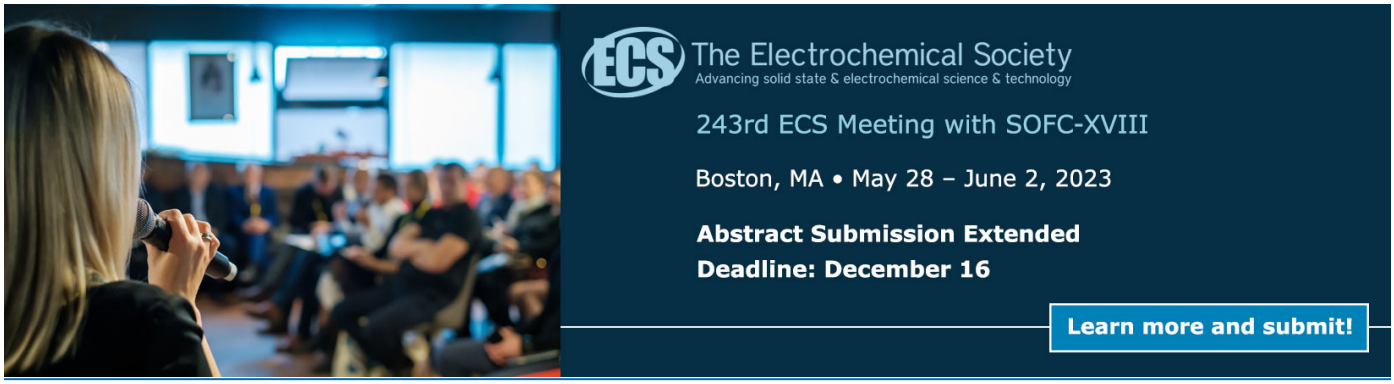
Probabilistic Wind Park Power Prediction using Bayesian Deep Learning and Generative Adversarial Networks



Probabilistic Wind Park Power Prediction using Bayesian Deep Learning and Generative Adversarial Networks

Lars Ødegaard Bentsen[1], Narada Dilp Warakagoda[1], Roy Stenbro[2] and Paal Engelstad[1]

[1]Department of Technology Systems, University of Oslo, PO Box 70, 2027 Kjeller, Norway.
[2]Institute for Energy Systems (IFE), PO Box 40, 2027 Kjeller, Norway.

E-mail: lars.bentsen@its.uio.no

Abstract. The rapid depletion of fossil-based energy supplies, along with the growing reliance on renewable resources, has placed supreme importance on the predictability of renewables. Research focusing on wind park power modelling has mainly been concerned with point estimators, while most probabilistic studies have been reserved for forecasting. In this paper, a few different approaches to estimate probability distributions for individual turbine powers in a real off-shore wind farm were studied. Two variational Bayesian inference models were used, one employing a multilayered perceptron and another a graph neural network (GNN) architecture. Furthermore, generative adversarial networks (GAN) have recently been proposed as Bayesian models and was here investigated as a novel area of research. The results showed that the two Bayesian models outperformed the GAN model with regards to mean absolute errors (MAE), with the GNN architecture yielding the best results. The GAN on the other hand, seemed potentially better at generating diverse distributions. Standard deviations of the predicted distributions were found to have a positive correlation with MAEs, indicating that the models could correctly provide estimates on the confidence associated with particular predictions.

1. Introduction

Wind energy projections are rapidly accelerating and in order to rely on these variable resources, better prediction models are required to maintain the security of power grids and curb the reliance on conventional power plants [1, 2]. Wind turbines are characterised by their power curves, which indicate the expected power production at different wind speeds. However, in non-ideal conditions and when turbines are situated close together, actual values deviate from power curve predictions. Wake losses are the reduction in power for turbines situated downstream in a farm, where some of the power is extracted by upstream turbines. Various analytical models aim to model these interaction losses [3, 4], but despite their appealing simplicity, their accuracy for real wind farms are becoming unsatisfactory. As increased computational resources have become readily available, once impractical numerical methods have now become available for advanced studies. Computational fluid dynamics can for example provide high resolution flow fields [5], but still impose significant assumptions on the systems and come at a high computational cost, making them less viable for real-time systems.

Recently, there has also been a surge in the historical data available for modern wind farms,

making machine learning (ML) methods gain traction, as these can significantly reduce run-times after training, while providing accurate results. Various ML methods have been studied, such as support vector regressors [6], fuzzy logic [7] and k-nearest neighbour algorithms [8], with a plethora of studies focusing on deep learning (DL) [9]. Yan et al. [10] used a multilayer perceptron (MLP) model to predict power at different wind speeds and directions for a real wind farm. Some have also improved farm predictions by considering multiple sites jointly to capture global wind characteristics [11, 12]. Convolutional Neural Networks (CNN) have been used to learn spatial relations in wind farms, where the features of each turbine were represented by a cell in a two dimensional grid [13, 14]. In contrast to CNNs, graph neural networks (GNN) can model arbitrary structures, such as wind farms, where turbines might not be arranged in a regular grid. Bleeg [15] used a GNN model to predict individual turbine powers for farms with considerable wake losses, using data simulated with a Reynolds-avereged Navier-Stokes model. Furthermore, Park et al. [16] proposed the physics-induced GNN, which was shown to yield excellent prediction performance. Finally, Yu et al. [17] constructed the superposition GNN, which considered both spatial and temporal characteristics, trained on real wind measurements.

For real measurements, there is a significant degree of uncertainty associated with the data. In particular, for a specific wind speed and direction value, the recorded wind power will not be exactly the same, due to factors such as sensor drifts, non-stationary atmosphere, non-uniform wind fields across a farm and recorded values being averaged over some time interval. Even though the previously described work achieved adequate prediction performance, they did not consider the inherent uncertainty associated with making predictions for real systems. Various other work address the issue by having models also output the prediction uncertainty [18, 19]. Bayesian methods show great promise for probabilistic wind power prediction [20, 21, 18]. Liu et al. [21] used a convolutional operator to capture spatial and temporal dynamics, and variational Bayesian inference to make the model probabilistic. Nevertheless, most studies considering Bayesian models have been concerned with forecasting, while the applications to wind park modelling, for which power production is predicted from the current wind condition, have been scarce [22, 23]. Furthermore, recent advancements within the field of Bayesian DL makes this an interesting area for further investigation.

Generative Adversarial Networks (GAN) were first introduced by Goodfellow et al. [24] and have since become the quintessential DL architecture for image generation. Recently, GANs have also been proposed as probabilistic prediction models [25, 26]. Adler et al. [25] used a GAN for image reconstruction of computed tomography images. By sampling from the model, the authors managed to obtain probabilistic predictions, with point-wise mean and standard deviation estimates for the reconstructed images. Similarly, Lee et al. [26] used a conditional GAN as a probabilistic regression model to estimate the uncertainty, or risk, associated with an expected return prediction for the stock market. The main contributions of this paper can be summarised as follows:

- Two Bayesian models, based on a GNN and MLP architecture, are used for predicting individual turbine powers for different wind conditions, where Flipout was employed for Bayesian variational inference (VI). These models were able to generate accurate probability distributions of the turbine power outputs, with the GNN architecture yielding the most accurate point predictions.
- A novel application of a GAN used for probabilistic wind power modelling was studied. The model leveraged recent advancements within GAN theory, focusing on a stacked Wasserstein GAN approach. It was found that the model achieved competitive results with the Bayesian models, while showing slightly different characteristics.
- All models estimated higher uncertainties for predictions with larger errors, showing that the models could correctly indicate the confidence associated with a particular prediction.

2. Theory

2.1. Graph Neural Networks

A graph can be defined as a tuple containing global, node and edge features, as $G = (u, V, E)$ [27]. E, contains edge-specific features, e_{ij}, meaning properties that provide information on the relationship between two nodes i and j. Similarly, V contains node-specific features, v_i, and u are the global features, which are shared across a graph. GNNs consist of stacked graph blocks, which update the edge, node and global features. For a single block, the features are updated using three functions, $\phi^{(\cdot)}$, in the following order:

$$e'_{ij} = \phi^e(e_{ij}, v_i, v_j, u) \tag{1}$$
$$v'_j = \phi^v(v_j, \bar{e}'_j, u) \tag{2}$$
$$u' = \phi^u(\bar{e}', \bar{v}', u). \tag{3}$$

Updated features, $(\cdot)'$, are fed as inputs to the next graph block. Three functions, $\rho^{(\cdot)\to(\cdot)}$, map edges-to-nodes, edges-to-globals and nodes-to-globals, to compute aggregated features, $\bar{(\cdot)}$, as

$$\bar{e}'_j = \rho^{e\to v}(E'_j), \text{ where } E'_j = \{e'_{ij} | \forall i \in R_j\} \tag{4}$$
$$\bar{e}' = \rho^{e\to u}(E'), \text{ where } E' = \{e'_{ij} | \forall (i,j) \in U\} \tag{5}$$
$$\bar{v}' = \rho^{v\to u}(V'), \text{ where } V' = \{v'_i | \forall i \in I\}. \tag{6}$$

R_j is a set containing indices for all nodes sending to a node, j. U and I are the sets containing all edges, (i, j), and nodes, i, respectively. These sets define the connectivity of a graph.

2.2. Bayesian Deep Learning and Variational Inference

Bayesian methods propose a prior for the model parameters and updates the belief about these parameters as more observations are available. A Bayesian neural network (BNN) can simply be regarded as a neural network for which the weights and biases are sampled from this updated distribution, known as the posterior, $p(\theta|\mathcal{D})$, where $\mathcal{D}$ is the dataset and θ the network weights. However, since computing the posterior directly is typically intractable, approximate inference is required. Markov Chain Monte Carlo methods for approximating the posterior are often computationally expensive and our study has instead focused on variational inference (VI), which is known to scale better [28, 29]. The aim of VI is to find a distribution $q(\theta)$ which is a good approximation of the posterior, by minimising the Kullback-Leibler (KL) divergence [30]:

$$q^*(\theta) = \arg\min_{q(\theta)} \mathrm{KL}(q(\theta) \parallel p(\theta|\mathcal{D})). \tag{7}$$

By writing out the terms of the KL-divergence it can be found that

$$\ln p(\mathcal{D}) = \mathbb{E}_{q(\theta)} \ln \frac{p(\theta, \mathcal{D})}{q(\theta)} + \mathrm{KL}(q(\theta) \parallel p(\theta|\mathcal{D})). \tag{8}$$

Even though the evidence, $p(\mathcal{D})$, is difficult to compute, it is constant, meaning that the KL-divergence can be minimised by maximising the expectation in eq.(8), known as the evidence lower bound (ELBO). Expanding the terms in eq.(8), the ELBO can be written as

$$\mathrm{ELBO} = \mathbb{E}_{q(\theta)} \ln p(\theta, \mathcal{D}) - \mathbb{E}_{q(\theta)} \ln q(\theta). \tag{9}$$

Flipout is a specific method for weight perturbation to infer $q(\theta)$ through gradient ascent based maximisation of the ELBO loss [31]. In particular, Flipout aims to reduce the variance of gradients, which is a deficit of the original Bayes by backprop formulation [32], by decorrelating gradients within a single update-step of model parameters.

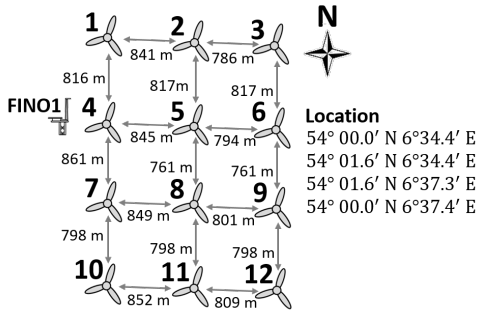


Figure 1: Layout of the Alpha Ventus wind farm. FINO1 is a wind measurement platform.

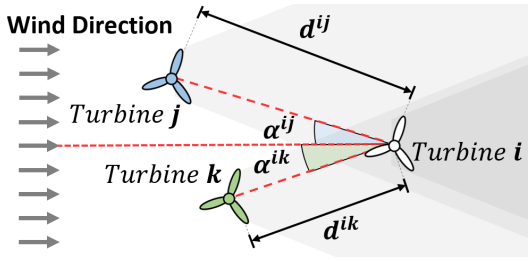


Figure 2: Visualisation of a turbine i with two upstream neighbours j and k.

2.3. Generative Adversarial Networks

GANs were first introduced by Goodfellow et al.[24], and have since become a de facto generative architecture within DL. A generator and discriminator are trained simultaneously and play a minimax two-player game, in which the generator aims to learn the underlying data distribution to generate samples that can fool the discriminator, which classifies samples as real or fake.

GANs are not trained in a conventional supervised manner, but have still showed excellent results in generating diverse samples that lie close to the input distribution, without replicating samples [24]. Nevertheless, GANs are inherently difficult to train, as the two competing networks are optimised simultaneously. The Wasserstein GAN (WGAN) was proposed to improve training by minimising the Wasserstein distance instead of the Jensen-Shannon (JS) divergence, which was argued to help avoid vanishing gradients [33, 34]. The game played by the discriminator and generator in WGANs, with a value function $V(G, D)$, can be summarised as

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[D(x; \theta_D)] - \mathbb{E}_{z \sim p_z(z)}[D(G(z; \theta_G); \theta_D)], \tag{10}$$

where $p_z(z)$ is a prior on the input noise variables, $p_{data}(x)$ the real data distribution and D and G are the discriminator and generators, parameterised by θ_D and θ_G, respectively.

Gulrajani et al. [34] proposed a gradient penalty term to improve the training of WGANs, instead of weight clipping, to enforce the Lipschitz constraint on the discriminator in WGANs [33]. Mirza et al. [35] fed a condition to both the generator and discriminator to control the samples generated. Learning the generator mapping can become increasingly challenging for complex data distributions. Zhang et al. [36] therefore introduced the StackGAN, where the process of mapping from the latent to the input space was split over multiple discriminator-generator (D-G) pairs. In their particular example, the first D-G pair generated 64x64 pixel images, which were fed to the second stage D-G pair to generate 256x256 full-scale images.

3. Methods

3.1. Dataset

The dataset contained 10-minute averaged power measurements for the 12 turbines in the Alpha Ventus off-shore wind farm, shown in Fig. 1, where turbines 1-6 and 7-12 were Senvion and Adwen 5M, respectively. The free stream velocity was estimated by taking the average wind speed measured by nacelle anemometers at the upstream turbines for a particular wind direction. With reference to Fig. 1, this meant for a wind direction of 270 degrees (i.e. blowing from left to right in Fig. 1), the velocity was taken as the mean measurements from turbines 1, 4, 7 and 10. Data were discarded for periods in which any turbine was derated or the wind speed was higher than the turbines' rating. The dataset contained 227,004 individual turbine power

measurements, where 20% of the data was held out for testing. The remaining 80% was split into training and validation sets, also using an 80-20% split, respectively. Throughout, we refer to a turbine, j, being upstream of turbine, i, if the angle with respect to the wind direction is within $\pm 30^o$ (i.e. $|\alpha^{ij}| < 30^o$, in Fig. 2).

3.2. Bayesian Models

Two Bayesian models were implemented, based on VI using the 'DenseFlipout' layers available in TensorFlow Probability [31, 37]. Predicted probability distributions for the turbine powers were obtained by sampling different weights and biases for a particular input.

3.2.1. BMLP Model: The first model, which will be referred to as the Bayesian MLP (BMLP) model, predicted the power for a single turbine, given the wind speed and location of upstream neighbours as inputs. Considering the scenario illustrated in Fig. 2, the upstream model would predict the power for turbine i at time t as

$$\hat{P}_t^{(i)} = f(ws_t, l_i, d^{ik}, \sin(\alpha_t^{ik})\cos(\alpha_t^{ik}), l_k, d^{ij}, \sin(\alpha_t^{ij})\cos(\alpha_t^{ij}), l_j), \tag{11}$$

where f is the function represented by an MLP formed by stacked 'DenseFlipout' layers, ws_t the wind speed, l the turbine type, set to either 1 (Senvion) or 0 (Adwen), d and α, the distance and angle between turbines, respectively. Even though we refer to 'time', t, this was only to indicate the samples in the dataset, i.e. measurements taken at different times, as temporal correlations were not considered. Depending on the wind direction and turbine concerned, the number of upstream neighbours would vary and inputs were therefore padded to ensure constant lengths.

3.2.2. BGNN Model: The second Bayesian model was a GNN (BGNN), in which the update functions, $\phi^{(\cdot)}$, in eq. (1-3) were MLPs formed by stacked 'DenseFlipout' layers. A set of graphs were created, where each graph corresponds to a particular time-step, t. For each graph, the wind speeds, ws_t, were assigned to the input global features, while the turbine types, l, were set as node features. The distances and relative angles between turbines were set to the input edge features as $e_t^{(ij)} = [d^{ij}, \sin(\alpha_t^{ij})\cos(\alpha_t^{ij})]$. The BGNN model updates edge, node and global features, trained to predict turbine powers embedded in the node features.

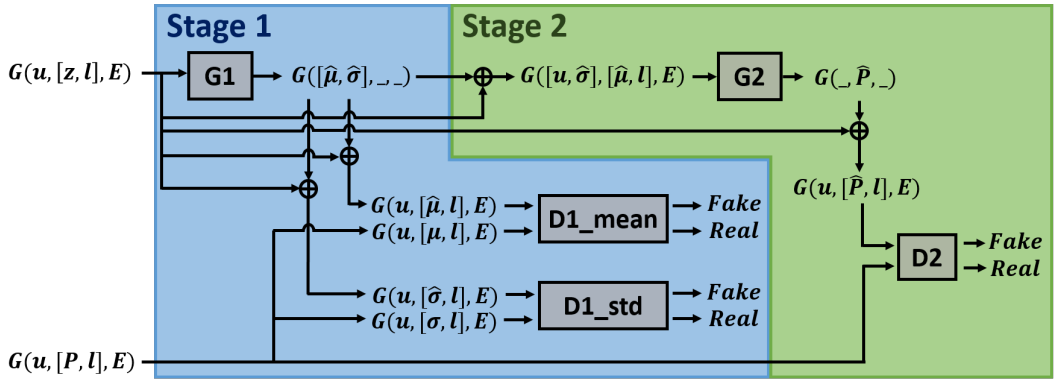


Figure 3: Illustration showing the proposed cStackWGAN architecture used to predict probability distributions for turbine powers.

3.3. cStackWGAN Model

For this study, a conditional Stack Wasserstein GAN (cStackWGAN) was proposed, having generators and discriminators represented by GNNs, with MLPs as the update functions, $\phi^{(\cdot)}$, in eq. (1-3). An illustration of how the cStackWGAN operates is given in Fig. 3. The input conditions were the same as for the GNN model described in Sec. 3.2.2, but with an additional random variable, z, embedded in the node features, shown as the input to G1 in Fig. 3. The first stage consisted of one generator and two discriminators, where the generator produced mean and standard deviation predictions for the power of the entire farm, denoted $\hat{\mu}$ and $\hat{\sigma}$ in Fig. 3, respectively. This was thought to be a simpler task than predicting individual turbine powers directly. Two discriminators, 'D1_mean' and 'D1_std', were used to classify the mean and standard deviation predictions, respectively, as real or fake, given the input conditions.

Outputs from the first stage generator were used as inputs to the second stage, where the predicted means and turbine types were cast to the node features, and the standard deviation predictions and wind speeds to the globals. The second stage generated individual turbine power predictions, $\hat{P}$, and a single discriminator was trained to classify samples as real or fake. By feeding different samples, z, from the latent distribution together with any given condition, $[u, l, E]$, we can generate multiple turbine power predictions. These predictions can be used to construct an empirical probability distribution model (histogram) for the predicted power.

A few well known techniques to stabilise the training of GANs were investigated when tuning hyperparameters. Since some wind conditions were more prevalent than others, oversampling was used to more frequently select samples with less represented wind conditions during training. Different gradient penalties and clipping techniques were also implemented [34, 38]. Noisy labels can be used to make models more robust against noise in the data and avoid saturation of the discriminator, by flipping the labels according to a small probability, meaning that some generated samples are labeled as real and vice versa. Furthermore, adding noise to the inputs of the discriminator can avoid the discriminator learning an optimal behaviour too quickly [39], and adding noise to the conditions could improve training for continuous conditions [40].

3.4. Training

To arrive at appropriate hyperparameters, tuning processes were conducted in Optuna [41]. Different batch sizes, learning rates, number of hidden layers and units, dropout rate, hidden activation functions, layer- and batch normalisation were tested for all models. Neither dropout, layer- or batch normalisation were used for the final Bayesian models, while layer normalisation was used for the first-stage discriminators and a dropout rate of 20% for the second-stage generator in the cStackWGAN model. The Adam optimiser yielded the best results for all models. Furthermore, oversampling was found advantageous for the first-stage D-G pair, as well as adding some noise to the discriminator inputs. Gradient penalties were implemented according to the proposed methods outlined in [34] and [38], but slightly altered to facilitate graph structures. A gradient penalty based on interpolation between real and generated samples, as in [34], was found to yield the best results for the second-stage discriminator. However, a slightly different gradient penalty was found favourable for the first stage, computed based on noise added to real samples [38]. For brevity, we will not go into the exact details of the gradient penalties, but wish to refer the interested reader to the literature used for this study [34, 38].

4. Results and Discussion

4.1. Prediction Performance

The aim of the models was to predict the probability distributions for the power of individual turbines, given the wind speed and direction. To asses the performance of the different models, it was necessary to obtain estimates of the true power distributions for different wind conditions. Given an input for a specific wind condition, the true power distribution was approximated by

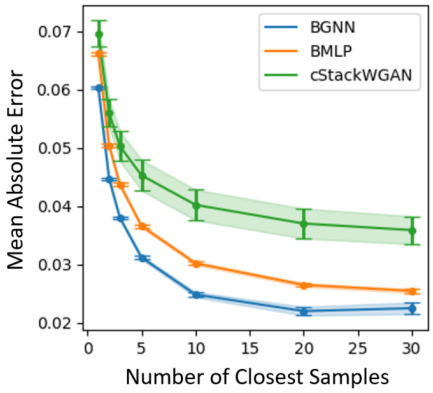


Figure 4: MAE for the different models against the number of closest samples in the dataset used to determine the labels. Results are the mean $\pm\sigma$ for five runs.

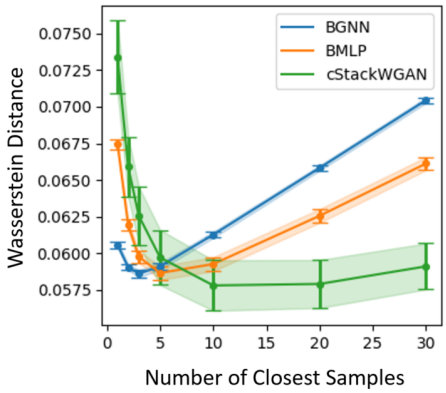


Figure 5: Mean Wasserstein distance between the generated and real distributions for different number of closest samples in the real distributions. Results are the mean $\pm\sigma$ for five runs.

finding the samples in the test set which had the most similar wind speeds and directions to the input, based on the Euclidean distance. It was assumed that the powers for data points with very similar wind conditions should be a reasonable estimate for the true power distribution for a specific wind condition. Fig. 4 shows the variation of mean absolute errors (MAE) over the entire test set, for different number of closest samples, n, used for the true labels. For $n = 10$, the label was taken as the average power for the 9 closest points to the input and the power for the input itself. To obtain point estimates, we sampled 1000 predictions from the models, given the same input, and computed the means of the generated power distributions. Each model was trained five times, using the same test, train and validation sets, with the results given in Fig. 4. As would be expected, the MAEs improved for all models as the number of samples used to compute the labels increased. The Bayesian models were superior to the cStackWGAN model, with the BGNN model yielding the best results. For real datasets, there will always be some uncertainty associated with the measurements, which could result in single values not being representable of global characteristics. The fact that MAEs curtailed when considering the mean of similar data points could indicate that the models did not overfit to single samples.

A potentially more informative measure to evaluate the predicted distributions is the Wasserstein metric, which is the work required to transform one probability distribution into another, and can therefore be used as a distance measure. Fig. 5 shows the Wasserstein distance between the predicted and true distributions. First, the BGNN model achieved very appreciable results, outperforming the other models, before quickly performing worse as the true distributions were estimated from more samples. The same behaviour was observed for the BMLP model, but with results between the other two models. Even though the cStackWGAN model initially performed worse than the Bayesian models, it seemed better equipped at predicting the true distributions as more neighbouring samples were considered. Since MAEs did not significantly increase in Fig. 4, it meant that the mean of the predicted distributions were not the reason for the degrading performance of the Bayesian models in Fig. 5. It was therefore thought that the Bayesian models struggled to predict the correct standard deviations or shape of the true probability distributions, while the GAN model was potentially better at this. When the true distributions were based on only 1-3 samples, the shape of the predicted distributions were potentially not very important, as long as the means were reasonably accurate. As more samples were considered for the true distributions, the shape would become more prominent, and it seemed that the cStackWGAN model was better at generating these. To confirm this,

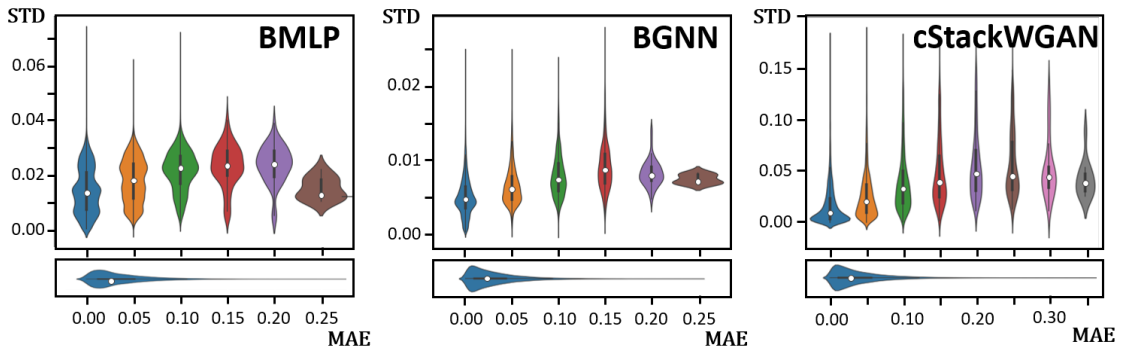


Figure 6: Violin plots of standard deviations (STD) of the predicted distributions against MAEs. The bottom row illustrates how the MAEs were distributed.

the standard deviations of the predicted and true distributions for the different models were compared and it was found that all models correctly captured increases in standard deviations, but had predictions that were generally too small. The GAN model was the best at capturing the spreads, while the Bayesian models' estimates on average had standard deviations that were even smaller than those for the cStackWGAN model. The difference in magnitudes between the models can for example be seen by the range of standard deviations in Fig. 6.

4.2. Evaluating Uncertainty

A desirable property of probabilistic models, is that the spread of predicted distributions gives an indication as to the models' confidence in a particular prediction. For this to hold, there should be a positive correlation between the standard deviation of predicted probability distributions and MAEs. In Fig. 6, the violin plots visualise how the standard deviations were distributed for particular MAE bins, with white dots indicating median values. Generally, the standard deviations of the predictions increased as the MAE increased, which meant that the spread of the generated distributions could in fact be used as an indication of the models' confidence. However for excessively large errors, it was seen that the standard deviations decreased, which was undesirable. Nevertheless, considering how the MAEs were distributed, illustrated by the second row of Fig. 6, most predictions on the test set were associated with MAEs smaller than 0.10. This meant that the plots for larger MAE bins were constructed from very few data points and might poorly reflect the true characteristics of the models.

Even though probabilistic models often yield slightly inferior results with regards to MAEs, compared to point estimators, providing more detailed information on the expected probability distributions could nevertheless help improve the security of prediction systems, as the users would obtain information on when to trust or disregard a prediction made by the model.

4.3. Investigation of Probability Distributions

Finally, some predicted and true distributions were plotted for two turbines, 4 and 10 (with reference to Fig. 1), for two wind conditions in Fig. 7. Here, the histograms were scaled in (0, 1) and fitted some approximate distributions, where the red bins correspond to the predicted distributions. The true powers for the input wind conditions are represented by the single black bins and the blue histograms are the powers for the 20 samples with the most similar wind conditions to the two inputs. The maximum and minimum wind speeds amongst the 20 nearest neighbours were ± 0.15 and ± 0.20 m/s, for the two wind conditions. Considering turbine 4, i.e. the bottom row of Fig. 7, the models seemed to capture the distributions fairly well. The cStackWGAN predicted slightly too small spread and a skewed distribution, while the BGNN's

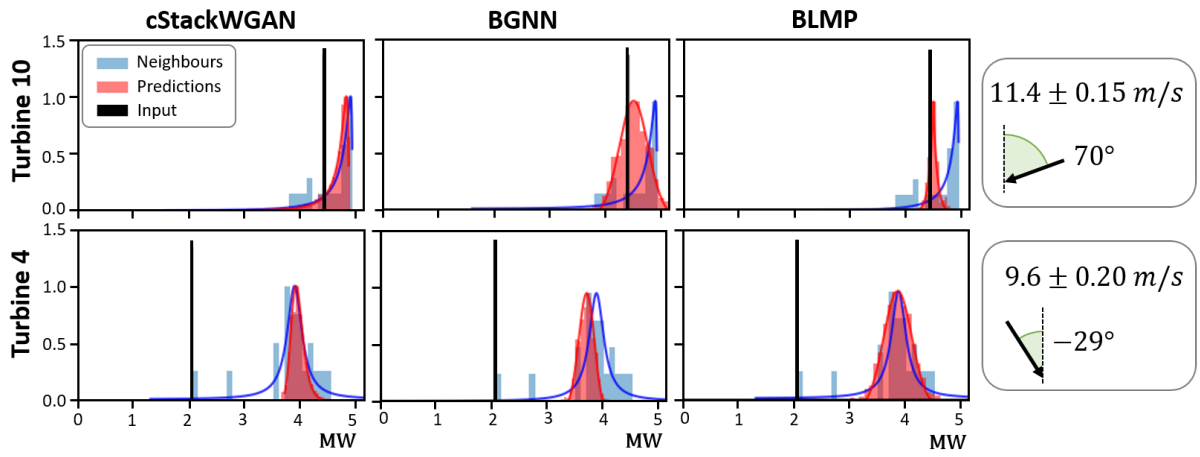


Figure 7: Predicted (red) and true (blue) distributions for turbines 4 and 10 for two randomly selected wind conditions. Histograms are scaled in (0, 1), while the power for the inputs used to compute the predicted distributions are shown by single grey bins. True distributions were taken as the 20 closest neighbours to the inputs based on the wind condition, while the predicted were constructed by sampling 1000 times from the models for the same input condition.

predictions had a slightly offset mean. However, it would not be sensible to conclude on relative performances based on a few random samples. Instead, it was more interesting to observe that all models captured the true distributions reasonably well for the second row of Fig. 7, despite the power corresponding to the input being very small. Particular measurements might be offset or seem somewhat random for real datasets, and it would be undesirable for the models to confidently predict powers to be around 2 MW for turbine 4, given the particular wind condition.

For wind conditions where the turbines produced powers close to the rating of 5 MW, the Bayesian models tended to predict slightly too small values, while the cStackWGAN performed very well, as seen for the Turbine 10 predictions in the top row of Fig. 7. The cStackWGAN model was able to generate slightly more diverse distributions, such as that seen in the top row of Fig. 7, while the Bayesian models generally seemed to produce Gaussian distributions.

5. Conclusion

Three different DL models were investigated for probabilistic wind park power modelling, focusing on two Bayesian VI models and a novel GAN architecture. The Bayesian models adopted recent developments within the field of Bayesian DL, namely VI using Flipout, on an MLP and GNN architecture, with the latter generally yielding the best results. By investigating a novel approach to probabilistic wind power modelling using a GAN, it was interesting to study the relative differences between more traditional Bayesian DL and a new application of a generative model. The Bayesian models seemed well equipped for predicting individual turbine powers (MAEs) and generating acceptable probability distributions. Even though the cStackWGAN model was inferior with regards to MAEs, the model seemed better at generating diverse distributions. Furthermore, all models showed a positive correlation between the standard deviation of generated distributions and MAEs, indicating that the models could provide estimates on the confidence associated with particular predictions. In the future, it would be particularly interesting to explore how the Bayesian models can be altered to produce more diverse distributions. Using datasets for farms with more turbines, as well as extending the scope to consider temporal correlations and forecasting would also be compelling. Further, aleatoric and epistemic uncertainties were not considered separately in this study, but it would

be interesting to do so in future work.

Acknowledgements
The research was financed in part by the research project ELOGOW (led by IFE), which is funded by the Research Council of Norway (project nr: 308838) under the PETROMAKS2 program with financial support from Equinor Energy AS, ConocoPhillips Skandinavia AS and Aibel AS. The dataset was made available by the RAVE initiative, which was funded by the German Federal Ministry of Economic Affairs and Energy (see: www.rave-offshore.de).

References

[1] Treiber N A, Heinermann J and Kramer O 2016 *Computational sustainability* 13–29
[2] Nazir M S, Ali N, Bilal M and Iqbal H M 2020 *Curr. Opin. Environ. Sci. Health* **13** 85–90
[3] Emeis S 2010 *Wind energy* **13** 459–469
[4] Jensen N O 1983
[5] Sanderse B, Van der Pijl S and Koren B 2011 *Wind energy* **14** 799–819
[6] Fang R, Wang Y, Shang R, Liang Y, Wang L and Peng C 2016 *Int. J. Hydrog. Energy* **41** 15733–15739
[7] Damousis I G, Alexiadis M C, Theocharis J B and Dokopoulos P S 2004 *IEEE Trans. Energy Convers.* **19** 352–361
[8] Kusiak A, Zheng H and Song Z 2009 *Renew. Energy* **34** 1487–1493
[9] Wang Y, Yu Y, Cao S, Zhang X and Gao S 2020 *Artif. Intell. Rev.* **53** 3447–3500
[10] Yan C, Pan Y and Archer C L 2019 *Wind Energy* **22** 1421–1432
[11] Ghaderi A, Sanandaji B M and Ghaderi F 2017 *arXiv preprint arXiv:1707.08110*
[12] Grover A, Kapoor A and Horvitz E 2015 *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.* 379–86
[13] Kou P, Wang C, Liang D, Cheng S and Gao L 2020 *IET Renew. Power Gener.* **14** 2416–28
[14] Zhu Q, Chen J, Shi D, Zhu L, Bai X, Duan X and Liu Y 2019 *IEEE Trans. Sustain. Energy* **11** 509–23
[15] Bleeg J 2020 *JPCS* **1618** 062054
[16] Park J and Park J 2019 *Energy* **187** 115883
[17] Yu M, Zhang Z, Li X, Yu J, Gao J, Liu Z, You B, Zheng X and Yu R 2020 *FGCS* **113** 145–57
[18] Adedipe T, Shafiee M and Zio E 2020 *Reliability Engineering & System Safety* **202** 107053
[19] Wang Y, Hu Q, Meng D and Zhu P 2017 *Appl. energy* **208** 1097–1112
[20] Li G and Shi J 2012 *Renew. Energy* **43** 1–8
[21] Liu Y, Qin H, Zhang Z, Pei S, Jiang Z, Feng Z and Zhou J 2020 *Appl. Energy* **260** 114259
[22] Yan J, Zhang H, Liu Y, Han S and Li L 2019 *Applied energy* **239** 1356–1370
[23] Rogers T, Gardner P, Dervilis N, Worden K, Maguire A, Papatheou E and Cross E 2020 *Renewable Energy* **148** 1124–1136
[24] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y 2014 *NIPS* **27**
[25] Adler J and Öktem O 2018 *arXiv preprint arXiv:1811.05910*
[26] Lee M and Seok J 2021 *Sensors* **21** 6194
[27] Battaglia P W, Hamrick J B, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R *et al.* 2018 *arXiv preprint arXiv:1806.01261*
[28] Salimans T, Kingma D and Welling M 2015 *ICML* 1218–1226
[29] Caceres J, Gonzalez D, Zhou T and Droguett E L 2021 *Struct Control Health Monit* **28** e2811
[30] Blei D M, Kucukelbir A and McAuliffe J D 2017 *J Am Stat Assoc J AM STAT ASSOC* **112** 859–877
[31] Wen Y, Vicol P, Ba J, Tran D and Grosse R 2018 *arXiv preprint arXiv:1803.04386*
[32] Blundell C, Cornebise J, Kavukcuoglu K and Wierstra D 2015 *ICML* 1613–1622
[33] Arjovsky M, Chintala S and Bottou L 2017 *ICML* 214–223
[34] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V and Courville A 2017 *arXiv preprint arXiv:1704.00028*
[35] Mirza M and Osindero S 2014 *arXiv preprint arXiv:1411.1784*
[36] Zhang H, Xu T, Li H, Zhang S, Wang X, Huang X and Metaxas D N 2017 *ICCV* 5907–5915
[37] Dillon J V, Langmore I, Tran D, Brevdo E, Vasudevan S, Moore D, Patton B, Alemi A, Hoffman M and Saurous R A 2017 *arXiv preprint arXiv:1711.10604*
[38] Kodali N, Abernethy J, Hays J and Kira Z 2017 *arXiv preprint arXiv:1705.07215*
[39] Arjovsky M and Bottou L 2017 *arXiv preprint arXiv:1701.04862*
[40] Ding X, Wang Y, Xu Z, Welch W J and Wang Z J 2020 *arXiv preprint arXiv:2011.07466*
[41] Akiba T, Sano S, Yanase T, Ohta T and Koyama M 2019 *25th Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.* 2623–2631