Deoxygenation Indicators and Forecasting Executive Summary

September 2024

Burning Hydrogen Team:
Alexander Ramussen, Afsin Ozdemir, Sanjay Kumar, Rafatu Salis, Touseef Haider

1 Objective

The 200 billion dollar fishing industry is predicted to see a reduction of 40 to 60 billion dollars due to climate change and the deoxygenation of the ocean. This motivates our project to determine the main factors that indicate an upcoming oxygen concentration drop in the ocean. Our objective is to determine whether we can predict deoxygenation of the ocean on long time scales (30+ days).

2 Stakeholders:

Since the fishing industry is of global importance, our primary stakeholders are economies and consumers reliant on the commodity. Through our project, we hope to achieve the following performance indicators: (1) Discovering early indicators to help mitigate future oxygen loss and (2) Accurately forecasting future levels oxygen concentrations as a metric for the health of the fishing industry.

3 Data Collection, Description and Cleaning

We collected data from Copernicus Marine Service (`https://marine.copernicus.eu/`) and from GEBCO (`https://www.gebco.net/`. From these databases we used the following variables: depth, O2 concentration, chlorophyll-a concentration, NO3 concentration, PO4 concentration, Si concentration, temperature, and salinity. These variables were measured daily, over the region with longitude from -100 to 0 degrees and latitude from -35 to 15 degrees. Each of these variables were averaged over a grid of $5°$ longitude $\times$ $5°$ latitude regions. Finally, we calculated lag variables and lead variables for each of the variables (i.e. lag indicates a value shifted back in time and lead indicates a value shifted forward in time). We used 10 days of lag variables and 30 days of lead variables based on a lag analysis carried out by one of our team members. The result is `dataset.csv` in `numpy_dataset`.

We transformed this into a numpy array using inside the folder `numpy_dataset`. This is an array of shape

```
num_dates x num_days x num_channels x height x width = 813 x 41 x 8 x 10 x 20.
```

Here, `dataset.csv` contains data from 813 dates. Here, 41 comes from the sum of amounts of lag, lead, and the current day. For each date, and a specified number of days of lag or lead we have 8 channels of data for our our 8 ocean variables: O2, PO4, etc. Height is 10, for the number of bins of latitude between -35 degrees and 15 degrees at 5 degrees per bin. Width is 20, for the number of bins of longitude. Finally, the variables were normalized by subtracting mean and dividing by standard deviation. In addition, missing values were imputed using the mean value 0.

4 Models

4.1 XGBoost

A simple XGboost model was fit to predict the O2 on a given day given the other ocean variables. In addition, by adding lag features to the dataset, we evaluated the impact of the variables on previous days to the O2 on the given day.

4.2 LSTM

LSTM cells are particularly useful for handling time series data because they can retain information over long time periods and handle the vanishing gradient problem. We fed 7 days of sequential data of oxygen to our model and asked for $10, 30$, and 60 days of prediction of the oxygen value. We add a dropout layer to prevent overfitting by randomly dropping a portion of the neurons (0.2) during training. We also used a learning rate scheduler, which helped the training. The command `scheduler.step` is responsible for checking if the loss has plateaued. If the loss hasn't improved for patience (5) epochs, it reduces the learning rate by multiplying it by factor (0.5). We observed that as the number of days in the future increases, the mean square error of the true value of oxygen against the predicted value increases. We also observed that if we fed 14 days of sequential data, the model behaved better than when feeding 7 days of sequential data.

4.3 ConvLSTM models

Our ConvLSTM model is implemented in `models/convlstm`. Convolutional LSTM (ConvLSTM) cells were introduced in [1] in order to take advantage of the memory properties of LSTM cells while using convolutions to deal with image data. ConvLSTM cells were originally designed to deal with spatiotemporal data with many variables (in particular, predicting precipitation given many variables measured over a wide spatial region and over long regions of time). Thus, may be well-suited to dealing with our spatiotemporal data.

Our ConvLSTM cells are implemented in `https://github.com/ndrplz/ConvLSTM_pytorch`. Our ConvLSTM model architecture is also modified from `https://github.com/ndrplz/ConvLSTM_pytorch`. It consists of several ConvLSTM cells stacked on top of each other. Images (each with 8 channels) in a sequence are fed into the first ConvLSTM cell one at a time. In the end, the hidden states of these ConvLSTM cells are concatenated into a single tensor in order to form the output of the model, allowing the model to yield many days' worth of images as output. Finally, we added dropout and batch normalization between the layers to prevent overfitting.

Our EncoderDecoderConvLSTM model is implemented in `models/convlstm`. The architecture was modified from `https://github.com/holmdk/Video-Prediction-using-PyTorch`. In this case, the architecture consists of an encoder, a decoder, and a single 3-dimensional convolutional layer to produce the output. The encoder and decoder are both made of stacked ConvLSTM cells. The decoder receives the final hidden state of the encoder after the encoder has received a sequence of images. The encoder then outputs a sequence of vectors, each being predicted based on the last. The whole sequence of vectors is finally transformed into an image by stacking and feeding into a 3-dimensional convolutional cell.

In both our models based on ConvLSTM cells, the predictions of the baseline model are added to the output. This allows the models to fit to the errors of the baseline model.

4.4 Conv3d stack

While our ConvLSTM cells use 2-dimensional convolutions to process images one at a time, an entire sequence of images can be transformed using a 3-dimensional convolution. This is the basis for our Conv3d model, implemented in `models/conv3d`. A block for our Conv3d layer processes a sequence of images and outputs the sum of the input, plus an output tensor of the same shape as the input. Thus it is a residual block. Each such residual block also utilizes batch normalization and dropout to prevent overfitting.

4.5 Gated Recurrent Unit

GRUs have gating mechanisms that allow them to capture the temporal dependencies in sequential data. These gating mechanisms also help mitigate problems associated with long-term dependencies. Our model consist of 2 layers: 64 units in the first layer and 32 units in the second layer. To reduce overfitting, dropout layers with a dropout rate of 0.2 were incorporated. We used a sequence of 30 days of data to forecast future oxygen levels for 10 and 30 days into the future. We trained the model for 15 epochs, optimized it using the Adam optimizer with a learning rate of 0.001, and used mean squared error (MSE) as the loss function.

5 Results

SHAP values were computed using an XGBoost model to predict O2 on a given day based on the other ocean variables. These can be viewed in the `models/xgboost` folder and in our final presentation. They indicate that temperature, salinity, and phosphate concentration correlate strongly with O2 concentration. In addition, the value of these variables up to 30 days in the past strongly influences O2 concentration.

A baseline to compare against our neural network time series models is the NoModel class. This class predicts that oxygen will stay the same for all 30 days into the future, for each $5^\circ \times 5^\circ$ region in our grid. The ConvLSTM, EncoderDecoderConvLSTM, and Conv3dModel classes were trained using an Adam optimizer.

	Baseline	Conv3d	ConvLSTM	Encoder-decoder
Overall MSE	0.06396	0.06258	0.05177	0.04950
Error day 1	0.001652	0.001657	0.003457	0.001904
Error day 30	0.1429	0.1387	0.1107	0.09902

We see that ConvLSTM and EncoderDecoderConvLSTM both outperformed the baseline significantly, especially for predictions on later days (with the encoder-decoder error being about 30% better than baseline on day 30). Gifs comparing the predictions of ConvLSTM and EncoderDecoderConvLSTM to the true values of O2 can be found in `models/comparison/gifs`. Graphs of the predicted O2 versus actual O2 for randomly selected dates and regions in our grid can be found in `models/convlstm/analysis` and `models/encoder_decoder/analysis`.

6 Conclusions

From our analysis, we have found significant correlation between oxygen loss and the features salinity, phosphate, nitrate, and temperature. Additionally, we have successfully built several models which significantly outperform our baseline model in predicting the oxygen level at future dates. In future work, we hope to consider the following: confounding variable search, further hyperparameter tuning, extending the analysis to a larger region, fine-tuning pre-trained models, and automating the model for continuously updated data.

7 Acknowledgements

We thanks the Erdos Institute and NewAtlantis Lab for giving us the opportunity and training. We also thank our mentors Stanely Bishop, Justin Kim and Malik Cameron.

References

[1] X. Shi, Z. Chen, H. Wang, D-Y Yeung, W-k Wong, W-c Woo; Convolutional LSTM network: a machine learning approach for precipitation nowcasting; Advances in neural information processing systems 28 (2015).