Learning Ensembles of Continuous Bayesian Networks: An Application to Rainfall Prediction

Scott Hellman
School of Computer Science
University of Oklahoma
Norman, Oklahoma
shellman@ou.edu

Amy McGovern
School of Computer Science
University of Oklahoma
Norman, Oklahoma
amcgovern@ou.edu

Ming Xue
Center for the Analysis and Prediction of Storms
School of Meteorology
University of Oklahoma
Norman, Oklahoma
mxue@ou.edu

***Abstract*—We introduce Ensembled Continuous Bayesian Networks (ECBN), an ensemble approach to learning salient dependence relationships and to predicting values for continuous data. By training individual Bayesian networks on both a subset of the data (bagging) and a subset of the attributes in the data (randomization), ECBN produces models for continuous domains that can be used to identify important variables in a dataset and to identify relationships between those variables. We use linear Gaussian distributions within our ensembles, providing efficient network-level inference. By ensembling these networks, we are able to represent nonlinear relationships. We empirically demonstrate that ECBN outperforms the meteorological forecast on a rainfall prediction task across the United States, and performs comparably to results reported for Random Forests.**

I. Introduction

We introduce Ensembled Continuous Bayesian Networks (ECBN), an ensemble approach for prediction of continuous variables using Bayesian networks. This work is motivated by our research in several earth-science domains, including tornadoes, drought, and floods [1]. We work with domain scientists who want to improve the prediction of such tasks while also increasing their understanding of the domain. The ECBN method aims to fit both of these goals.

Bayes nets are graphical models that represent the joint probability distribution of a dataset in a compact way. This is done by identifying conditional independencies in the data. By ensembling such networks, we aim to increase predictive strength compared to a single network's abilities.

We apply ECBN to the task of improving the prediction of rainfall across the United States. We use the 2010 Storm Scale Ensemble Forecast (SSEF) data from the Center for the Analysis and Prediction of Storms (CAPS) [2] [3]. In particular, we use the forecasts of 14 meteorological models within the SSEF to predict the actual rainfall. By further refining these model outputs using machine learning techniques, we can account for disagreement between models and reduce the bias in the final prediction.

ECBN provides a novel and powerful prediction method that gracefully handles missing values and that allows domain scientists to analyze how continuous variables within the domain interact. We empirically demonstrate that ECBN performs well at rainfall prediction, and demonstrate how ECBN can be used to investigate the predictive importance of SSEF variables and the dependence relationships between those variables.

II. Related Work

Conventional Bayesian Networks have proven useful in meteorological domains. Cofino et al. apply Bayes nets to rainfall prediction in the Iberian peninsula, treating the amount of rainfall at a given location as a single vertex in their network [4]. Similar work has been done for temperature prediction over a geographical area [5]. Additionally, Kennet et al. use Bayes nets to predict sea breezes based off of various meteorological readings [6].

For continuous networks, Linear Gaussian Networks (LGNs) provide a method for representing continuous probability distributions within a Bayes net [7]. Further extensions of Bayes nets to continuous variables have been developed, such as representing local probability distributions as Gaussian mixture models [8]. Another possible approach is to use kernel density estimation to estimate the probability distribution, which requires storing all training data [9] [10].

Feng et al. present an approach for combining bagging, an ensemble method for randomly sampling data to train on, and Bayesian Networks, but their method ultimately results in a single network [11]. Boosting, an ensemble approach that iteratively trains models on weighted training data, has been applied to networks with a Naive Bayes structure. [12]. And Utz combined bagging and randomization, which trains each model on a subset of variables, to learn ensembles of discrete Bayesian networks [13]. Our approach differs from the above in that we model continuous variables in our networks.

Another approach to ensembling Bayesian Networks is the Bayesian Multinet [14]. Multinets learn a single network for different partitions of the class label, such that each network corresponds to a disjoint subset of the possible class values. In contrast, conventional ensembling represents every class value within every network.

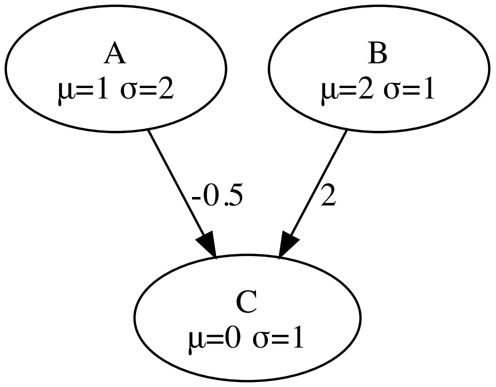

Figure 1. An example Bayesian Network. This network is a Linear Gaussian Network with values $\mu_A = 1$, $\mu_B = 2$, $\mu_C = 0$, $\sigma_A = 2$, $\sigma_B = 1$, $\sigma_C = 1$, $w_{AC} = -0.5$, and $w_{BC} = 2$.

III. Ensembled Continuous Bayesian Networks

What follows is a brief description of our methodology. For a more thorough explanation, see Hellman [15].

A. Notation

We use capital letters to refer to random variables or sets of random variables, such as A or X. We use bold to distinguish vectors and matrices from scalars, and for a random variable A, $\mathbf{A}$ denotes a vector of values sampled from A, while a denotes a single value sampled from A. We refer to the unconditional mean of A as μ_A, and to the standard deviation of A as σ_A. For a set of random variables X, $\mathbf{\Sigma_X}$ denotes the covariance of X. For any vector or matrix, $\mathbf{A_i}$ refers to the i-th value or row in the vector or matrix.

We represent a Gaussian distribution with mean μ and standard deviation σ as $N(\mu, \sigma)$. The value of the Gaussian distribution with mean μ and standard deviation σ at a point x is represented by $N(x; \mu, \sigma)$.

B. Linear Gaussian Networks

Bayesian Networks represent a joint probability distribution as a directed acyclic graph (DAG) [16]. Each vertex in the graph corresponds to a random variable, and each edge in the graph corresponds to a dependence relationship between two variables. Both the underlying structure of a Bayes net and the parameters for the conditional probability distributions at each vertex can be learned from training data. Learning the structure of a Bayes net can help uncover important relationships within a dataset, and with learned parameters, Bayes nets be used for prediction.

Since we are interested in continuous data, we use Linear Gaussian Networks (LGNs) in our ensembles [7]. In a LGN, each vertex represents a univariate Gaussian distribution with a set standard deviation and a mean that is a linear combination of an unconditional mean and the values sampled from the parent vertices.

For example, given a LGN of the form $A \rightarrow C \leftarrow B$ (Figure 1), the following describes the conditional probability distributions of the three vertices

$$\begin{aligned} A &\sim N(\mu_A, \sigma_A) \\ B &\sim N(\mu_B, \sigma_B) \\ C &\sim N(\mu_C + w_{AC}(a - \mu_A) + w_{BC}(b - \mu_B), \sigma_C) \end{aligned}$$

where w_{AC} is the weight of the edge between A and C.

Finding the optimal Bayesian Network structure for a dataset is NP-complete [17]. This forces the use of heuristics, of which there are two main approaches. The first approach is search and score, which searches through the set of possible structures and scores them according to some metric, eventually choosing the highest scoring model found during the search (e.g. [18]). The second approach is constraint identification, which uses statistical independence tests to identify edges to place in the final graph (e.g. [19]). To learn the structure of our LGNs, we use a hybrid constraint and search based approach (e.g. [20]). Such approaches build a skeleton network that is refined with local search to generate the final network. The skeleton network contains edges between all vertices that cannot be shown to be conditionally independent at an α confidence level.

During the constraint identification phase, the order in which the conditional independence tests are performed affects the structure of the skeleton. To order our tests, we use the Fast Adjacency Search algorithm [21]. For our conditional independence tests, we use the Fisher z-transformation of the partial correlation coefficient.

After generating a skeleton, we create our final network by using hill-climbing to orient and select edges. At each step of the search, our moves are: remove an edge in the final graph, add an edge that is in the skeleton to the final graph, or reverse an edge in the final graph if the corresponding edge in the skeleton is undirected. The search continues until no moves result in a higher scoring network. We make the assumption that the underlying structure of our data is multivariate normal so that we can use BGe to score our networks [7]. We expect to violate this assumption on most datasets, and rely on ensembling to increase predictive strength.

The BGe score of a network with structure S (denoted B_S) is

$$BGe(B_S) = p(B_S)p(D|B_S) \tag{1}$$

which, for a given dataset, is proportional to the likelihood of the network given the training data. We use a uniform prior for our prior probability $p(B_S)$.

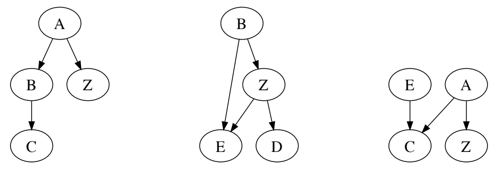

Figure 2. An ensemble of size 3 with 4 attributes per model. The possible attributes are A,B,C,D,E, and Z. Z is the target variable and is in every ensemble.

Once we have the structure of our network, we still need to estimate the parameters (mean, variance, and weights) for the linear Gaussians at each vertex. Parameter estimation of LGNs is done by linear least-squares regression [22]. Given a network structure and a set of training data, we traverse the graph in topological order. For each vertex visited, the effects of all parent vertices are subtracted out via least-squares regression, and then the mean and variance are computed from the residuals.

With both the structure and the parameters, we are able to perform inference to predict values given a test example. In general, exact inference in Bayesian networks is hard [23]. Fortunately, exact inference in LGNs is tractable, because a LGN is a multivariate Gaussian distribution [24].

C. Ensembled Continuous Bayesian Networks

A single LGN can only represent linear relationships. Ensembling can be used to increase the robustness and predictive strength as well as to represent non-linear relationships. Our approach, Ensembled Continuous Bayesian Networks, is an ensemble of LGNs that can model non-linear relationships. Similar to Utz, we draw inspiration from Random Forests and improve ECBN's performance by using bagging and randomization [13] [25].

Bagging increases diversity in an ensemble by providing a slightly different data set for each ensemble member. Each member is trained on a bootstrap resampled set of data drawn from the original training data. Randomization also increases ensemble diversity by limiting the subset of variables used by each model. ECBNs guarantee that the class variable is in each component. Figure 2 shows a small ensemble built with randomization. Combining bagging and randomization creates more inter-model variance than conventional ensemble learning, allowing for a more effective ensemble [25].

Given an ensemble of LGNs, how do we combine individual predictions in such a way that we can represent nonlinear relationships? Each LGN can only represent linear relationships, so a conventional linear combination of the predictions will not help. A key insight is that, since each component of our ensemble is a multivariate Gaussian, our whole ensemble is a Gaussian mixture model (GMM).

Given that our ensemble is a GMM, we can utilize Gaussian Mixture Regression (GMR) [26]. GMR is similar to the aforementioned linear combination of the network predictions, but each prediction is weighted by the likelihood of that network given the test example. As the likelihood is nonlinear, we have a nonlinear weighting, and are able to represent nonlinear relationships.

More specifically, to perform regression we first need to find component weights w_i of the form:

$$w_i(\mathbf{x}) = \frac{\pi_i N(\mathbf{x}; \mu_{ix}, \mathbf{\Sigma}_{ix})}{\sum_{j=1}^{k} \pi_j N(\mathbf{x}; \mu_{jx}, \mathbf{\Sigma}_{jx})} \tag{2}$$

In the case of our networks, the weights π_i are uniform.

The predicted value $\mathbf{v}$ of a subset of random variables $\mathbf{X_1}$ given a set of evidence $\mathbf{X_2} = \mathbf{x}$ using Gaussian mixture regression is

$$\mathbf{v} = E[\mathbf{X_1}|\mathbf{X_2} = \mathbf{x}] = \sum_{i=1}^{k} w_i(\mathbf{x}) m_i(\mathbf{x}) \tag{3}$$

where $m_i(\mathbf{x})$ is the mean vector of the i-th network conditioned on $\mathbf{x}$.

Because Equation 2 relies on evaluating the underlying pdf of a network, it interacts poorly with randomization. If a network is a poor predictor for a given test example, the network may still evaluate to a high likelihood because of the removal of variables by randomization. To fix this problem, we learn weights for each network in our ensemble by ridge regression [27]. Gven a vector of true target values $\mathbf{t}$ and a matrix of each network's predictions for every training example $\mathbf{V}$, we find our weights ω by minimizing

$$||\mathbf{V}\omega - \mathbf{t}||^2 + ||\mathbf{\Gamma x}||^2$$

where $|| \cdot ||$ denotes the Euclidean norm and $\mathbf{\Gamma}$ is a user-defined parameter. These weights differ from those used in GMR in that these weights are not normalized.

To perform regression with our new weights, we augment Equation 3 with the new ridge regression weights as follows

$$v = \sum_{i=1}^{k} w_i(\mathbf{x}) m_i(\mathbf{x}) \omega_{\mathbf{i}} \tag{4}$$

That is, we weight our predictions by both the ridge regression weights and the conventional GMR weights.

IV. Empirical Results

The Storm Scale Ensemble Forecast (SSEF) dataset consists of quantile values for the outputs of variables of 14 different meteorological models. Table I lists the meteorological variables in the SSEF dataset. The variables that come directly from the component models have three values within the SSEF data: The 5th, 50th, and 95th percentile value of the member outputs. The raw SSEF forecast is the mean rainfall forecast across all 14 models. This data comes from Gagne et al., where it was used for rainfall prediction using various machine learning models [28].

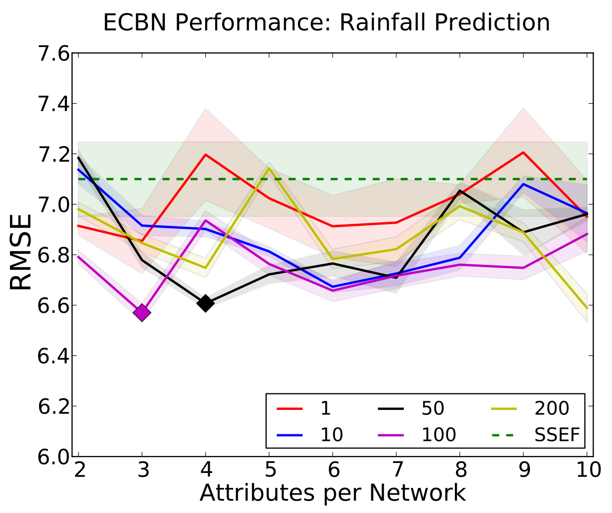

Figure 3. RMSE on the rainfall prediction task. Each line corresponds to the number of attributes kept during randomization. The raw SSEF RMSE is shown for comparison. Shaded area represents standard error. Diamonds indicate a statistically significant improvement over the raw SSEF performance.

1-hour accumulated precipitation
Composite reflectivity
2 m dew point temperature
1-hour maximum reflectivity
Mean sea level pressure
Surface-based Convective Available Potential Energy
Surface-based Convective Inhibition
Precipitable water
2 m air temperature
700 mv temperature
700 mb east-west wind
700 mb north-south wind
700 mb height
500 mv north-south wind
1-hour max upward vertical velocity
1-hour max downward vertical velocity

Table I
THE MODEL VARIABLES SAMPLED FROM THE SSEF RUNS. FROM [28]

Our goal is to predict the error in the hourly rainfall forecast at a given timestep. By subtracting the predicted error from the forecast, we can use ECBN to improve the SSEF forecast. We focus on the forecast for the central plains of the US from May 3rd 2010 to June 18th 2010. The model outputs are at a 4km grid spacing.

A. *SSEF Prediction*

We compare our ECBN results to the RMSE of the raw SSEF forecast. All experiments were run 30 times. Each run left 10 days out (chosen randomly) as a test set and trained on the other 24 days.

Each network was trained on 2000 training examples (selected with bagging). This allows for quick training of each individual network while still allowing the ensemble to be trained on the whole dataset. Ridge regression weights were learned using 10% of the total training set, selected randomly.

We varied both the attributes per model and the size of the ensemble. The significance cutoff for dependence testing during structure learning (α) was set to 0.05. We used a uniform structure prior for structure learning, and uninformative priors for the prior mean and covariance used during the BGe calculation. For ridge regression, we used $\mathbf{\Gamma} = 0.1\mathbf{I}$, where $\mathbf{I}$ is the identity matrix. This value was determined by preliminary experiments performed with ensembles of 100 models.

RMSE results are shown in Figure 3. We compare ECBN's performance at every parameter setting to the raw SSEF performance. However, as we are performing 45 tests, we face a multiple comparisons problem. Aiming for a 5% confidence level, we use Bonferroni correction, resulting in a much lower $\alpha = 0.05/45 = 0.0011$. Using a one-tailed t-test, we find that two parameter settings perform significantly better than the raw SSEF prediction (3 attributes and 100 models as well as 4 attributes and 50 models). From these results, we conclude that ECBN performs best when a number of simple models are combined, although the fact that ensembles of size 200 never performed significantly better than the raw SSEF prediction indicates that our choice of $\mathbf{\Gamma}$ is insufficient to prevent overfitting as the number of models increases.

On the SSEF rainfall prediction task, Gagne et al. report regression performance similar to ECBN's performance using Random Forests (6.62 RMSE with 100 trees across all forecast hours, whereas ECBN with 3 attributes and 100 models achieves an RMSE of 6.57) [28].

We perform a case study using data from May 20, 2010 at 0000 UTC. We first use a Random Forest trained to estimate the probability of rainfall to determine what areas are likely to receive rainfall, and then use ECBN to estimate the amount of rainfall in these areas (see Gagne et al [28] for a more detailed description of this process). For the case study, we use 3 attributes and 100 networks. Figure 4 shows the ECBN forecast across along with the raw SSEF forecast and the true observed precipitation. Compared to the raw SSEF forecast, the ECBN forecast overestimates the amount of rainfall in the western portion of the central plains, but detects rainfall in Missouri and Southeastern Oklahoma that the raw forecast misses.

B. *Variable Importance*

We can also use ECBN to explore the relationships within the SSEF dataset by computing variable importance, which allows for analysis of which variables impact predictive

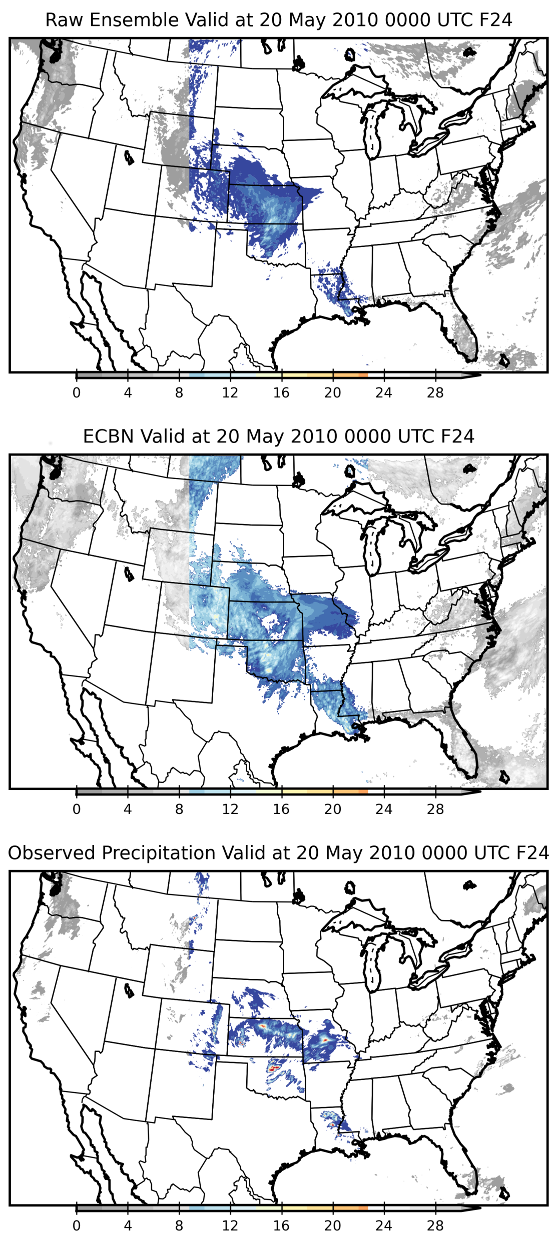
Raw Ensemble Valid at 20 May 2010 0000 UTC F24
0
4
8
12
16
20
24
28
ECBN Valid at 20 May 2010 0000 UTC F24
0
4
8
12
16
20
24
28
Observed Precipitation Valid at 20 May 2010 0000 UTC F24
0
4
8
12
16
20
24
28

Figure 4. Maps of raw SSEF prediction, ECBN prediction, and observed rainfall at 20 May 2010 0000 UTC. Grayed regions represent areas that ECBN was not trained on.

strength [25]. Variable importance is computed by evaluating each component on its out of bag data, and then permuting the values of one variable across the out of bag examples. We recompute RMSE using this permuted data. The difference between the permuted RMSE and the true RMSE is the variable importance score. This procedure is repeated for all variables. Higher variable importance scores indicate that a variable is highly important in prediction. Similarly, lower scores indicate that a variable is not as important in prediction.

Variable	z-score
Max downward vertical velocity 5%	12.6
Accumulated precipitation 50%	12.5
Precipitable water 5%	9.9
Radar quality index	9.8
Precipitable water 50%	9.0
Max downward vertical velocity 50%	8.8
Surface-based CIN 5%	8.4
Accumulated precipitation 5%	8.4
Surface-based CAPE 95%	7.9
Composite reflectivity 50%	7.3

Table II
ALL SIGNIFICANT VARIABLE IMPORTANCE SCORES FOR THE SSEF DATA. 5%, 50%, AND 95% REFER TO THE PERCENTILE.

The variable importance results for the SSEF data are shown in Table II. We compute the z-score for each attribute by averaging each score across 1000 networks and dividing by the standard error. We report the top 10 variables in Table II. These significant variables include precipitable water, CAPE, CIN, and reflectivity, indicating that ECBN is using the atmospheric ingredients for storms to predict rainfall. Accumulated precipitation is the SSEF forecast for rainfall, and so its appearance as a significantly important variable is expected. Our top variable importance results differ from those reported by Gagne et al. for Random Forests [28], where the most important variables included East-West windspeed, North-South windspeed, temperature, and mean sea-level pressure. This indicates that, while both random forests and ECBN can perform well in the SSEF domain, they utilize different information within the domain when predicting rainfall.

V. Conclusion and Future Work

We introduced Ensembled Continuous Bayesian Networks, an approach that allows for prediction of continuous random variables and the discovery of important variables for prediction. We have empirically shown on the SSEF data that ECBN performs better than the raw SSEF prediction for rainfall prediction, achieving results comparable to those reported by Gagne et al. for Random Forests [28]. In other work, we have applied ECBN to reflectivity prediction, achieving better results than both climatology predictions and the raw reflectivity forecast [15].

Temporal data is common, and the ability to use ECBN to model temporal relationships would be useful. Dynamic Bayesian Networks, that is, Bayes nets for temporal data, are well studied. In particular, the BGe score of Linear Gaussian Networks can also be used to score dynamic LGNs [29]. ECBN is able to incorporate dynamic networks into the ensemble.

ECBN could be further extended by the use of hybrid networks within the ensemble, and by learning network parameters with expectation-maximization. Additionally, part of ECBN's predictive strength comes from the differences between individual components in the ensemble. However, we also learn weights across the whole ensemble using ridge regression, which means that the predictions of each component are not actually independent of the other models in the ensemble. Having all component predictions be independent of one another would be ideal, but the weighting is necessary for ECBN to achieve reasonable accuracies. To compensate, we could create ensembles of ECBNs, which would allow for each ECBN to predict independently of the other ECBNs in the ensemble while still retaining the benefits of learning individual network weights across a single ECBN.

Acknowledgements

Thanks to David John Gagne II for his help with the SSEF dataset. This material is based upon work supported by the National Science Foundation under Grant No. EPS-0919466.

References

[1] A. McGovern, D. John Gagne, II, N. Troutman, R. A. Brown, J. Basara, and J. K. Williams, "Understanding severe weather processes through spatiotemporal relational random forests," in *Proceedings of the NASA Conference on Intelligent Data Understanding*. CIDU, 2010.

[2] M. Xue, F. Kong, K. W. Thomas, Y. Wang, K. Brewster, J. Gao, X. Wang, S. J. Weiss, A. J. Clark, J. S. Kain, M. C. Coniglio, J. Du, L. Jensen, and Y. H. Kuo, "Caps realtime storm scale ensemble and high resolution forecasts for the noaa hazardous weather testbed 2010 spring experiment," in *24th Conf. Wea. Forecasting/20th Conf. Num. Wea. Pred. Seattle, WA*. Amer. Meteor. Soc., 2011, p. PAPER 9A.2.

[3] F. Kong, M. Xue, K. W. Thomas, Y. Wang, K. Brewster, X. Wang, J. Gao, S. J. Weiss, J. S. Kain, and J. Du, "Caps multi-model storm-scale ensemble forecast for the noaa hwt 2010 spring experiment," in *24th Conf. Wea. Forecasting/20th Conf. Num. Wea. Pred. Seattle, WA*. Amer. Meteor. Soc., 2011, p. PAPER 457.

[4] A. S. Cofo, R. Cano, C. Sordo, and J. M. Gutirrez, "Bayesian networks for probabilistic weather prediction," in *In Proceedings of the 15th Eureopean Conference on Artificial Intelligence, ECAI2002*. Press, 2002, pp. 695–699.

[5] E. W. Dereszynski and T. G. Dietterich, "Spatiotemporal models for data-anomaly detection in dynamic environmental monitoring campaigns," *ACM Trans. Sen. Netw.*, vol. 8, no. 1, pp. 3:1–3:36, Aug. 2011.

[6] R. J. Kennett, K. B. Korb, and A. E. Nicholson, "Seabreeze prediction using bayesian networks," in *Proc. Fifth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'01). Hong Kong*, 2001, pp. 148–153.

[7] D. Geiger and D. Heckerman, "Learning gaussian networks," in *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, ser. UAI'94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 235–243.

[8] N. Friedman, M. Goldszmidt, and T. J. Lee, "Bayesian network classification with continuous attributes: Getting the best of both discretization and parametric fitting," in *In Proceedings of the International Conference on Machine Learning (ICML*. Morgan Kaufmann, 1998, pp. 179–187.

[9] R. Hofmann and V. Tresp, "Discovering structure in continuous variables using bayesian networks," in *Advances in Neural Information Processing Systems 8*. MIT Press, 1995, pp. 500–506.

[10] A. Pérez, P. Larraòaga, and I. Inza, "Bayesian classifiers based on kernel density estimation: Flexible classifiers," *Int. J. Approx. Reasoning*, vol. 50, no. 2, pp. 341–362, Feb. 2009.

[11] F. Liu, F. Tian, and Q. Zhu, "Ensembling Bayesian network structure learning on limited data," in *CIKM '07: Proceedings of the sixteenth ACM conference on information and knowledge management*. New York, NY, USA: Association of Computing Machinary, 2007, pp. 927–930.

[12] Y. Jing, V. Pavlović, and J. M. Rehg, "Boosted bayesian network classifiers," *Mach. Learn.*, vol. 73, no. 2, pp. 155–184, Nov. 2008.

[13] C. Utz, "Learning ensembles of bayesian network structures using random forest techniques," Master's thesis, University of Oklahoma, 2010.

[14] D. Geiger and D. Heckerman, "Knowledge representation and inference in similarity networks and bayesian multinets," *Artificial Intelligence*, vol. 82, no. 12, pp. 45 – 74, 1996.

[15] S. Hellman, "Learning ensembled dynamic bayesian networks," Master's thesis, University of Oklahoma, 2012.

[16] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[17] D. M. Chickering, "Learning bayesian networks is np-complete," 1996.

[18] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning bayesian networks: The combination of knowledge and statistical data," in *MACHINE LEARNING*, 1995, pp. 197–243.

[19] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, 2nd ed. Cambridge, MA, USA: The MIT Press, Jan. 2001.

[20] I. Tsamardinos, L. Brown, and C. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," in *Machine Learning*, vol. 65, no. 1. Hingham, MA, USA: Kluwer Academic Publishers, March 2006, pp. 31–78.

[21] A. Fast, "Learning the structure of bayesian networks with constraint satisfaction," Ph.D. dissertation, University of Massachusetts Amherst, 2009.

[22] S. G. Bøttcher, "Learning Bayesian Networks with Mixed Variables," Ph.D. dissertation, AALBORG University, 2004.

[23] G. F. Cooper, "The computational complexity of probabilistic inference using bayesian belief networks," *Artificial Intelligence*, vol. 42, no. 23, pp. 393 – 405, 1990.

[24] R. Shachter and C. Kenley, "Gaussian influence diagrams," *Management Science*, vol. 35, pp. 527–550, 1989.

[25] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001, 10.1023/A:1010933404324.

[26] H. Sung, "Gaussian mixture regression and classification," 2004.

[27] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[28] D. John Gange, II, A. McGovern, and M. Xue, "Machine learning enhancement of storm scale ensemble precipitation forecasts," in *Proceedings of the NASA Conference on Intelligent Data Understanding*. CIDU, 2012.

[29] M. Grzegorczyk and D. Husmeier, "Non-stationary continuous dynamic Bayesian networks," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds., 2009, pp. 682–690.