Technical Report: Demo Tasks

Space @ VT's GSoC 2018 Program

Sneha Singhania*

As part of my GSoC 2018 application, this report outlines the results obtained solving demo task 3[1], "Correlations between OMNI Data and Dst Index Date" and demo task 4[2], "SuperDARN HF Radar Data Clustering" using machine learning. A deep predictive model using LSTMs was designed and trained in TensorFlow for task 3 and clustering algorithms (K-means, Agglomerative and GMM) were used in task 4.

1 Correlations between OMNI Data and DST Index

1.1 Literature Review

During my exploratory analysis of the DST Index time series curve, I observed a sharp decrease in value (highly negative DST index) during the occurrence of a solar storm. Since solar storms are rare events, a simple time series predictive model would not be enough to capture the correlations at these rare events. I therefore shifted my focus to deep learning. In the paper, "Time-series Extreme Event Forecasting with Neural Networks at Uber" [1], a deep learning model is designed to predict Uber cab demand during extreme rare events (like New Year's Eve). Taking inspiration from this paper, a similar model using stacked LSTMs (for greater depth) was designed and trained in Tensorflow [2] to predict DST Index data especially during rare events like solar storms.

1.2 Deep-LSTM Model

LSTMs have proven to be a scalable deep learning sequence model for modelling time series data. The ability of LSTMs to capture long term dependencies in the input data by modelling the non linear feature interactions makes it a useful model in the temporal domain. A model architecture as visualized in figure 1 was used. To obtain the input and output data for training, a sliding window approach was used with a look back window of 10 time steps and a forecast window of 1 time step. Three different models were trained on different input-output pairs. The first model `DST to DST` used previous DST data to forecast DST data, the second model `DST plus IMF to DST` used both previous DST data and previous IMF data to forecast DST data and the third model `IMF to DST` used only previous IMF data to forecast DST data.

The `Deep-LSTM` models accept a sequence of inputs from the look-back window of size 10, which is passed to a stacked LSTM layer of depth 2 and 100 hidden units each, followed

*sneha.a@iiitb.org

[1] Code for task 3

[2] Code for task 4

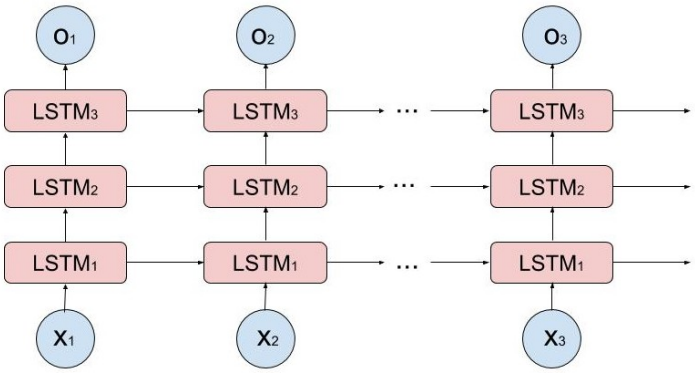

Figure 1: An Example of a Stacked LSTM Architecture

by a dropout layer to prevent overfitting and finally a dense layer regressor to predict the next DST index. The model architecture in the form of the TensorFlow computation graph is shown in 2. $50,000$ samples of time series data were taken from the year 1963 onwards and split into a 80-20 train-test split. Although only the previous 10 immediate values are used by the LSTM to predict the next value, I also wished to use a summary of the previous values since the beginning of time. To do so, I chose not to shuffle the dataset to preserve time context. Therefore the internal weight matrix of the LSTM will store a summary of all previous values since the beginning. I verified this hypothesis by experimenting with models trained on shuffled and the non-shuffled dataset, non-shuffling resulting in a lesser mean squared error on the test data set.

The models were training using early stopping to prevent overfitting and generally ran for around 50 epochs. The models were trained on the NVIDIA Tesla V100 GPU. Mean squared error was used to define the loss function with Adam as the gradient descent algorithm. All code was written in Python and `Deep-LSTM` was designed and trained in Tensorflow. Two Python scripts were written containing a scraper to obtain IMF and DST Index data from the website (`get_data.py` and `combine_data.py`). The three models files are `model1.py` to `model3.py`. Additionally, a helper `utils.py` was used to provide the functionality of a batch generator. The file `visualize.py` plot the ground truth curves for DST and IMF data with the predicted DST curve.

1.3 Results

My Deep-LSTM network performed highly accurately and surprisingly well to predict rare events in the DST Index data (ex: highly negative values during solar storms) using only the IMF data (B_z component) using `model3` as shown in 3. Notice how the predicted DST index also drops sharply with the groundtruth DST index. Since this drop in the groundtruth DST index occurs during solar storms, it is rarely present in the training data, making it hard to predict. This is expected since the DST Index is prone to disturbances in space weather, as well as the B_z value is perpendicular to the ecliptic and is created by waves and other disturbances in the solar wind. After visually plotting both B_z and the DST Index, we can observe a corresponding increase in B_z values for decrease in DST Index values (especially during solar storms) as shown in 4.

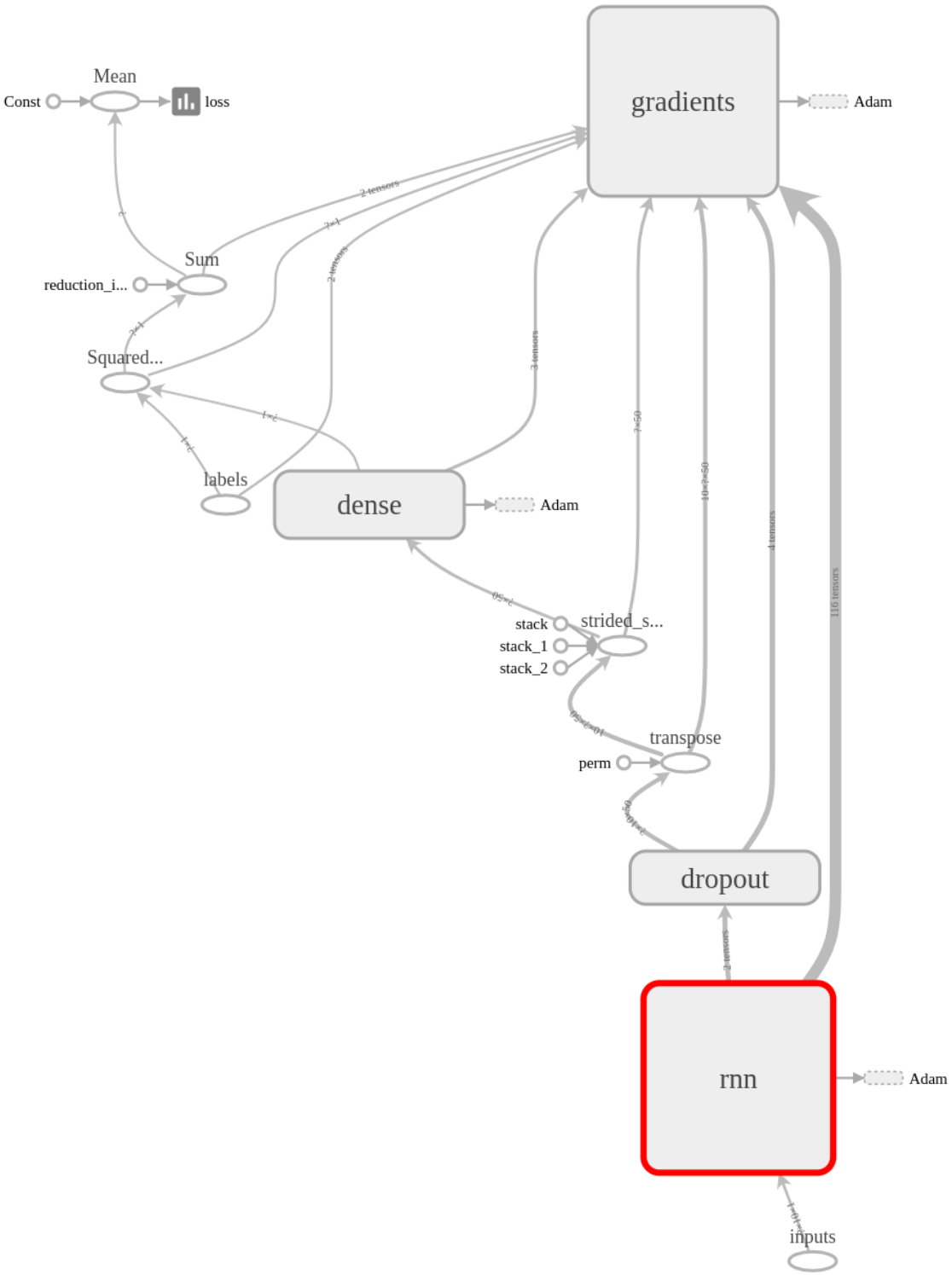

Figure 2: Computation Graph of Deep-LSTM in TensorFlow

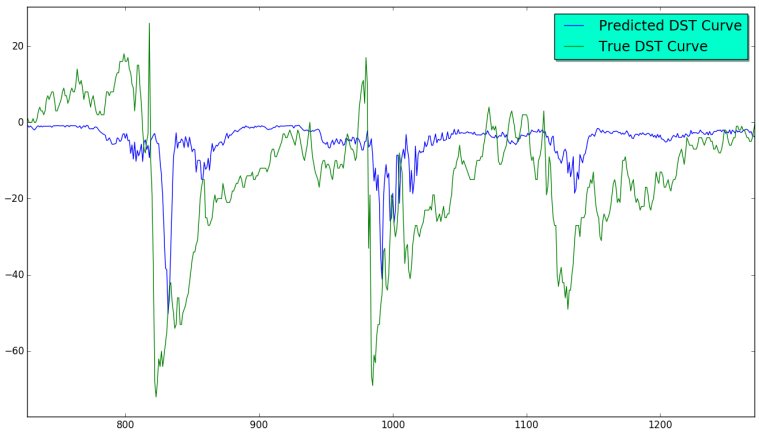

Figure 3: Ground Truth and Predicted DST Index

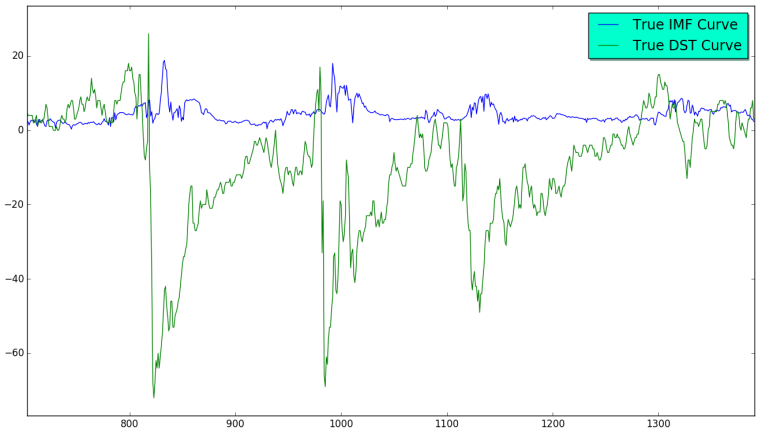

Figure 4: Correlation between DST Index with IMF Time Series

2 SuperDARN HF Radar Data Clustering

Literature Review

For this task, I read the paper, "Mapping ionospheric backscatter measured by the SuperDARN HF radars – Part 1: A new empirical virtual height model" [3] which outlined the major classes of backscatter echoes, namely ground scatter and ionospheric scatter. Ionospheric scatter can be further classified as $\frac{1}{2}$ hop E-region scatter, $\frac{1}{2}$ hop F-region scatter and $1\frac{1}{2}$ hop F-region scatter. For this task I will aim to complete a simpler classification task assuming only two classes, namely, ground scatter and ionospheric scatter.

2.1 Data Collection

I used the beam data from the Saskatoon radar since a large proportion of the backscatter measured by the Saskatoon radar have reliable elevation angle measurements with a good coverage in range, frequency, and magnetic local time (MLT) as mentioned in the paper [3]. I established a data pipeline to the Virginia Tech server to obtain the beam data using `DaVitpy`. During the data collection process, including the installation and usage of `DaVitpy`, I managed to solve a few issues and have communicated them on Git. I collected data on $1-1-2011$ for 1 hour from the Saskatoon radar and processed it using the `davitpy.pydarn.proc.fov.update_backscatter` module. The data was preprocessed to remove `nan` value elevation angles from the data. Further, `preprocessing.scale` from `sklearn` was used to scale the data since the clustering algorithms especially k-means is sensitive to normalization.

2.2 Clustering Models

The following attributes were used to cluster the data:

1. Power
2. Doppler velocity
3. Range gate
4. Spectral width
5. Elevation angle

The clustering algorithms used were K-means, Agglomerative and GMM. The K-means algorithm tries to form clusters of equal variance by iteratively minimizing the within-cluster sum-of-squares measure. K-means scales well to large number of samples and can be used to cluster flat geometry. I used the `k-means++` version of `sklearn` with 10 as the number of initializations. The optimal number of classes k was found using the elbow curve method giving the optimal k equal to 5. The elbow plot is shown in 5. I also tried out the Agglomerative clustering algorithm which performs a hierarchical clustering using a bottom-up approach. The `ward` version of Agglomerative clustering which minimizes the sum of squared differences within all clusters was used. The GMM algorithm using Mahalanobis distances is good for density estimation and flat geometry. These other algorithms were tried out in an explorative spirit.

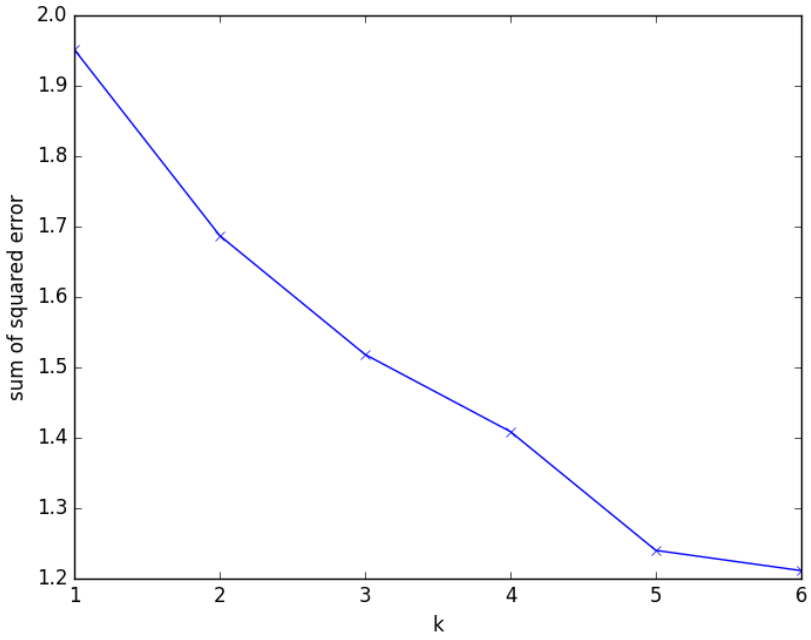

Figure 5: Elbow Plot to Find Optimal k

2.3 Results

The resulting cluster visualizations are shown in the following figures obtained using K-means 6, Agglomerative 7 and GMM 8 clustering algorithms. The cluster with the lesser mean Doppler velocity and lesser mean spectral width is the ground scatter as mentioned in the paper [3]. For these figures the number of clusters to form, k, was set to 2 for better visualization.

3 Biography

I'm Sneha Singhania, an integrated master's student in Computer Science at IIIT-Bangalore, India. I'm pursuing my master's thesis in deep learning. I'm passionate about research (both hands on coding and theoretical) in Artificial Intelligence. I have extensive experience in AI through a paper publication in an A rank AI conference (ICONIP 2017), working in two AI startups, VideoKen founded by the director of IBM Research and Aganitha Cognitive Solutions, and real life projects on Kaggle. More: LinkedIn.

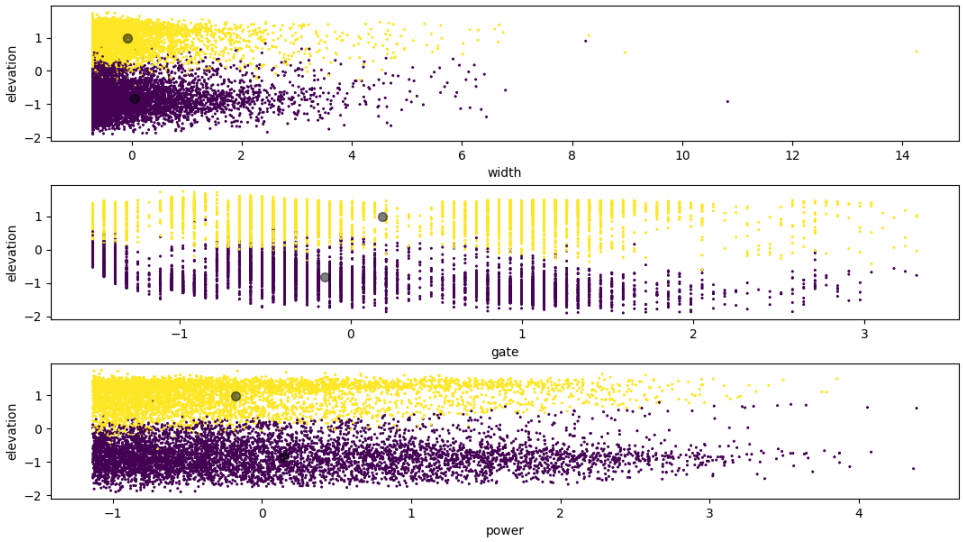

Figure 6: Cluster Output using K-means Clustering

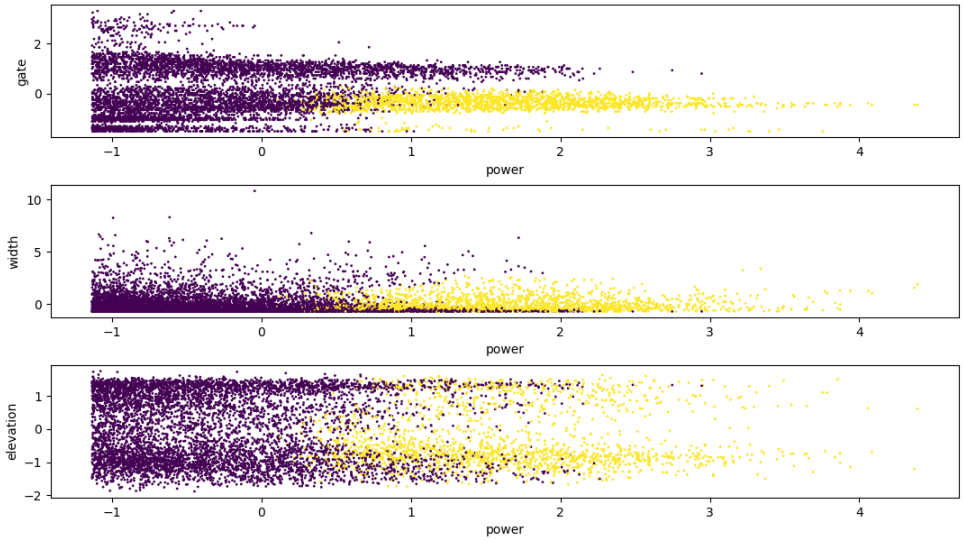

Figure 7: Cluster Output using the Agglomerative Clustering

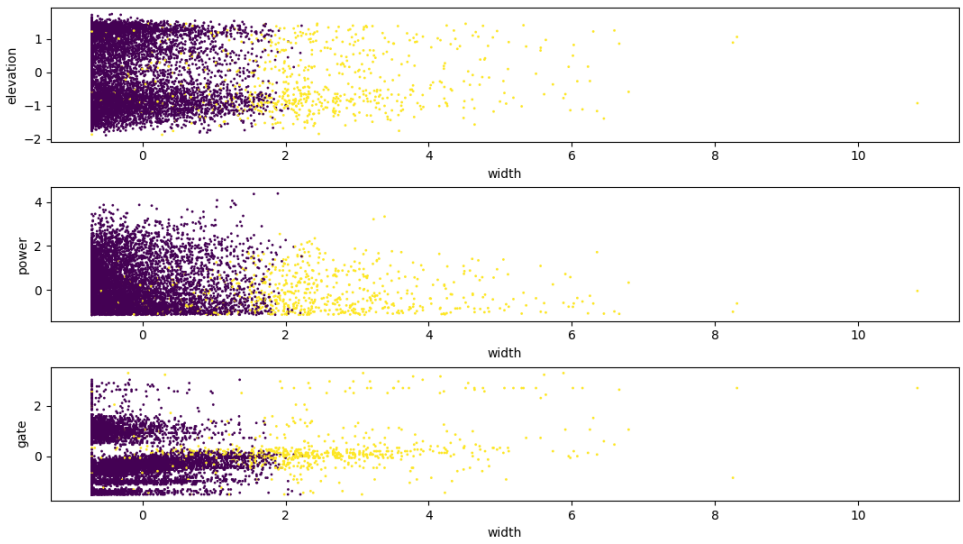

Figure 8: Cluster Output using GMM

Bibliography

[1] Nikolay Laptev et al. "Time-series extreme event forecasting with neural networks at uber". In: *International Conference on Machine Learning*. 2017.

[2] Martin Abadi et al. "TensorFlow: A System for Large-Scale Machine Learning." In: *OSDI*. Vol. 16. 2016, pp. 265–283.

[3] G Chisham, Tim K Yeoman, and GJ Sofko. "Mapping ionospheric backscatter measured by the SuperDARN HF radars–Part 1: A new empirical virtual height model". In: (2008).