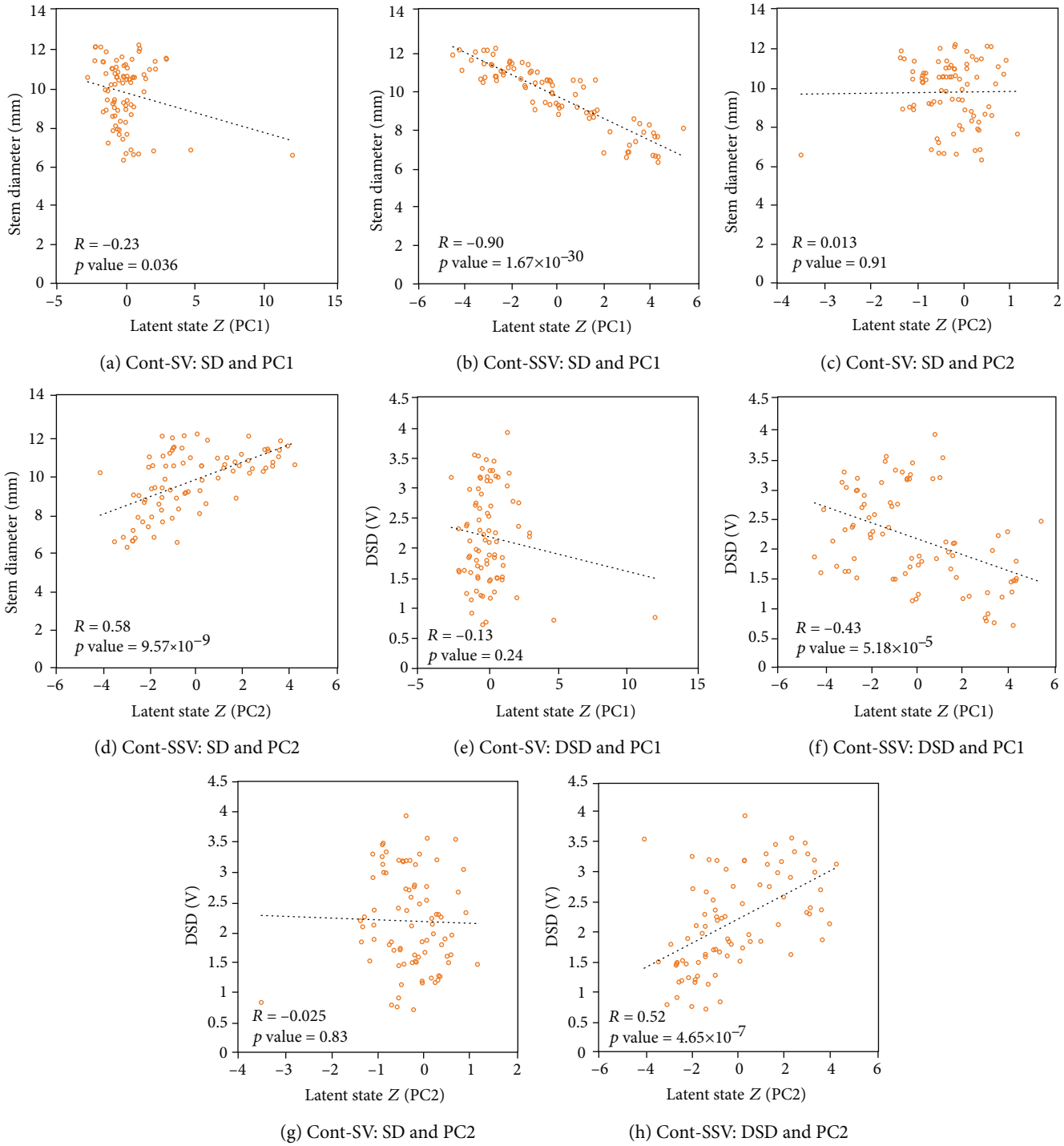


Figure 8: Scatter plots showing principal components and stem diameter or DSD.

deviation and estimation error are positively correlated. This is because the agent of the model-based RL explores an environmental model while considering the uncertainty of the estimation based on the standard deviation of the estimation value. Therefore, estimating the uncertainty correctly allows the agent to learn efficiently. MLP and LSTM look like they produce the same tendency as the supervised SDSSMS, and the estimated values in the period during October are relatively close to the true values, resulting in small errors, as shown in Figure 6(d). Considering only the numerical values, the MLP and LSTM look like they produce higher estimates than the supervised SDSSMs. However, as can be seen from Figure 6(d), the MLP and LSTM are overfitted to a particular dataset in the period after October even though dropout was applied in both cases, and the generalization performance is low.

Figure 7 shows scatter plots of the standard deviations and the absolute errors for the compared models with the test

dataset of pattern A in Table 1. Our proposed method is based on a generative model, which outputs both estimation values and standard deviations. Therefore, our proposed methods can evaluate the uncertainty of estimates by using the standard deviation. These results indicate that Cont-SSV and Disc-SSV significantly improve the correlation coefficient compared to supervised SDSSMs such as Cont-SV and Disc-SV. The correlation coefficient of 2L-SSV is still negative and is likely to cause incorrect exploration, although 2L-SSV slightly improves the correlation coefficient compared to 2L-SV. Conceivably, the significantly high correlation of 0.47 for Disc-SSV demonstrates that its standard deviations can assist agents to better seek an appropriate model and promote learning of the optimal control to achieve high sugar content.

Figure 8 shows the principal component and stem diameter or the difference in stem diameter (DSD) in the model learned using the dataset of pattern A in Table 1 as a scatter diagram on the x-axis and y-axis, respectively. The DSD is one of the water stress indicators and is expressed as the difference between the maximum stem diameter (SD) observed thus far and the current stem diameter (SD_i) as follows: $DSD_i = \max(SD_0, \cdots, SD_i) - SD_i$. The maximum value continuously updates with plant growth. By calculating the decrease from the maximum stem diameter, the variation due to plant growth is ignored, and only the amount of water stress can be quantified from the stem diameter. This figure demonstrates that each principal component of the semisupervised Cont-SSV has a higher correlation with the stem diameter and DSD compared with Cont-SV. In particular, in Cont-SSV, the stem diameter has a significantly high correlation of approximately -0.9 with the first component, as shown in Figure 8(b), and DSD has a significantly high correlation of approximately 0.52 with the second component, as shown in Figure 8(h). This result suggests that Cont-SSV represents both plant growth and plant water content in the latent space owing to the reasonable inference achieved by using two observation variables sharing latent variables in our semisupervised learning model.

Additionally, the latent space is represented as a linear combination of these two plant states. This result confirms the assumption that the two types of latent variables are independent of each other. Cont-SSV and Disc-SSV have different natures, and Cont-SSV has better estimation accuracy, but Disc-SSV estimates the uncertainty better.

The results indicate that our three types of proposed models (Cont-SSV, Disc-SSV, and 2L-SSV) work better than the same models with supervised learning and other typical deep neural networks. In particular, Cont-SSV has good potential to estimate sugar content with high accuracy and valid uncertainty. Considering the appropriate representation of the latent states, it is believed that Cont-SSV will perform well as an environmental model of model-based RL for the optimal control of sugar content.

4. Conclusion

We have proposed a novel plant growth model using a semisupervised deep state-space model (SDSSM) for model-based reinforcement learning to determine the optimal control of sugar content. There have been several studies on tomato growth modeling [42, 43], but we could not find any similar study for modeling time-series tomato sugar content. SDSSM is a sequential deep generative model that uses structured variational inference to model the slow dynamics of living organisms (such as plant growth). In particular, SDSSM was trained using our semisupervised learning method that complementarily infers the latent states by introducing two observation variables to efficiently utilize sugar content data which is difficult to collect.

Additionally, we designed three types of SDSSMs under different assumptions regarding each latent space. The experimental results demonstrated that the introduction of two observation variables sharing latent variables improved the generalization performance and enabled all SDSSMs to track the variation of sugar content appropriately. Moreover, tomatoes grown during the experiment had a maximum brix rating of 10.73 and minimum brix rating of 4.67. The average brix rating was 6.81. The highest accuracy model is 0.78 in MAE; thus, our model has a potential to estimate time-series sugar content variation with high accuracy.

We have designed a combined model (2L-SDSSM); however, the combined model was not the highest accuracy model. Therefore, we still need to consider other ways to combine the two models more appropriately, i.e., assuming the independence of two latent states. In a future study, we intend to improve the 2L-SDSSM which is the combination of two different latent variables. Furthermore, we will improve time-series data (sensor data of the temperature, humidity, solar radiation, CO_2 concentration, stem diameter, and plant growth) in a greenhouse different from that used in this study. We will continue to verify the performance of our model by comparing our model with typical machine learning and typical deep neural networks.

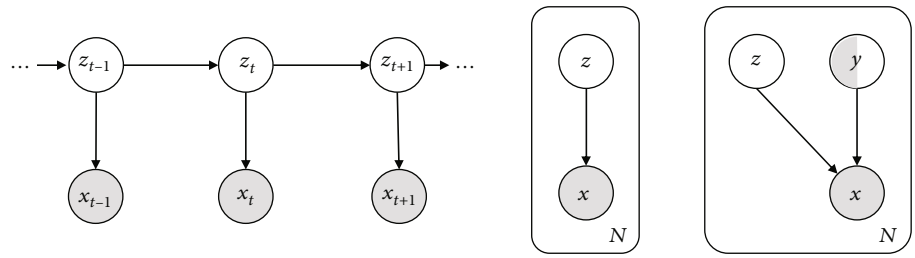
Appendix

A. Plant Growth Model Design

A.1 State-Space Model. The state-space model (SSM) is a generative model that represents time series based on two types of models: one is the system model and the other is the observation model. SSM models a sequence of observed variables $x_1, x_2, \cdots, x_T$ and a corresponding sequence of latent variables $z_1, z_2, \cdots, z_T$ at each time step t ($t = 1, 2, \cdots, T$). The probabilistic model is shown in Figure 9(a), and a general expression of the model is

$$\begin{aligned} z_t &\sim p(z_t \mid z_{t-1}, \theta), \text{(system model)}, \\ x_t &\sim p(x_t \mid z_t, \varphi), \text{(observation model)}, \end{aligned} \tag{A.1}$$

where the system model is the conditional probability distribution of the latent state z_t conditioned on the previous latent state z_{t-1} and the observation model is the conditional probability distribution of x_t conditioned on the corresponding latent state z_t at time step t. The variables θ and φ denote the parameter vectors of the system model and the observation model, respectively. The system model is initialized by



(a) General SSM (b) VAE (c) CVAE

Figure 9: Graphical models of SSM, VAE, and CVAE.

$z_0 = p(z_0)$. We use available arbitrary probability distributions in both the system model and the observation model. Some SSMs with specific distributions and constraints have unique names. For example, one of the simplest models, called a linear Gaussian model (LGM) [44], can be written mathematically as

$$\begin{aligned} z_{t+1} &= A_t z_t + \epsilon_t, \epsilon_t \sim N(0, Q), \text{(system model)}, \\ x_t &= B_t z_t + \omega_t, \omega_t \sim N(0, R), \text{(observation model)}, \end{aligned} \tag{A.2}$$

where vector ϵ_t and vector ω_t are random variables representing the state and observation noises, respectively. In addition, x_t and z_t are vectors. Both of these noises are independent of each other, the corresponding latent state z_t, and the conditional probability distribution of x_t. Both of these noise sources are a Gaussian distribution with zero covariance matrix representing the mean Q and R, each independent of the time step. A_t and B_t denote coefficient matrices. LGMs can fit with time-series data and are utilized for many applications whose observations and latent states have a linear transition and a normal distribution, respectively.

Additionally, there are methods available to model time-series data with nonlinear transitions, e.g., the extended Kalman filter [45] and the quadrature particle filter [46]. Raiko et al. [47] and Valpola and Karhunen [22] attempted to estimate an intractable posterior of the complex generative model using nonlinear dynamic factor analysis, which is impractical for large-scale datasets owing to the inherent quadratic scale regarding observed dimensions. Recently, Kingma and Welling [29] introduced stochastic gradient variational Bayes (SGVB) and a learning algorithm named a variational autoencoder (VAE) to obtain a tractable posterior.

A.2 Variational Autoencoder. The variational autoencoder (VAE) [22] is a deep generative model for nonsequential data. The parameters are optimized via SGVB [23], and the probabilistic model is shown in Figure 9(b). In particular, conditional VAE (CVAE), as shown in Figure 9(c), is a typical deep generative model trained by semisupervised learning [21]. The N in the plate (the part surrounded by the frame) shown in Figure 9 means that N nodes are omitted and only the representative nodes are shown in Figures 9(b) and 9(c). The details will be discussed later in this section. In the VAE, the generative process can be written mathematically as

$$\begin{aligned} z &\sim p(z), \\ x &\sim p_\theta(x \mid z). \end{aligned} \tag{A.3}$$

The latent variable z is obtained from a prior $p(z)$. The observed variable x is drawn from a probability distribution $p_\theta(x \mid z)$ conditioned on z with the parameter vector θ, and the posterior $p_\theta(z \mid x)$ is assumed to be intractable. The parameter is estimated simultaneously with the latent states through maximization of the following marginal log-likelihood:

$$\text{Log} p_\theta(x) = \log \int p_\theta(x \mid z) p(z) dz. \tag{A.4}$$

When the posterior is intractable, the standard expectation-maximization (EM) algorithm and variational inference do not work well because the EM algorithm needs to compute the posterior and variational inference requires closed-form solutions of the expectations of the joint probability density function. Additionally, sampling-based EM algorithms require sufficient time to obtain the posterior when the data size is large. In SGVB, the approximate posterior $q_\varphi(z \mid x)$ with parameter φ is introduced, and we consider the following evidence lower bound (ELBO) $\mathscr{L}(x; \theta, \varphi)$ as would be the case for the variational inference. ELBO is an objective function used to optimize the parameters θ and φ.

$$\mathscr{L}(x; \theta, \varphi) = E_{q_\varphi(z|x)}[\log p_\theta(x \mid z)] - \text{KL}\left(q_\varphi(z \mid x) \| p(z)\right). \tag{A.5}$$

KL denotes a Kullback–Leibler divergence. In addition, parameter φ denotes the parameter of $q_\varphi(z \mid x)$ that approximates $p_\theta(x)$. In fact, in the VAE, φ represents the neural network's weights and bias, and these parameters are optimized via back-propagation. The difference in variational inference is the use of differentiable Monte Carlo

expectations instead of applying the mean-field assumption. Formally, the reformulation of the ELBO is as follows:

$$\mathscr{L}(x;\theta,\varphi) \cong \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\left(x \mid z^{(l)}\right) - \mathrm{KL}\left(q_\varphi(z \mid x) \| p(z)\right), \tag{A.6}$$

where z is sampled via a reparameterization trick ($z^{(l)} = \mu + \sigma\varepsilon^{(l)}$ and $\epsilon^{(l)} \sim N(0, I)$) to acquire a differentiable ELBO instead of sampling from the posterior $q_\varphi(z \mid x)$ (which is not differentiable with respect to φ). Specifically, the reparameterization trick represents a sampling $z \sim q_\varphi(z \mid x)$ as a deterministic transformation $g_\varphi(\epsilon, x)$ by adding a random noise $\epsilon \sim p(\epsilon)$ to input x. By using the reparameterization trick, the expectation term can be written mathematically as $E_{q_\varphi(z|x)}[f(z)] \cong 1/L\sum_{l=1}^{L} f(g_\varphi(\epsilon^{(l)}, x))$ where $f(z) = \log p_\theta(x \mid z)$ and $\epsilon^{(l)} \sim p(\epsilon)$. This expectation is differentiable with respect to θ and φ. For example, when a latent variable z is distributed according to a Gaussian distribution $N(z \mid \mu, \sigma^2)$, the ELBO is as follows:

$$E_{q_\varphi(z|x)}[\log p_\theta(x \mid z)] \cong \frac{1}{L}\sum_{l=1}^{L} \log p_\theta\left(x \mid z^{(l)}\right). \tag{A.7}$$

The likelihood $p_\theta(x \mid z)$ and posterior $q_\varphi(z \mid x)$ are represented by neural networks, and the parameters are optimized via back-propagation.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Authors' Contributions

S. Shibata conceived the idea. S. Shibata and R. Mizuno designed the experimental procedures in detail, evaluated the idea, and preformed data analysis. S. Shibata, R. Mizuno, and H. Mineno provided important help during the experimental section and writing of this paper. H. Mineno directed the research. All of the authors participated in the preparation of the manuscript.

Acknowledgments

This work was supported by JST PRESTO Grant Number JPMJPR15O5, Japan. Additionally, we greatly appreciate the support of Mr. Maejima and Mr. Imahara who provided the data collection location, Shizuoka Prefectural Research Institute of Agriculture and Forestry.

References

[1] X. Wang, Z. Meng, X. Chang, Z. Deng, Y. Li, and M. Lv, "Determination of a suitable indicator of tomato water content based on stem diameter variation," *Scientia Horticulturae*, vol. 215, pp. 142–148, 2017.

[2] M. Sano, Y. Nakagawa, T. Sugimoto et al., "Estimation of water stress of plant by vibration measurement of leaf using acoustic radiation force," *Acoustical Science and Technology*, vol. 36, no. 3, pp. 248–253, 2015.

[3] J. A. Sánchez-Molina, F. Rodríguez, J. L. Guzmán, and J. A. Ramírez-Arias, "Water content virtual sensor for tomatoes in coconut coir substrate for irrigation control design," *Agricultural Water Management*, vol. 151, pp. 114–125, 2015.

[4] R. Prasad, K. R. Ranjan, and A. K. Sinha, "AMRAPALIKA: an expert system for the diagnosis of pests, diseases, and disorders in Indian mango," *Knowledge-Based Systems*, vol. 19, no. 1, pp. 9–21, 2006.

[5] L. A. Richards and W. Gardner, "Tensiometers for measuring the capillary tension of soil water," *Journal of the American Society of Agronomy*, vol. 28, no. 1, pp. 352–358, 1936.

[6] G. C. Topp, J. L. Davis, and A. P. Annan, "Electromagnetic determination of soil water content: measurements in coaxial transmission lines," *Water Resources Research*, vol. 16, no. 3, pp. 574–582, 1980.

[7] C. Patanè and S. L. Cosentino, "Effects of soil water deficit on yield and quality of processing tomato under a Mediterranean climate," *Agricultural Water Management*, vol. 97, no. 1, pp. 131–138, 2010.

[8] M. F. Othman and K. Shazali, "Wireless sensor network applications: a study in environment monitoring system," *Procedia Engineering*, vol. 41, pp. 1204–1210, 2012.

[9] D. H. Park and J. W. Park, "Wireless sensor network-based greenhouse environment monitoring and automatic control system for dew condensation prevention," *Sensors*, vol. 11, no. 4, pp. 3640–3651, 2011.

[10] H. Ibayashi, Y. Kaneda, J. Imahara, N. Oishi, M. Kuroda, and H. Mineno, "A reliable wireless control system for tomato hydroponics," *Sensors*, vol. 16, no. 5, p. 644, 2016.

[11] S. Shibata, Y. Kaneda, and H. Mineno, "Motion-specialized deep convolutional descriptor for plant water stress estimation," in *Engineering Applications of Neural Networks*, G. Boracchi, L. Iliadis, C. Jayne, and A. Likas, Eds., vol. 744 of EANN 2017. Communications in Computer and Information Science, pp. 3–14, Springer, Cham, 2017.

[12] Y. Kaneda, S. Shibata, and H. Mineno, "Multi-modal sliding window-based support vector regression for predicting plant water stress," *Knowledge-Based Systems*, vol. 134, pp. 135–148, 2017.

[13] F. Hernández-del-Olmo, E. Gaudioso, R. Dormido, and N. Duro, "Tackling the start-up of a reinforcement learning agent for the control of wastewater treatment plants," *Knowledge-Based Systems*, vol. 144, pp. 9–15, 2017.

[14] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics," *Foundations and Trends in Robotics*, vol. 2, no. 1-2, pp. 1–142, 2013.

[15] X. Zhang, T. Yu, B. Yang, and L. Cheng, "Accelerating bio-inspired optimizer with transfer reinforcement learning for reactive power optimization," *Knowledge-Based Systems*, vol. 116, pp. 26–38, 2017.

[16] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, vol. 16, pp. 2094–2100, Phoenix, USA, 2016.

[17] V. Mnih, A. P. Badia, M. Mirza et al., "Asynchronous methods for deep reinforcement learning," in *International Conference*

on Machine Learning, pp. 1928–1937, New York City, USA, 2016.

[18] M. Deisenroth and C. E. Rasmussen, "PILCO: a model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Con-ference on Machine Learning*, pp. 465–472, Bellevue, WA, USA, 2011.

[19] P. Poupart and N. Vlassis, "Model-based Bayesian reinforcement learning in partially observable domains," in *Proceedings of the International Symposium on Artificial Intelligence and Mathematics*, pp. 1-2, Fort Lauderdale, USA, 2008.

[20] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," *Pattern Recognition Letters*, vol. 42, pp. 11–24, 2014.

[21] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Advances in neural information processing systems*, pp. 3581–3589, Montreal, Canada, 2014.

[22] H. Valpola and J. Karhunen, "An unsupervised ensemble learning method for nonlinear dynamic state-space models," *Neural Computation*, vol. 14, no. 11, pp. 2647–2692, 2002.

[23] J. Bayer and C. Osendorfer, "Learning stochastic recurrent networks," in *Proceedings of Workshop on Advances in Variational Inference*, Montreal, Canada, 2014.

[24] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2962–2970, Montreal, Canada, 2015.

[25] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: a locally linear latent dynamics model for control from raw images," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2746–2754, Montreal, Canada, 2015.

[26] R. G. Krishnan, U. Shalit, and D. Sontag, "Deep Kalman Filters," https://arxiv.org/abs/1511.05121.

[27] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther, "Auxiliary deep generative models," https://arxiv.org/abs/1602.05473.

[28] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proceedings of the Advances in Neural Information Processing*, pp. 2234–2242, Barcelona Spain, 2016.

[29] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," https://arxiv.org/abs/1312.6114.

[30] M. Fraccaro, S. K. Sønderby, U. Paquet, and O. Winther, "Sequential neural models with stochastic layers," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 2199–2207, Barcelona Spain, 2016.

[31] U. Meier, "Growth stages of mono-and dicotyledonous plants: BBCH-monograph," http://pub.jki.bund.de/index.php/BBCH/issue/view/161, 2001.

[32] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," https://arxiv.org/abs/1611.01144.

[33] D. Zhang, Q. Du, Z. Zhang, X. Jiao, X. Song, and J. Li, "Vapour pressure deficit control in relation to water transport and water productivity in greenhouse tomato production during summer," *Science Reports*, vol. 7, no. 1, article 43461, 2017.

[34] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814, Haifa, Israel, 2010.

[35] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," https://arxiv.org/abs/1502.03167.

[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 1999.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on imagenet classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, Santiago, Chile, 2015.

[39] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[40] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a next-generation open source framework for deep learning," in *Proceedings of the Workshop on Machine Learning Systems in the Advances in Neural Information Processing Systems*, pp. 1–6, Montreal, Canada, 2015.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort et al., "Scikit-learn: machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] L. Sun, Y. Yang, J. Hu, D. Porter, T. Marek, and C. Hillyer, "Respiration climacteric in tomato fruits elucidated by constraint-based modeling," *ISPA/IUCC*, vol. 213, pp. 1726–1739, 2017.

[43] M. Kang and F.-Y. Wang, "From parallel plants to smart plants: intelligent control and management for plant growth," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, pp. 161–166, 2017.

[44] S. Roweis and Z. Ghahramani, "A unifying review of linear Gaussian models," *Neural Computation*, vol. 11, no. 2, pp. 305–345, 1999.

[45] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*, Courier Corporation, 2007.

[46] L. Liang-qun, X. Wei-xin, and L. Zong-xiang, "A novel quadrature particle filtering based on fuzzy c-means clustering," *Knowledge-Based Systems*, vol. 106, pp. 105–115, 2016.

[47] T. Raiko, M. Tornio, A. Honkela, and J. Karhunen, "State inference in variational Bayesian nonlinear state-space models," in *Independent Component Analysis and Blind Signal Separation*, vol. 3889 of ICA 2006. Lecture Notes in Computer Science, pp. 222–229, Springer, Berlin, Heidelberg, 2006.