Forecasting the Disturbance Storm Time Index with Bayesian Deep Learning

Table of Contents

Author(s)

- Author1 = {"name": "Yasser Abduallah", "affiliation": "Department of Computer Science, New Jersey Institute of Technology", "email": "ya54@njit.edu (mailto:ya54@njit.edu)", "orcid": "https://orcid.org/0000-0003-0792-2270"} (https://orcid.org/0000-0003-0792-2270"%7D)
- Author2 = {"name": "Jason T. L. Wang", "affiliation": "Department of Computer Science, New Jersey Institute of Technology", "email": "wangj@njit.edu (mailto:wangj@njit.edu)", "orcid": "https://orcid.org/0000-0002-2486-1097"} (https://orcid.org/0000-0002-2486-1097"%7D)
- Author3 = {"name": "Prianka Bose", "affiliation": "Department of Computer Science, New Jersey Institute of Technology", "email": "pb524@njit.edu (mailto:pb524@njit.edu)", "orcid": ""}
- Author4 = {"name": "Genwei Zhang", "affiliation": "Department of Computer Science, New Jersey Institute of Technology", "email": "gz6@njit.edu (mailto:gz6@njit.edu)", "orcid": ""}
- Author5 = {"name": "Firas Gerges", "affiliation": "Department of Computer Science, New Jersey Institute of Technology", "email": "fg92@njit.edu (mailto:fg92@njit.edu)", "orcid": ""}
- Author6 = {"name": "Haimin Wang", "affiliation": "Institute for Space Weather Sciences, New Jersey Institute of Technology", "email": "haimin.wang@njit.edu

(mailto:haimin.wang@njit.edu)", "orcid": "https://orcid.org/0000-0002-5233-565X"} (https://orcid.org/0000-0002-5233-565X"%7D)

Purpose

The disturbance storm time (Dst) index is an important and useful measurement in space weather research. It has been used to characterize the size and intensity of a geomagnetic storm. A negative Dst value means that the Earth's magnetic field is weakened, which happens during storms. Here, we present a novel deep learning method, called the Dst Transformer (or DSTT for short), to perform short-term, 1-6 hour ahead, forecasting of the Dst index based on the solar wind parameters provided by the NASA Space Science Data Coordinated Archive. The Dst Transformer combines a multi-head attention layer with Bayesian inference, which is capable of quantifying both aleatoric uncertainty and epistemic uncertainty when making Dst predictions. Experimental results show that the proposed Dst Transformer outperforms related machine learning methods in terms of the root mean square error and R-squared. Furthermore, the Dst Transformer can produce both data and model uncertainty quantification results, which can not be done by the existing methods. To our knowledge, this is the first time that Bayesian deep learning has been used for Dst index forecasting.

In this notebook we provide an overview of the DSTT system to demonstrate how to forecast Dst index using deep learning (DL) and solar wind parameters and provide uncertatinty quantification with Bayesian network.

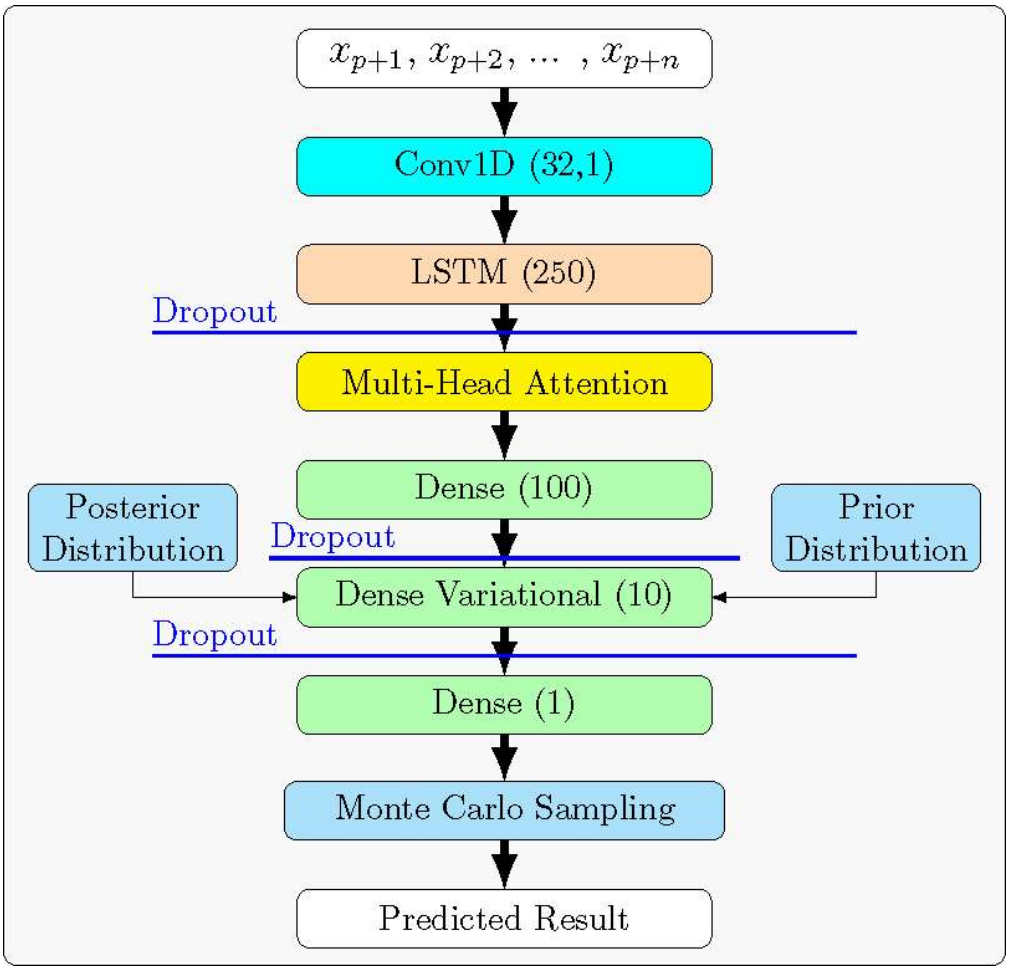
Technical Contributions

- We provide the community with a new tool to forecast the occurrance of the disturbance storm time (Dst) index for the next 1, 2, 3, 4, 5, or 6 hours ahead. The tools also provide data and model uncertainty quantification.

Methodology

The figure below presents the architecture of our DSTT model. DSTT is created using the tensorflow keras framework. We add multiple layers to DSTT to enhance its performance and improve its learning capability. The model accepts as input non-overlapping sequences of records $x_{p+1}, x_{p+2}, \ldots, x_{p+n}$, where n is set to 1024 in our study. Each sequence is passed to a one-dimensional convolution neural network (Conv1D) with 32 kernels where the size of each kernel is 1. Conv1D is well suited for sequential data; it learns patterns from the input data sequence and passes them to a long short-term memory (LSTM) layer that is configured with 250 LSTM units. Combining Conv1D and LSTM layers has shown significant improvement in performance when dealing with sequential data such as time series. LSTM hands the learned patterns down to a multi-head attention layer. The multi-head attention layer provides transformation on the sequential input of values to obtain distinct metrics of size h. Here, h is the number of attention heads that is set to 3 and the size of each attention head is also set to 3 because a number greater than 3 caused overhead and less than 3 caused performance degradation. The other parameters are left with their default values. Furthermore, we add custom attention to instruct the layers to focus and pay more attention to critical information of the input data sequence and capture the correlation between the input and output by computing the weighted sum of the data sequence. In addition, we add a dense variational layer (DVL) with 10 neurons that uses variational inference to

approximate the posterior distribution over the model weights. DVL is similar to a regular dense layer, but requires two input functions that define the prior and posterior distributions over the model weights. DVL allows our DSTT model to represent the weights by a distribution instead of estimated points.
DSTT also includes multiple dense and dropout layers. Each dense layer is strongly connected with its preceding layer where every neuron in the dense layer is connected with every neuron in the preceding layer. Each dropout layer instructs the DSTT model to randomly drop a percentage of its hidden neurons throughout the training phase to avoid over-fitting of training data.



Uncertainty Quantification

Quantifying uncertainty with a deep learning model hasbeen used in many applications such as medical image processing, computer vision, space weather and solar physics. Our proposed DSTT model contains a dense variational layer (DVL) that provides a weight distribution and multiple dropout layers that drop or turn off certain number of neurons during the training phase. Dropout is mainly used in deep learning to prevent over-fitting, where a trained model can be generalized for prediction instead of fitting exactly against its training data. With the dropout, the model's internal architecture is slightly different each time the neurons are dropped. This is an important behavior to the Monte Carlo (MC) class of algorithms that depends on random sampling and provides useful information. More details about uncertainty quantification can be found in our full paper at:
https://iopscience.iop.org/article/10.3847/1538-4365/ac5f56
(https://iopscience.iop.org/article/10.3847/1538-4365/ac5f56)

This notebook leverages python deep learning to describe the steps on how to use the DSTT tool to forecast the Dst index for 1 to 6 hours ahead.

Funding

This work was supported by U.S. NSF grants AGS-1927578 and AGS-1954737 and supported by NASA under grants 80NSSC18K1705, 80NSSC19K0068, and 80NSSC20K1282.

Keywords

keywords=["DST","Index","Forecasting", "Prediction", "Machine", "Learning","Solar",'Wind"]

Citation

To cite this notebook: Yasser Abduallah, Jason T. L. Wang, Prianka Bose, Genwei Zhang, Firas Gerges, and Haimin Wang. Forecasting the Disturbance Storm Time Index with Bayesian Deep Learning, available at: https://github.com/ccsc-tools/Dst-prediction/YA_01_ForecastingDstIndexBayesianDeepLearning.ipynb (https://github.com/ccsc-tools/Dst-prediction/YA_01_ForecastingDstIndexBayesianDeepLearning.ipynb).

Acknowledgements

We acknowledge the use of NASA/GSFC's Space Physics Data Facility's OMNIWeb service and OMNI data

Setup

Installation on Local Machine
Running this notebook in a local machine requires Python version 3.9.x with the following packages and their version:

Library	Version	Description
keras	2.6.0	Deep learning API
numpy	1.21.5	Array manipulation
scikit-learn	1.0.1	Machine learning
sklearn	latest	Tools for predictive data analysis
matlabplot	3.4.3	Visutalization tool
pandas	1.3.4	Data loading and manipulation
seaborn	0.11.2	Visualization tool
scipy	1.7.1	Provides algorithms for optimization and statistics
tensorboard	2.7.0	Provides the visualization and tooling needed for machine learning
tensorflow-gpu	2.6.1	Deep learning tool for high performance computation
tensorflow-probability	0.14.1	For probabilistic models

Library Import

The following libraries need to be imported.

In [1]:
```
#supress warning messages
import warnings
warnings.filterwarnings('ignore')
print('Importing packages..')
# Data manipulation
import pandas as pd
import numpy as np
import os
print('Packages imported')
#make sure the scripts are executed in the correct pacakage location.
if os.path.exists('DSTT_Package'):
    print('Changing working directory to DSTT_Package..')
    os.chdir('DSTT_Package')
import sys
sys.path.append('.')

```

```
Importing packages..
Packages imported
Changing working directory to DSTT_Package..
```

Data Processing and Analysis

The Dst index measurements used in this study are provided by the NASA Space Science Data Coordinated Archive. The data source provides other widely accessed data that are frequently used in solar wind analysis. The data source is being periodically updated with Advanced Composition Explorer (ACE).2 We used the Dst index data in the time period between January 1, 2010 and November 15, 2021. We selected the time resolution of the hourly average for the Dst index. We considred seven solar wind parameters, namely the interplanetary magnetic field (IMF), magnetic field Bz component, plasma temperature, proton density, plasma speed, flow pressure, and electric field. The total number of records in our dataset is 104,080. The Dst index values in the dataset range from 77 nT to −223 nT.

We divided our dataset into two parts: training set and test set. The training set contains 102,976 records from January 1, 2010 to September 30, 2021. The test set contains 1104 records from October 1, 2021 to November 15, 2021. The training set and test set are disjoint. The records are labeled as follows. Let t be a time point of interest and let w be the time window ahead of t, where w ranges from 1 to 6 hours for the short-term Dst forecasting studied here. The label of the record at time point t is defined as the Dst index value at time point $t + w$ for w-hour-ahead forecasting. Each record in the training set has eight values including the seven solar wind parameter values and the label of the training record. Each record in the test set contains only the seven solar wind parameter values; the label of each testing record in the test set will be predicted by our DSTT model.

Binder

This notebook is Binder enabled and can be run on mybinder.org (https://mybinder.org/) by using the image link below:

launch binder (https://mybinder.org/v2/gh/ccsc-tools/Dst-prediction/HEAD?labpath=YA_01_ForecastingDstIndexBayesianDeepLearning.ipynb)

Please note that starting Binder might take some time to create and start the image.

DSTT Workflow and Results

Data Preparation and Loading

The data directory inside the DSTT_Package folder includes all training and test data sets required to run the run the notebook. The files are loaded and used during the testing and training process.

Predicting with Pretrained Models

There are default and pretrained models that can be used to predict without running your own trained model. The models_directory is set to default_models which uses all pretrained algorithms.

In [2]:
```
#Test default models for 1-6 hours.
from DSTT_test import test

models_directory='default_models'
print('Test default models for Dst forecasting for 1-6 hours ahead.')
start_hour=1
end_hour=6
test(start_hour,end_hour+1,models_directory=models_directory)

```

```
39/100  [===== Uncertainty Quantification ======]  -   39/100 %
40/100  [===== Uncertainty Quantification ======]  -   40/100 %
41/100  [===== Uncertainty Quantification ======]  -   41/100 %
42/100  [===== Uncertainty Quantification ======]  -   42/100 %
43/100  [===== Uncertainty Quantification ======]  -   43/100 %
44/100  [===== Uncertainty Quantification ======]  -   44/100 %
45/100  [===== Uncertainty Quantification ======]  -   45/100 %
46/100  [===== Uncertainty Quantification ======]  -   46/100 %
47/100  [===== Uncertainty Quantification ======]  -   47/100 %
48/100  [===== Uncertainty Quantification ======]  -   48/100 %
49/100  [===== Uncertainty Quantification ======]  -   49/100 %
50/100  [===== Uncertainty Quantification ======]  -   50/100 %
51/100  [===== Uncertainty Quantification ======]  -   51/100 %
52/100  [===== Uncertainty Quantification ======]  -   52/100 %
53/100  [===== Uncertainty Quantification ======]  -   53/100 %
54/100  [===== Uncertainty Quantification ======]  -   54/100 %
55/100  [===== Uncertainty Quantification ======]  -   55/100 %
56/100  [===== Uncertainty Quantification ======]  -   56/100 %
57/100  [===== Uncertainty Quantification ======]  -   56/100 %
58/100  [===== Uncertainty Quantification ======]  -   57/100 %
```

Plotting the Pretrained Models Results

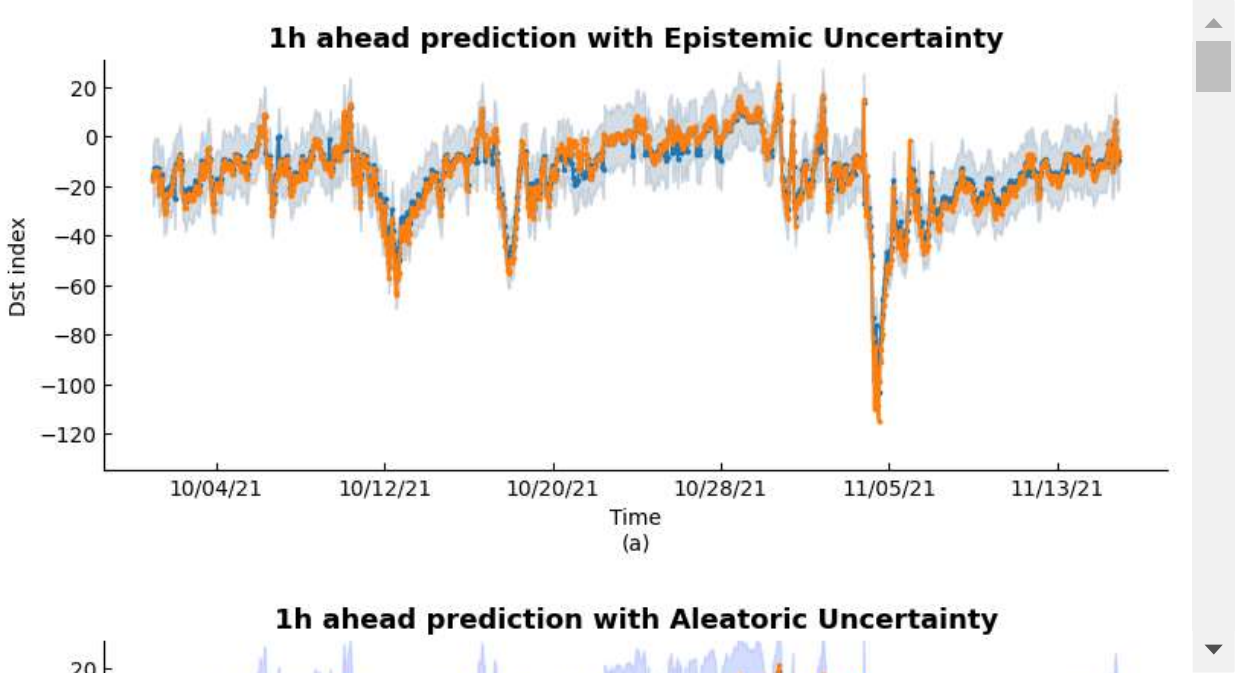
The prediction result can be plotted using the function plot_figures. It uses the results produced by the model from the "default_results" directory.

In [3]:
```
from DSTT_plot_results_figures import plot_figures

figures_dir='default_figures'
results_dir='default_results'
print('Plotting figures for default models results for Dst forecasting for 1
start_hour=1
end_hour=6

plot_figures(start_hour, end_hour+1,show_figures=True,figures_dir = figures_

```



DSTT Model Training and Testing Example

DSTT Model Training with Sample Data

Here, we show how to train the model with sample data example. In this exmaple, we show how to train the model for tiwe window h = 1 to 6 hour ahead.

In [4]:
```
#Training for FC_S 12-72 hours on sample data.
print('Loading the train_model function...')
from DSTT_train import train_model
print('Train custom model for h=1-6')
start_hour=1
end_hour=6
#set the number of epochs=100
epochs=100
train_model(start_hour,end_hour+1,epochs=epochs)

```

Predicting with Your Trained DSTT Model

To predict the testing data using the model you trained above, make sure the models_directory variable is set to models:
models_directory='models'

Note: this is training job is only an example that uses less, training proceses, epochs, therefore the results and performance metrics are not comparable to fully developed pretrained and default models.

In [5]:
```
#Test default models for 1-6 hours.
from DSTT_test import test

models_directory='models'
print('Test your trained models for Dst forecasting for 1-6 hours ahead.')
start_hour=1
end_hour=6
test(start_hour,end_hour+1,models_directory=models_directory)

```

Plotting the Results for Your Trained Model

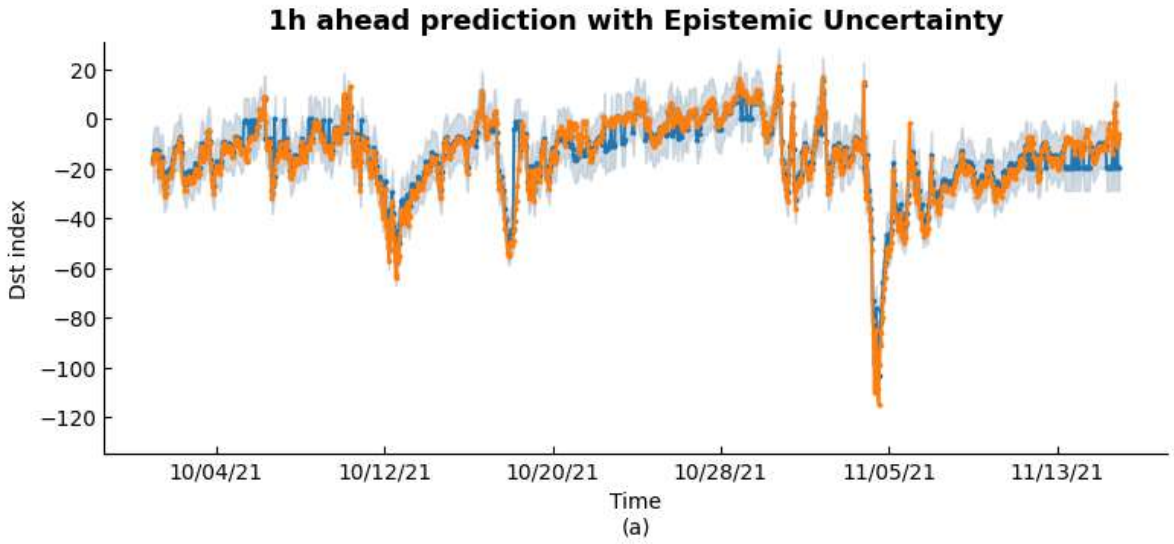
The prediction result can be plotted using the function plot_figures as shown in the following example. The example plots your trained model results for h=1-6 hours ahead.

In [6]:
```
from DSTT_plot_results_figures import plot_figures

figures_dir='figures'
results_dir='results'
print('Plotting figures for default models results for Dst forecasting for 1
start_hour=1
end_hour=6

plot_figures(start_hour, end_hour+1,show_figures=True,figures_dir = figures_


```



Timing

Please note that the execution time in mybinder varies based on the availability of resources. The average time to run the notebook is 10-15 minutes, but it could be more.

Conclusions ¶

The disturbance storm time (Dst) index is an important and useful measurement in space weather research, which is used to understand the severity of a geomagnetic storm. The Dst index is also known as the measure of the decrease in the Earth's magnetic field. In this paper, we present a novel deep learning model, called the Dst Transformer or DSTT, to perform short-term, 1-6 hour ahead predictions of the Dst index. Our empirical study demonstrated the good performance of the Dst Transformer and its superiority over related methods.

Our experiments were based on the data collected in the period between January 1, 2010 and November 15, 2021. The training set contained hourly records from January 1, 2010 to September 30, 2021. The test set contained hourly records from October 1, 2021 to November 15, 2021. To avoid bias in our findings, we performed additional experiments using 10-fold cross validation (CV). For the CV tests, we used the original data set described above and another data set ranging from November 28, 1963 to March 1, 2022 that has 510,696 records.
In addition, we generated synthetic data with up to 1.2 million records to further assess the performance and stability of our DSTT model. With the 10-fold CV tests, the data was divided into 10 approximately equal partitions or folds. The sequential order of the data in each fold was maintained. In each run, one fold was used for testing and the other nine folds together were used for training. There were 10 folds and hence 10 runs. We computed the performance metrics including RMSE and R^{2} for each method studied in the paper in each run. The means and standard deviations of the metric values over the 10 runs were calculated and recorded. Results from the 10-fold CV tests were consistent with those reported in the paper. Thus we conclude that the proposed Dst Transformer (DSTT) is a feasible machine learning method for short-term, 1-6 hour ahead predictions of the Dst index. Furthermore, our DST Transformer can quantify both data and model uncertainties in making the predictions, which can not be done by the related methods.

Our work focuses on short-term predictions of the Dst index by utilizing solar wind parameters. These solar wind parameters are collected by instruments near Earth and are suited for short-term predictions of the geomagnetic storms near Earth. When using the solar wind parameters to perform long-term (e.g., 3-day ahead) predictions of the Dst index, the accuracy is low. In future work, we plan to perform long-term predictions of the Dst index by utilizing solar data collected by instruments near the Sun. The solar data reflects solar activity, which is the source of geomagnetic activity. We plan to extend the Bayesian deep learning method described here to mine the solar data for performing long-term Dst index forecasts.

References

1. Forecasting the Disturbance Storm Time Index with Bayesian Deep Learning
Yasser Abduallah, Jason T. L. Wang, Prianka Bose, Genwei Zhang, Firas Gerges, and Haimin Wang
https://iopscience.iop.org/article/10.3847/1538-4365/ac5f56 (https://iopscience.iop.org/article/10.3847/1538-4365/ac5f56)

2. DeepSun: Machine-Learning-as-a-Service for Solar Flare Prediction
Yasser Abduallah, Jason T. L. Wang and Haimin Wang
https://doi.org/10.32473/flairs.v35i.130564 (https://doi.org/10.32473/flairs.v35i.130564)
3. Tracing Hα Fibrils through Bayesian Deep Learning
Haodi Jiang, Ju Jing, Jiasheng Wang, Chang Liu, Qin Li, Yan Xu, Jason T. L. Wang, and Haimin Wang
https://doi.org/10.3847/1538-4365/ac14b7 (https://doi.org/10.3847/1538-4365/ac14b7)
4. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning
Yarin Gal and Zoubin Ghahramani
https://dl.acm.org/doi/10.5555/3045390.3045502
(https://dl.acm.org/doi/10.5555/3045390.3045502)

In [ ]:
```
1
```