Imperial College London
Department of Earth Science and Engineering
MSc in Applied Computational Science and Engineering

Independent Research Project

ARGO Trainer: Developing of a new software platform to annotate, visualise and analyse ARGO floats data

by

Vagif R. Aliyev

Va719@imperial.ac.uk

GitHub login: acse-va719

Supervisors:

Dr. Yves Plancherel

Ophelie Meuriot

August 2020

Abstract

In the oceanography sector, problems such as lack of data has been overcome through usage of ARGO floats. However, there is now a lack of tools to deal with the ever-increasing volume of observational data available. The goal of the project is to take advantage of this data revolution to apply modern computing techniques to provide new user-friendly software to democratise access to this new ARGO data stream. ARGO trainer fills the gap in the tools available in the oceanographic community to help annotate, visualise and analyse argo data. Providing a platform where user can freely label parts of data interested in and define features, apply their own predictive algorithms for their specific needs, observe the predictions both in graphical and numerical format. It is a python package, written in modular structure for users to implement and test out various algorithms with ease.

Keywords Argo Floats, Data Labelling, Mapping, Feature extraction, Predictor

Acknowledgment

Firstly, I would like to express appreciation to my supervisor: Professor Yves Plancherel. Also, Ophelie Meuriot for providing user feed back through all the stages of the project and helping form a platform that benefits the researchers in the sector. Their constant feedback and enthusiasm has helped me overcome challenges and stay motivated during the difficult pandemic period. Finally, I would also like to offer my thanks to my parents, for their continual and unlimited support throughout my life.

Contents

List of Figures

List of Tables

1 Introduction

Advancement in Oceanography tools is essential as oceans cover 70 % of earths surface, absorbs 1/3 of carbon dioxide emissions and plays the major role in protecting the world from the effects of greenhouse (Le Quéré et al. 2012). Obtaining ocean data and analysing it has become even more essential as these factors have become more threatening. One of the leading providers of ocean data is ARGO floats with over 3200 floats (growing) reporting about 10,000 profiles per month, providing a range of properties such as salinity, temperature, depth and geographical position (Argo 2020*d*). This data enables scientists to understand how the state of oceans changes over time and helps humanity monitor issues such as sea level rise, ocean heat content and warming, ocean circulation and the water cycle (Argo 2020*e*).

Having access to more information is beneficial, however it is difficult to navigate through so much data without assistance. The climate science community is always in need of new set of tools to deal with a very large, ever-increasing volume of observational and computer model data (Jones et al. 2019). The overall objective of this project is to aid and facilitate analysis of these data for researchers in this field. Specifically, the goal is to deliver a python package able to help the oceanographic community to annotate, visualise and analyse ARGO data. Before the publication of this software, there was no platform or mechanism to be able to record knowledge of studies as processed data, by labelling or predictive algorithms, as there was no defined architecture to learn from each others analysis other than publications. To address that problem, the main goal of the project is to provide a platform for researches across the field to share their data analysis and learn from each other in an interactive way.

The software created (ARGO Trainer) provides automated method of creating a local database for argo data, visualisation platform for graphical data and geographical mapping of sensor locations. Most importantly, allows the users to freely label data by interacting with graphs via multiple tools such as selector, zooming, panning and saving. Coding has been implemented in a modular way, thus user can insert their own prediction algorithms easily, observe its results through authenticator and further provide more training data till satisfactory results have been achieved. The final software is a product of constant user feed-back provided, matching the needs of researchers in the field of oceanography.

2 Existing Argo Infrastructures

2.1 Argo Floats

Argo floats was first deployed in 1999 to eliminate the lack of data in the field and efforts are still being put to increase the number of floats (Roemmich 2012). Now there is an abundance of data available to study the ocean behaviour in more depth. Floats work autonomously across all the oceans and at minimum provide data on the temperature and salinity of the ocean by periodically taking vertical profiles. The sensors live down 2,0000 m deep and every 10 days rises back to surface while taking a vertical profile of the water column along the way.

These readings are then transmitted via satellite, gone through Quality Control (QC) by experts at Brest, France and Monterey, California and afterwards made available for public use (Argo 2020*a*). The QC flags helps users identify which data they should use for their research. For this purpose, we are going to use data with QC flag of 1 which indicates good data that is statistically consistent and contain reasonably small errors (Argo 2020*c*).

All the data used is freely available without restriction. It's use should be acknowledged as such: " These data were collected and made freely available by the International Argo Program and the national programs that contribute to it. (http://www.argo.ucsd.edu, http://argo.jcommops.org). The Argo Program is part of the Global Ocean Observing System. "

2.2 Argo related Software

Readily accessing argo data was an issue however recently developed argopy python library eases the Argo data access, manipulation and visualisation (Maze. 2020*a*). This library was incorporated to access the data and study the visualisation aspects however the latter part was found lacking and will be the one of the focuses of this research to fill that gap in the community.

There has been multiple projects that use Matlab to create visualisation tools to fill this gap such as "MPV: Matlab Profiles Visualization" (Maze. 2020*b*) however none of them provide a platform where the user can freely label data, apply prediction algorithms and observe their performance. The algorithm can be an essential tool in automated data labelling and analyse possible outliers in QC.

From user feedback on information displayed, implementing a new mapping tool that can simply provide the user with visual aid where the float lies was deemed as a necessary aid since the behaviour of data is highly depended on which body of water it resides in. There are few alternatives already available however it does not provide the same quality and ability to manipulate the data freely by the user. (Argo 2020*b*) The main priority of the mapping tool is to not make it too complicated so that no maintenance will be required as in the case of the other alternatives available.

2.3 Data Labelling

Currently, there is no labelled data available on Argo data as there is no software that enables this to be created and shared easily. Lack of labelled data also makes it very difficult for scientist to apply complex predictive algorithms such as machine learning as it requires a large database of labeled data to be able to make successful predictions. The first issue that needs to be solved is to tackle the lack of data by providing a mechanism to harvest expertise so that a predictive algorithm can be developed in the future to leverage machine learning. Machine learning is very desirable as it can decrease the need for human interaction with data and quality control protocols.

3 Argo Trainer

The software has been created as a standalone work with the exception of usage of argopy to access the online database. The primary goal of the project is to provide user with a platform that enables them to label data freely and test out predictive algorithms easily. The current algorithm is designed as a simple placeholder, modular design of the programme allows for more complex algorithms to be placed instead with ease. The algorithm can be trained by the users to detect similar patterns in salinity and temperature across other current and historical data. This will be beneficial to scientist to observe how one part of the ocean can affect other parts across different or same timeline. All of the features implemented have been created with the one main goal in mind, which is to to facilitate data analysis. Main driving factor for design and feature creation has been user feedback.

The software delivered as part of this project can be divided into 3 main modules: data retrieval, labelling and processing. A general flowchart can be seen below:

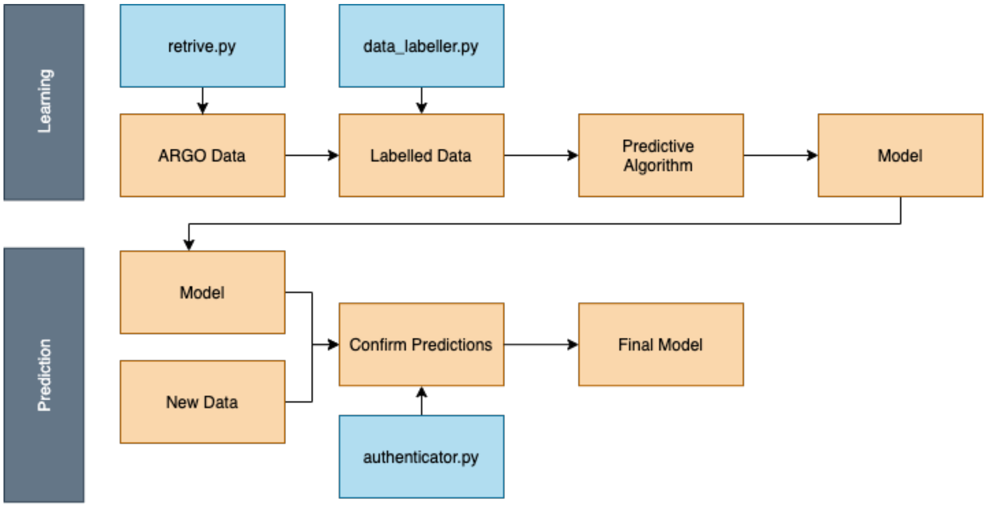


Figure 1: General Flowchart

3.1 Retrieval

Initially, the requested float and profile data was being downloaded from online database, used for graphing and afterwards discarded. However, this caused few issues: time taken to download the data was seen as an issue by user testing and also the data would later have to be downloaded again during the training phase. Thus, storing the files locally was deemed as a better option. Currently, the database is 2.54 GB however if the size is deemed as too much for the user, they can decrease the number of floats they would like to work with. For users without enough space to store data base locally and option of retrieving data online is presented. However, this slows down the system considerably. The training for salinity minimum takes on average about 7 seconds on local and about 4 minutes using online data base. This is a substantial difference, thus a local data base should be preferred more.

With the help of argopy, the downloading of the data has been automated using a script. No other argo related piece of research has been implemented as part of this work. Since the data set of available argo floats is very large, the user can specify which exact float data they would like to download and work on. As mentioned before, only data with QC flag of 1 (good) is downloaded for statistical consistency. If any problem occurs, such as due to corrupted files, in successfully downloading the data a notification pops up to notify the user of the argo float id that caused the problem. The addition to the list can be made and already downloaded files will not be downloaded again.

Instructions on how to run the software, and details on how to alter them is included in detail inside the GitHub repository README.md file.

3.2 Labelling

Data labelling allows the user to create a data set of the features they are interested in. At startup, user can enter details of their name, feature interested in and number of samples they wish to examine.

Afterwards, two subplots are generated. Firstly, a salinity against temperature graph of random float and profile is drawn with scatter points colour coded by pressure. Since the behaviour of floats is highly dependent on its location a fully interactive map is also implemented as the second subplot, showing the location of current float and also the ones previously selected which can be seen in Figure 4.

The feature of interest can then be selected using a rectangular tool which is activated on launch or re-activated by key press of ”T”. Once the feature is highlighted the user has 2 choices to label the selection as good by pressing the ”G” key or as bad by pressing ”R” key. Colour coded as green and red respectively in the map subplot. While the current float examined is labeled with a white dot. Data with anomalies should be labelled as red and data to be used for prediction by green. In the scenario where the user closes the application without making any selection, no data will be entered to the database and the next random float will be presented. The selection made is also displayed on the screen for the user. The key presses have been designed not to be case-sensitive.

Once the requested sample size has been reached, the user entries are presented to the user and given an option to name the CSV file where the selections will be stored for later usage in training. The name consist of the username initially entered and their entry. An example of the CSV file generated is presented below in Table 1.

Table 1: vagif_atlantic.csv , Output from labeler.py

Feature	Float	Profile	Colour	x1	x2	y1	y2	lat	long
min	1901341	95	green	34.446	34.542	4.364	6.840	-23.053	58.507
min	1901324	284	green	34.298	34.395	4.000	5.549	-32.309	-1.517
min	1901685	20	green	34.469	34.606	4.374	5.943	-0.523	-17.454
min	1901685	45	red	34.416	34.512	4.713	5.977	-1.12972	-22.66678

During the user testing phase, one of the complaint was the time it took generate the graphs which averaged about 30 seconds. This was seen as too long by the user, thus few adjustments was made to speed up the process. As mentioned previously, a local database was created to address the issue of online data retrieval. The map data was also being downloaded using internet, thus the time it took the generate the map plot was also heavily reliant on internet speed. Initially, only downloading the close vicinity where the argo float was located tried. However, this resulted in losing functionality in being able to interact with other sections of the map and did not speed the process significantly.

Most effective solution was downloading the whole world map, storing in the local machine and then zooming to float location more efficient. This still allowed for the map to be fully interactive and internet speed was only an issue at the start when downloading the file. There are multiple qualities of map available, high quality is about 110 MB while the lowest is of 501 KB. Users indicated that for this project, the quality of the map is irrelevant thus the lowest one was deemed as more appropriate. Overall, the new method resulted in about a decrease of 27 seconds compared to previous method to generate both of the plots.

Created interface also has other assisting features such as zooming, panning, saving the plots that can be selected in the lower corner of the application. To note, these tools apply to both graphs thus other sections of the map can be navigated to with the assistance of the panning tool.

3.3 Predictor

This section of the code can be altered by the user to meet their own needs more specifically. To demonstrate the usefulness of the software, prediction of salinity minimum will be demonstrated as it is one of the most well defined features in the oceanography field. The predictive algorithm implemented does not take the equation of state into account meaning the relationship between 3 main variables temperature, salinity and pressure is not taken into account. The method implemented as placeholder predicts values independently of each other. The method implemented is detailed below.

First stage is to normalise the user input, this will help in eliminating the unit of measurement of data and enable the software to compare data from different floats more easily. A method of doing this is to re-scale the data so that values range from 0 to 1. This results in smaller standard deviations which can aid in suppressing possible outliers in data. This can achieved via the equation below:

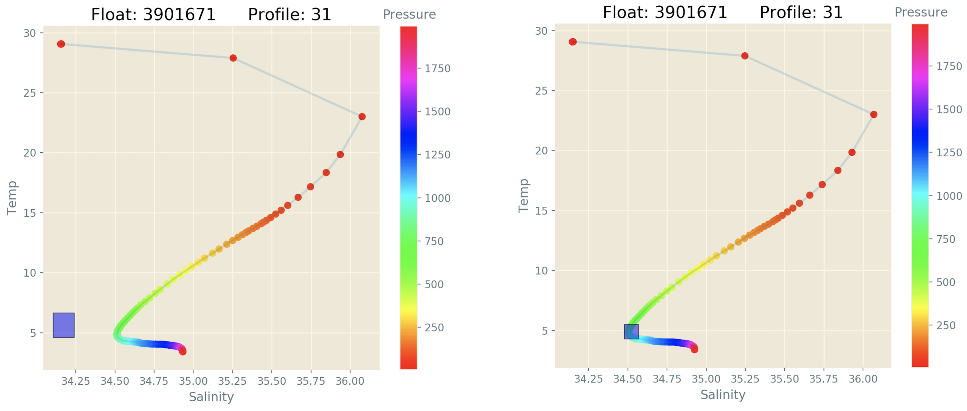
$$Norm = \frac{value - max}{max - min} \tag{1}$$

This algorithm is applied to every user entry, and the mean user entry values are saved. This mean value then can be used with new float data and renormalised to give a prediction where the feature should lie. However before renormalising, possible outliers have to be dealt with so the prediction is not altered negatively. Outlier can be classified as a data points whose values are quite different from other values in the data set that is analysed. However, there is no absolute method to define outliers thus various methods should be tried and the one suited best for the data set should be applied.

One of the tried methods to eliminate the outliers, was to scored the values depending on how much it deviates from the expected model and historical deviation of the variable. A common approach knows as Z score, scoring the outliers with the number of standard deviations of the outlier value. Usually values above 3 are considered as outlier data.(Gustafsson & Sandin 2016).

$$z = \frac{x - \mu}{\sigma} = \frac{x - mean}{StandardDeviation} \tag{2}$$

However, Z-score did not yield satisfactory results. So another method was applied, the interquartile range (IQR) is the range of the middle 50% of the values in a data set, which is calculated as the difference between the 75th (upper quartile Q3) and 25th percentile (lower quartile Q1)values (Boslaugh 2012). This method was only applied to scenarios where the initial prediction did not lie within the original dataset, meaning faulty prediction. The effect of correction made by the IQR can be seen below in Figure 2. The highlighted rectangle in blue illustrates what the algorithm has predicted.



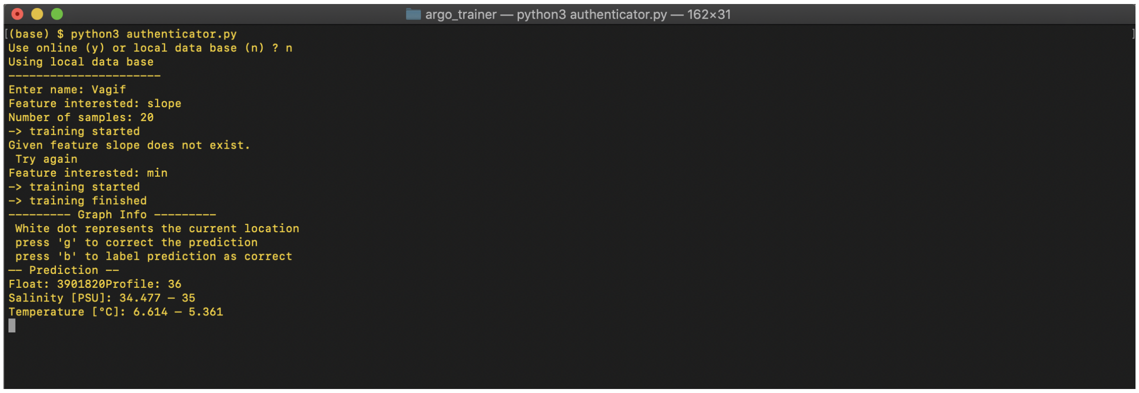
(a) Before

(b) After

Figure 2: Effects of IQR

3.4 Authenticator

Authenticator allows the user to observe the prediction made by the algorithm. On start up, user is asked to enter username, feature interested and the number of samples. In accordance to the feature interested related data entries are collected from the data created using labeller and passed on to the prediction algorithm. In the case where user has requested a feature that has not been defined yet, user will prompted to enter a feature that has been defined previously. The terminal window is displayed below in Figure 3 to demonstrate the process



```
(base) $ python3 authenticator.py
Use online (y) or local data base (n) ? n
Using local data base
----------------------
Enter name: Vagif
Feature interested: slope
Number of samples: 20
-> training started
Given feature slope does not exist.
 Try again
Feature interested: min
-> training started
-> training finished
--------- Graph Info ---------
 White dot represents the current location
 press 'g' to correct the prediction
 press 'b' to label prediction as correct
-- Prediction --
Float: 3901820Profile: 36
Salinity [PSU]: 34.477 – 35
Temperature [°C]: 6.614 – 5.361
```

Figure 3: Terminal

User can approve the prediction as correct by pressing the ”B” key thus increasing the training data set or correct the prediction by making a new selection as previously described. Predictions confirmed as true by the user will be coloured in blue in the map which can be seen in Figure 4. Every new entry made by the user will be added to the training data at the end of the programme.

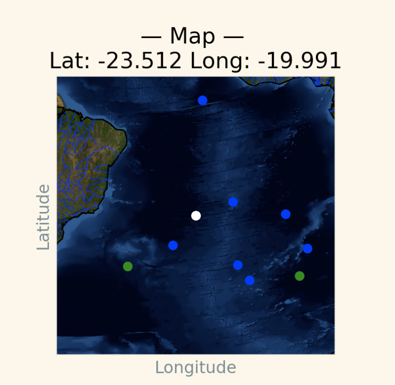


Figure 4: Map Section

Once the user is satisfied with the predictions, the training is concluded. User can now request the prediction to be applied to new data sets and system will output the predictions as a csv file along with png formatted graphs as before.

Machine learning tools are expected to be superior to the predictors used here as a placeholder. Even though, no machine learning algorithm was applied as part of this study. The technique behind the Authenticator where user can simultaneously see the prediction and either confirm/correct it, increasing the data set while checking the quality of predictions at the same time was inspired from machine learning techniques. Dealing with outliers and maintaining physical constraint (i.e equation of state linking parameters) are still in development.

3.5 Code metadata

3.5.1 Platform

Development of this software has been solely done on Python3 programming language. The platform used for development of the software is macOS Catalina Version 10.15.6. The software has also been tested on Windows 10 and successfully executed.

3.5.2 Dependecies

Below the dependencies are included and more information on how to install them is included in the GitHub repository

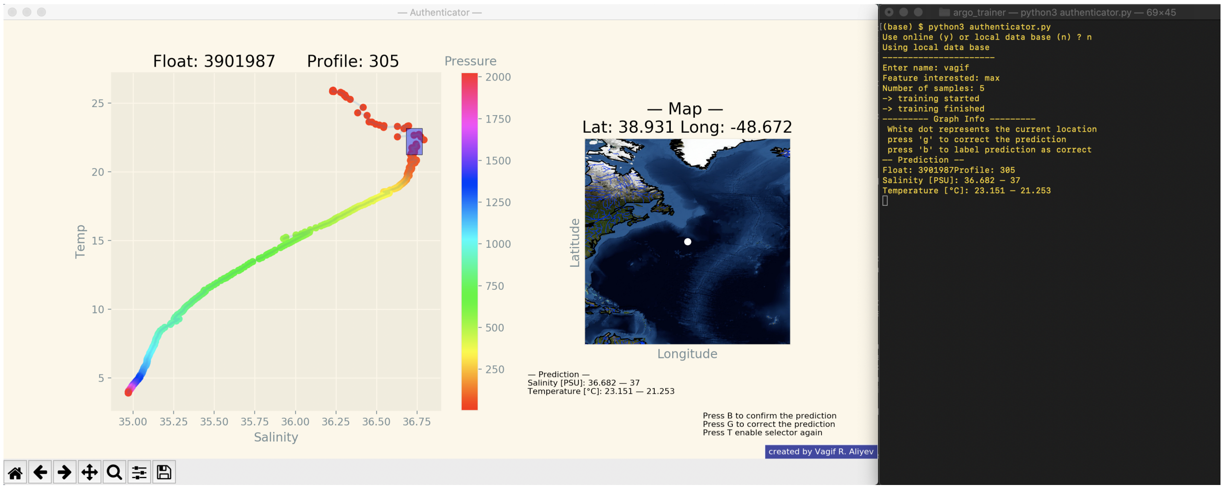
1. Matplotlib (3.1.0) Matplotlib is one of the most commonly used comprehensive open-source library for creating static and interactive visualisations in Python. It is the underlying framework for the visualisation segment of the thesis.
2. Pandas (1.0.5) Pandas is a high-level data manipulation and analysis tool. Through this working with data is more coherent and easier to manage.
3. Argopy (v0.1.5) argopy is a python library that eases the access to Argo data. It is mainly used in retrive.py for automated data retrieval.

Link to the Repositories

— *University Private:* https://github.com/acse-2019/irp-acse-va719
— *Public:* https://github.com/vagifaliyev/argo-trainer

3.6 Illustration

The predictive algorithm has also been tried out with another feature, salinity maximum. However, this feature is not as well defined as minimum thus is more complex to predict correctly as some samples do not even have proper maximums. This proves the algorithm is able to function for different features as well. The system can now be used to start labelling features. With help from the community, an online repository of labelled profiles should be build and community learning exercises can be implemented. Seen below in Figure 5 is an example of prediction of maximum and the platform as a whole.



— Prediction —
Salinity [PSU]: 36.682 — 37
Temperature [°C]: 23.151 — 21.253

Press B to confirm the prediction
Press G to correct the prediction
Press T enable selector again

```
(base) $ python3 authenticator.py
Use online (y) or local data base (n) ? n
Using local data base
----------------------
Enter name: vagif
Feature interested: max
Number of samples: 5
-> training started
-> training finished
--------- Graph Info ---------
 White dot represents the current location
 press 'g' to correct the prediction
 press 'b' to label prediction as correct
-- Prediction --
Float: 3901987Profile: 305
Salinity [PSU]: 36.682 - 37
Temperature [°C]: 23.151 - 21.253
```

Figure 5: Application

4 Discussion and Future Work

One of the biggest constraints in implementing a complex predictive algorithm was the lack of labelled data in the argo field despite the large amount of data available. This project has provided a platform where researchers can easily create labelled files and share with each other. However, due to time constraints satisfactory amount of labelled data was not generated and thus a much more complex predictive algorithm could not be implemented. However, that would be the next natural progression for the continuation in this project as the requirements for creating such a system has been successfully implemented. For that reason, programme has been written in such a way that other users can insert their own predictive algorithm very easily to the already existing infrastructure and observe its performance. As seen in Figure 1, modularity of software allows for any orange highlighted sections to be replaced by the user. Below, more detail on what should be the next progression in this project is discussed.

4.1 Other methods

There are many available data labelling services such as DataTurks. This programme is widely used by large companies and also allows for teams to collaborate easily which will be required in our case to obtain a larger dataset. However, these treat the graphs like images thus information on axes and physical constraints between variables (e.g density is a function of Temperature and Salinity) is lost. There are other available open-source tools such as Trainset (Kapoor. 2020) that allows for labelling of time-series data. However, it requires data to be in a certain format and does not allow multiple parameters to be included in the graph. Hence, a self made data labelling tool for graphical data is considered more appropriate for any future use.

4.2 Community

Creating a large data set of labelled data is a community driven effort. Preserving and sharing acquired knowledge of techniques and data among the community is very essential. The ultimate goal of the project is beyond the scope of a single person team and a short 3 months period of work. Nevertheless, actions have been taken to share the software with as many people as possible.

Contact with Argo Program Office has been made to include this software in their list of Argo software tools, once more researches can access the platform and help in increasing the database of feature selections Machine learning algorithms can be started to be implemented.

The work done has also been made freely available on GitHub for anyone to contribute or learn how to implement a similar platform for different sensor data analysis. Making the software available will allow for more user feedback to be received. This is essential as the developer of this software does not hail from an oceanography background. Getting feedback from more experts will bring in more new ideas that can aide the capability and use fullness of the application.

Once a substantial labelled data has been collected, this could be freely available for ML model experiments on platforms such as Kaggle.

4.3 Predictive Algorithm

The project successfully tackled the problem of not being able to share labelled data among peers in the field. Thus, with contribution from many more user, complex predictive algorithms can be implemented once much more data is labelled. One of the newest trends in the predictive algorithm is machine learning which requires a large amount of data to be able to make predictions. This is not an issue as there is an abundance of data available from argo floats, however these data set is of little use till now as they have not been labelled and processed. Now that there is a platform that allows users to share their findings and research with others through interactive way, better predictive algorithms can be looked into and applied in the future.

Initially, the algorithms will be used to label the incoming data but forecasting is also a possibility. Use of neural networks has been proven previously in oceanography to forecasting sea level and temperatures. Detection algorithms have been explored slightly with examples of oil spill mapping and detection. However, it still remains as an unexplored territory (Ahmad 2019).

There are two main groups of learning available: supervised and unsupervised learning. In this problem, there is 2 known inputs and constant expected 4 outputs. Thus, the learning problem can be classified into the supervised type as the algorithm needs to generates a function that maps inputs into desired outputs (Nasteski 2017).

Decision trees allow for data to be grouped by attributes by sorting them based on their values. It is mainly used for classification purposes (Dey 2016). However, the values are generally preferred to be categorical thus is not suitable for our needs. In the field, it has been used for resource management. It is important to note that the variables do not have a clear linear relationship, hence classic methods such as linear regression model does not apply. However, the equation of state of seawater is known, can be used as constraint and does not have to be 'guesed' by the model (Millero 2010).

The most promising method is Artificial Neural Networks (ANN), it is one of the most fundamental models of machine learning and has been proven to estimate various ocean parameters successfully. ANN algorithms are also computationally efficient and useful tools in modelling nonlinear relationships among given parameters. Allows complex functions such as classification, identification and pattern recognition to be carried out and once performed well with the validation data, the system can be deployed to work in real time (Bhaskaran et al. 2010). The ANN can be trained with respect to known equation of state of seawater. The development of this algorithm will require many experiments as the number of hidden layer, number of neurons and activation function are important in the construction of the model. A grid-search approach can be applied to find the optimised hyperparameters. If overfitting occurs, the Bayesian regularization is recommended to use as it will add smoothness to the cost function to prevent it from occurring (Su et al. 2020). Advancement in this area will open new possibilities in the research of ARGO data.

4.4 Future Goals

The data received from argo floats is continuous with many more new readings coming via satellites daily. With the appropriate amount of data being labelled by experts, more complex algorithms can be implemented that take into account the equation of the state and filter the dataset based on the geographical location of the float to be examined. With more involvement from the community in feedback and data labelling, the final goal would be for the expert who perform quality checks can use the platform to automatically label the continuous stream of new data coming and make their tasks easier to perform so they can focus on solving other issue in the oceanography field.

5 Conclusion

Software created is only part of the solution to the absence of tools in the argo community. Created system allows for people to share their knowledge and findings in a user friendly interactive platform. The goal of this platform was much needed in the community, so that projects and analysis carried out by researches can be shared much easily. The exchange of data will allow new projects to be carried out as discussed in previous section. With the user testing results, features requested have been implemented under the scope of the length of the project. It is now much easier to observe argo data, share findings with others and ease the analysis with the use of a predictive algorithm. This tool will be very useful for the community and can evolve into something even more powerful with collaboration from many.

References

Ahmad, H. (2019), 'Machine learning applications in oceanography', *Aquatic Research* **2**(3), 161–169.

Argo, E. (2020*a*), 'About argo', `http://www.argo.ucsd.edu/About_Argo.html`.

Argo, E. (2020*b*), 'Data visualizations', `https://argo.ucsd.edu/data/data-visualizations/`.

Argo, E. (2020*c*), 'How to use argo profile files', `https://argo.ucsd.edu/data/how-to-use-argo-files/`.

Argo, E. (2020*d*), 'Research use', `https://argo.ucsd.edu/Research_use.html`.

Argo, E. (2020*e*), 'Science highlights', `https://argo.ucsd.edu/science/science-highlights/`.

Bhaskaran, P. K., Kumar, R. R., Barman, R. & Muthalagu, R. (2010), 'A new approach for deriving temperature and salinity fields in the indian ocean using artificial neural networks', *Journal of marine science and technology* **15**(2), 160–175.

Boslaugh, S. (2012), *Statistics in a nutshell: A desktop quick reference*, " O'Reilly Media, Inc.".

Dey, A. (2016), 'Machine learning algorithms: a review', *International Journal of Computer Science and Information Technologies* **7**(3), 1174–1179.

Gustafsson, J. & Sandin, F. (2016), District heating monitoring and control systems, *in* 'Advanced District Heating and Cooling (DHC) Systems', Elsevier, pp. 241–258.

Jones, D. C., Holt, H. J., Meijers, A. J. & Shuckburgh, E. (2019), 'Unsupervised clustering of southern ocean argo float temperature profiles', *Journal of Geophysical Research: Oceans* **124**(1), 390–402.

Kapoor., R. (2020), 'Trainset', `https://github.com/geocene/trainset`.

Le Quéré, C., Andres, R. J., Boden, T., Conway, T., Houghton, R. A., House, J. I., Marland, G., Peters, G. P., Van der Werf, G., Ahlström, A. et al. (2012), 'The global carbon budget 1959–2011', *Earth System Science Data Discussions* **5**(2), 1107–1157.

Maze., G. (2020*a*), 'Argopy', `https://github.com/euroargodev/argopy`.

Maze., G. (2020*b*), 'Mpv:matlabprofilesvisualization', `https://github.com/euroargodev/matlab_profiles_visualization`.

Millero, F. J. (2010), 'History of the equation of state of seawater', *Oceanography* **23**(3), 18–33.

Nasteski, V. (2017), 'An overview of the supervised machine learning methods', *HORIZONS. B* **4**, 51–62.

Roemmich, D. (2012), On the beginnings of argo: ingredients of an ocean observing system, *in* 'Proceedings of the 13th Meeting of the Argo Steering Committee'.

Su, H., Zhang, H., Geng, X., Qin, T., Lu, W. & Yan, X.-H. (2020), 'Open: A new estimation of global ocean heat content for upper 2000 meters from remote sensing data', *Remote Sensing* **12**(14), 2294.