NTNU
Norges teknisk-naturvitenskapelige universitet

Fakultet for informasjonsteknologi, matematikk og elektroteknikk
Institutt for teknisk kybernetikk

MASTEROPPGAVE

Kandidatens navn: Even Løberg Wanvik

Fag: Teknisk Kybernetikk

Oppgavens tittel (norsk): Maskinlæring for identifikasjon av egenskaper i havmodelldata og fjernmålinger

Oppgavens tittel (engelsk): Machine learning for identification of features in ocean model and remote sensing data

Oppgavens tekst:
Prediksjon av havdynamikk ved hjelp av modeller er utfordrende på grunn av systemets ulineære dynamikk og relativt få tilgjengelige målinger. For å kunne gjøre mest mulig effektive modelloppdateringer basert på målinger er det interessant å etablere filtre som kan "tolke" modelldata og fjernmålinger fra satellitter for å finne egenskaper som plassering og bevegelse av virvler. Denne oppgaven går ut på å utvikle og teste maskinlæringsmetoder på datasett fra havmodellen SINMOD og satellittbaserte målinger med spesielt fokus på virvler og evt. fronter.

Oppgaven oppsummeres i følgende punkter:

- Litteraturstudie for å få oversikt over forskningsfronten på dette området, og for å identifisere relevante maskinlæringsmetoder.
- Etablere datasett bestående av modelldata (SINMOD og/eller data fra CMEMS) og observasjonsdata (fra CMEMS eller andre kilder).
- Annotere data for opptrening og test av algoritmer. Dette kan gjøres via en rekke tilgjengelige verktøy, noen med mulighet for delautomatisering av annoteringsprosessen.
- Teste en eller flere egendefinerte eller etablerte maskinlæringsalgoritmer.

Oppgaven gitt: 6. januar 2020

Besvarelsen leveres innen: 1. juni 2020

Utført ved Institutt for teknisk kybernetikk

Veileder: Morten Omholt Alver

Trondheim, 6. januar 2020

Morten Omholt Alver

Fagærer

Preface

This thesis is a culmination of my study at a 2-year master's programme in Cybernetics and Robotics at NTNU, Trondheim. After finishing my years working at CERN and writing my bachelor's, I had a full year hiatus in all control theory and complex mathematical equations, making the initial semester challenging.

I want to thank my classmates for making the experience enriching and enjoyable experience (especially with whom I have shared office). I also want to thank all my close friends and my family for being a strong support network during my long years of studying. And of course, lastly, I would sincerely like to thank my supervisor Morten O. Alver, who allowed me to complete this master's project and was a great help with exceptional enthusiasm and expertise.

Abstract

Predicting ocean dynamics using numerical models is challenging because of the unpredictable non-linearities and limited access to physical observations. Ocean eddies pose a complicated conceptual and practical challenges to theory and the models. Knowing the position and scale of mesoscale eddies can be used as a part of the observations in the assimilation process, improving the model's certainty and exactness. Due to the advantages such information, it was of interest to research novel machine learning methods to interpret data produced by models (e.g., SINMOD) or observed through remote measurements to recognize ocean features such as ocean eddies.

Because there is no "one fits all" machine learning (ML) algorithm, three of the most common supervised learning algorithms were evaluated: support vector machines (SVM), random forest, and Convolutional Neural Networks (CNN). Because the models need a sufficient amount of training samples, a data annotation application was created to generate a sufficient amount of training samples containing eddy features. The final training set included 2045 samples containing sea surface height, temperature, and ocean currents. After a smaller investigation of their ocean eddy predictability, an ensemble consisting of the ocean current vectors were found to provide much better performance without both sea surface height and temperature. In an initial trial of finding the best ML algorithm, CNN were found to be the best performing. The second trial examined three modified versions of the original CNN architectures: VGG, ResNet, and Inception. The best performing model was found to be a simplified modification of a VGG network structure. The final model could be used to detect multiple sliding windows on a selected grid of sea surface currents. The final predictions are merged using grouping techniques, which is further refined using well-established flow-field equations such as the Okubo-Weiss parameter and vorticity to provide more precise boundaries encapsulating the predictions.

The CNN model performed well when tested on a hold-out set of the training data, reaching accuracies above 96%. The aggregated system of prediction and post-processing provided satisfactory results when tested on both SINMOD and other models and observational datasets. After analyzing the system's performance across the datasets, there were negligible similarities between the assimilated and observed ocean dynamics, although a year-long comparison seemed to find seemingly comparable trends in eddy activity due to bathymetry and the season.

Sammendrag

Å forutsi havdynamikk ved bruk av numeriske modeller er utfordrende på grunn av uforutsigbare ikke-lineariteter og begrenset tilgang til fysiske observasjoner. Havvirvler utgjør konseptuelle og praktiske utfordringer for teori og modeller. Å vite posisjonen og dimensjonene til mesoskala-virvler kan brukes som en del av observasjonene i assimilasjonsprosessen til modellen, noe som forbedrer dens robusthet og nøyaktighet. På grunn av fordelene av å vite karakteristikker som virvelens dimensjon og posisjon, var det av interesse å forske på nye maskinlæringsmetoder for å tolke data produsert av modeller (f.eks. SINMOD) eller observert gjennom fjernmålinger, for å gjenkjenne havfunksjoner som havverdier.

Fordi det ikke eksisterer en "passer-til-alt" maskinlæringsalgoritme, ble tre av de vanligste veiledet læringsalgoritmene evaluert: Støttevektormaskiner (SVM), beslutningstrær og Konvolusjonelt nevralt nettverk (CNN). På grunn av at modellene trenger en tilstrekkelig mengde treningsdata, ble det opprettet en annoteringsverktøy for å produsere en tilstrekkelig mengde data som inneholder gjenkjennbare trekk som gjenkjenner en virvel. Det endelige treningssettet inkluderte 2045 treningseksempler som inneholder variablene havoverflatehøyde, -temperatur og -havstrømmer. Etter en mindre undersøkelse av hvilke variabler som har sterkest sammenheng med utfallene av interesse, ble det funnet ut at havstrømsvektorene ga mye bedre ytelse, med eller uten bruk av havoverflatehøyde og temperatur. I en første test av maskinlæringsalgoritmer ble CNN funnet å være den som presterte best. En videre test av CNN arkitekturer undersøkte tre modifiserte versjoner av de originale CNN-arkitekturene: VGG, ResNet og Inception. Modellen som presterte best, ble funnet å være en enkel modifisering av en VGG-nettverksstruktur. Den endelige modellen kan brukes til å detektere et ensemble av såkalte glidende-vinduer ("sliding windows" på engelsk) på et større utvalgt område av havoverflatestrømmer. De endelige predikterte vinduene blir deretter slått sammen ved hjelp av grupperingsteknikker, som deretter etter-prosesseres ved hjelp av veletablerte flyt-felt-ligninger som Okubo-Weiss-parameteren (OW) og vortisitet for å gi mer presise grenser som innkapsler prediksjonene.

CNN-modellen fungerte bra da den ble testet på et sett som ikke ble brukt for trening, og nådde nøyaktighetsverdier over 96 %. Det aggregerte systemet med prediksjon og etterprosessering ga tilfredsstillende resultater når de ble testet på både SINMOD og andre modell- og observasjonsdatasett. Etter å ha analysert systemets ytelse på tvers av datasettene, var det ubetydelige likheter mellom den assimilerte og observerte havdynamikken, selv om en lengre 15-måneders sammenligning så ut til å vise tilsynelatende sammenlignbare trender i virvelaktivitet på grunn av batymetri og sesongen.

Table of Contents

Abbreviations

ML	=	Machine Learning
AI	=	Artificial Intelligence
SVM	=	Support Vector Machine
MPL	=	Multilayered Perceptron
CNN	=	Convolutional Neural Network
RNN	=	Recurrent Neural Network
LSTM	=	Long Short-Term Memory
CMEMS	=	Copernicus Marine Environment Monitoring Service
RBF	=	Radial Basis Function
SLA	=	Sea Level Anomalies
SSH	=	Sea Surface Height
OW	=	Okubo-Weiss
NetCDF	=	Network Common Data Form

Chapter 1

Introduction

1.1 The Datasets

It is within the interest of this project to investigate the possibility of constructing a system that could provide locations of ocean features such as eddies. In the case of detection by recognition systems such as ML, one needs training samples. The training and testing samples are collected from both the SINMOD and the Copernicus Marine Environment Monitoring Service (CMEMS).

1.1.1 SINMOD

SINMOD is a numerical model system that has been developed at SINTEF since the early 80s. The system's primary model is a 3d hydrodynamic model, which is based on the Navier-Stokes equations that solve the finite difference method on a regular Arakawa C-grid. The model employs atmospheric forces, freshwater runoff, and the tides as inputs, and the current and density as border conditions. A more detailed review of the hydrodynamic model can be found in "Slagstad and McClimans(2005)" [1].

1.1.2 CMEMS

CMEMS is one of the multiple services provided by the Copernicus program[2]. The CMEMS provides regular and systematic reference information on the physical state, variability, and dynamics of the ocean, ice, and marine ecosystems for the global ocean and the European regional seas. Two sources of global ocean variables are were downloaded and used for this project: CMEMS-phys [3] and CMEMS-multiobs [4]. The CMEMS-phys (physics analysis) dataset contains data produced by data assimilation, while the CMEMS-multiobs (multi observations) use satellite-based remote measurements and in-situ observations.

1.2 State-of-the-art

The contribution to automated oceanic eddy tracking algorithms mostly comprises of three main methods of detection: The first method is based on wavelet packet decomposition of the sea level anomalies (SLA) data; the second on the Okubo-Weiss (OW) parameter; and the third on a geometric criterion using the winding-angle approach (Souza et al. [5] have made a comparative analysis of these approaches). While examining the state-of-the-art eddy detection, most of the methods seemed to employ OW criteria combined with the sea surface height (SSH) to trace the core of the eddy. The OW parameter was proposed by Okubo [6] and Weiss [7] to provide a measure of the relative rotation and deformation of particles passing through a flow field. The properties of their equations play a central role in a proposed refinement procedure for this project and will be thoroughly explained in the theory section. In more recent years, there has been an increase in papers using more novel artificial intelligence (AI) to detect eddies. This study will shortly discuss the conventional detection and tracking methods before exploring the state of the more contemporary methods employing machine learning techniques.

1.2.1 Conventional methods of detection

Over the past decades, numerous studies have tackled the problem of automated eddy detection from both observational and assimilated model data for both global and regional data. Most methods until recently have, for the most part, employed well-established special-purpose equations and models to detect the presence of oceanic features (a few examples: [8, 9, 10, 11, 12, 13, 14, 15, 16, 17]). While the research for more novel use and modifications has been very active, only a few studies make public their eddy identification software or trajectories (a few examples: [18, 19, 20, 21, 22, 23]).

Isern-Fontanet [19, 20, 21] created a framework that revealed strong matching between the eddy features of the SSHA and OW fields. While citeRN84 developed a vorticity based heuristic Euler-Lagrangian descriptor employing an idea of Coherent Lagrangian Structures that separate flow into regions of different dynamical behaviors, showing that the Lagrangian descriptor gave a more robust detection and tracking than the OW method.

[18] performed a comprehensive three-dimensional ocean eddy census over the duration of 7 years employing the OW fields in combination with a R_2 goodness of fit measure for finding the location and sizes. The method had multiple criteria as input parameters depending on the confidence level of its predictions, i.e., stricter criteria would output the larger and more likely eddies. The project contains open source software available for testing on NetCDF variables, with a user-friendly display of eddy properties.

1.2.2 Detection using machine learning

The more recent years have seen a surge in the implementation and modification of machine learning techniques to aid in identifying and tracking ocean features such as eddies.

[24] investigated using the phase angle between the northward and eastward ocean current velocity to train and test a support vector machine (SVM) to detect eddy features. The phase angle (direction angle) will correspond to a clockwise or counterclockwise pattern because of the ocean current's circulatory motion in an eddy. The core of a vortex

will be characterized by having a full 360-degree sequence of phase values. They compared the model with existing eddy detection models, which provided satisfactory results even though they had limited data.

[25] looked at satellite images as textural patterns of water temperatures and aimed at revealing the structural signature characterizing a meddy (Mediterranean eddy). The proposed pre-processing method extracts a numerical vector containing information on the surrounding region's thermal gradient field. A fully connected multilayered perceptron (MPL) ANN composed of three or four layers of processing elements, was then used to detect the structural signature patterns corresponding to a meddy.

[26] proposed an eddy detection and tracking framework, combining feature learning by CNNs with an established image processing tool as a feature tracker. It compares the Okubo-Weiss (OW) tracking with recurrent neural network (RNN), long short-term memory (LSTM), trained for eddy identification. They utilize the OW method to tackle the lack of annotated data by detecting a few yet precise eddies used as training data. The absence of training data is a familiar problem for all pattern recognition systems studied analyzed during this study.

1.3 Motivation

1.3.1 Importance of mesoscale eddies

Laminar flow is known for its smooth and deterministic behavior, whereas turbulent flow is perceived as the ugly duckling of flow, random and chaotic. Turbulent flow does not even have a formal and universal definition. Although Navier Stokes equations are meant to govern all fluid flow, including turbulence, they are notoriously hard to solve. A million-dollar price [27] is being awarded for anyone that can contribute any progress towards getting insight into these equations. One of the defining characteristics of turbulent flow is that it consists of many interacting swirls of fluid, also known as eddies or vortices. Eddies are circulatory currents of fluids on Earth, which span sizes from a few meters or less in diameter (microscale) to synoptic scales on the order of 1000 kilometers. Image 1.1 shows surface flows and gyres in the Northern Hemisphere.

Mesoscale eddies have an immense influence on ocean dynamics and the distribution of biomass, such as phytoplankton. Ocean currents, gyres, and eddies play a critical role in shaping the ocean dynamics and the distribution of biomass – sustaining countless plants and animals – including humans [16]. Figure 1.2 shows how eddies can stir the ocean and pull nutrients up from the deep, fertilize the surface waters to create blooms of phytoplankton in the usually barren open ocean. Eddies also play an important role in shipping routes and have been known to cause harm to larger offshore structures such as oil platforms.

In the past few decades, satellite altimetry has revealed the ubiquity of mesoscale eddies in both the global ocean [29, 30]. In recent years, automated oceanic eddy tracking algorithms have emerged because of their role in the dynamics of the large-scale oceanic circulation and impact on the transport of biological and physical material.

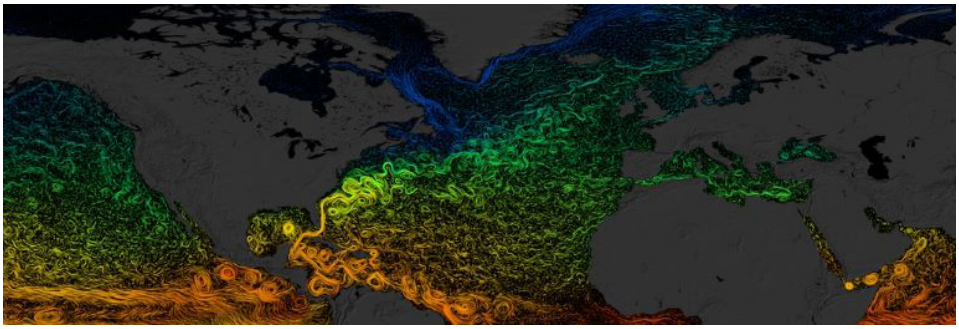


Figure 1.1: The image is credited to NASA's Goddard Space Flight [28]: "The image is a combination between NASA satellite data with field measurements to present a model view of surface flows and gyres in the Northern Hemisphere from March 2007 to march 2008. Observe the dramatic difference in strength between westward and eastward currents as they hook clockwise in the Atlantic and Pacific oceans. And notice how westward currents explode into spiraling, turbulent flows off the eastern coasts of Asia and North America."

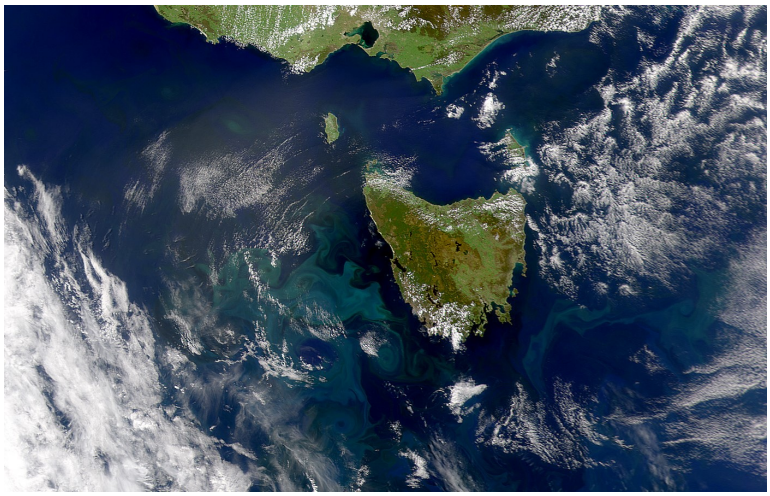


Figure 1.2: A satellite image that shows large concentrations of phytoplankton bloom over hundreds of square kilometers in size, off the west coast of Tasmania in the Indian Ocean. High concentrations of phytoplankton create a lighter turquoise appearance, highlighting the ocean current's patterns, such as the eddies, or vortices, in the water. Image courtesy of NASA's SeaWiFS Project [28]

1.3.2 Assimilated models

The application of ocean and biomass state predictions in systems related to farming, marine agriculture, and contamination control are numerous. Ocean models are widely used in determining states such as the salinity, temperature, current, sea ice, and ecosystem. The reliability of the estimated states relies on the variability and uncertainty in the model's calculations, parameters, initial values, driving forces, and boundary conditions. The

process of optimally combining the theory, usually in the form of a numerical model, with observations is known as data assimilation.

There are two classes of data assimilation methods: variational and sequential. The variational methods rely on back-calculating initial and boundary values and driving forces such that the assimilated variables match the data. Both are similar in that they are sequential; however, sequential methods employ model estimation from advanced filtering techniques combined with statistically-based correction terms.

Eddies pose a complicated conceptual and practical challenges to theory and model. However important they are for the general circulation, the limits of computer resources make course models produce eddies with high variability. By knowing the position and scale of mesoscale eddies and using it as a part of the data assimilation process, one could conceivably improve a model's certainty and exactness. The problem posed for this project was to research the possibilities of using machine learning techniques to detect eddies that could provide the model with observations such as position and size. The long-term concept, beyond the scope of this project, is to create a system that works as an eddy-census feedback loop that could provide daily updates on eddies and their attributes.

1.4 Research aims and objectives

As mentioned in the previous section, the prediction of ocean dynamics with the help of numerical models is challenging because of the unpredictable non-linearities and the limited access to physical observations. It is of interest to build filters, such as AI models, to interpret data produced by the model or observed through remote measurement systems to locate and track the movement of features such as ocean eddies. The recognition and interpretation of ocean features will help adjust the model and improve the model's predictions.

The project's aims and objectives can be summed up by the following principal points:

- Litterature study to gain comprehension of the state of the art ocean feature detection system and machine learning algorithms.
- Establish datasets containing data from both ocean models (SINMOD or CMEMS, or both) and real-world observations (from CMEMS or other sources).
- Annotate training data for training and testing machine learning algorithms, either by using available annotation tools (if possible) or by creating a special-purpose tool (manual or partially automated).
- Train and test one or more custom, or well-founded machine learning algorithms and methods.
- Compare the performance of the final machine learning model on the SINMOD model data and observational data.

1.5 Outline of the report

The project report is divided into four chapters, excluding the introduction, followed by references: Theory, Methods, Results & Discussion, and Conclusion.

The second chapter (post introduction) will introduce the theory followed by the relevant citations needed to comprehend the objectives and methodology of the project. First, the section explains the ocean dynamics and physics related to ocean eddies and what features ocean variables such as SSH, SST, and ocean current exhibit in the presence of an eddy. Secondly, the chapter describes the ML techniques used in this project and other aspects of the ML pipeline.

The third chapter firstly demonstrates how the necessary training data have been collected. Next, it investigates the performance of relevant ML techniques on the collected training data. After finding the best performing model, the report proposes a post-processing procedure for refining the predicted cyclones and anti-cyclones.

Chapter four will both analyze and discuss the results of the procedures described above. It will contain a reiteration as to why the ML model and training data was the final choice and discuss their validity. The proposed detection system's performance across the three mentioned datasets will be tested and cross-correlated.

The closing chapter will then conclude the report by addressing the principal features and details of the report in a brief summary.

Chapter 2

Theory

2.1 Ocean dynamics

The geostrophic ocean currents created by the interaction between baroclinic instabilities and the Coriolis effect caused by the rotation of the earth create circulatory ocean currents and oddities in both sea surface height (SSH) and sea surface temperature (SST). The local spinning motion of moving fluid parcels can be evaluated by deriving the curl of the vector field, also known as its relative vorticity. Okubo [6] and Weiss [7] applied both the relative vorticity and other kinematic properties to evaluate if a parcel is dominated by vorticity or normal or shear strain.

2.1.1 Measuring oceanographic variables

The sea surface is exceptionally complex, full of smooth hills and valleys with vertical amplitudes ranging anywhere between one to hundreds of meters. Ocean remote sensing has made tremendous progress in the last decades to help study sea surface dynamics and other meteorological conditions. With the advent of satellite measurement systems, researchers can obtain global measurements of key variables such as sea level anomalies, temperatures, salinity, and ocean currents.

The satellite determines the sea surface from a round trip of emitting and receiving microwave pulses reflected from the ocean. By determining its three-dimensional position relative to a fixed Earth coordinate system, the measurement system yields the sea surface level, or topography, profiles. The geostrophic surface currents originate from horizontal pressure gradients and manifest as sea surface height slopes relative to the marine geoid (global mean sea level). The most substantial fluctuations in energy have typical horizontal scales of 100-300 km and are attributable to the mesoscale ocean eddy field (the meandering of narrow currents and migration of detached vortices). Together with the geoid and the ocean's density field, the altimetry provides a method for determining the ocean's geostrophic currents. The sea surface temperature and salinity are measured using infrared satellite sensors picking up the thermal radiation from the sea surface.

2.1.2 Spatial and temporal scales

A glass of water, swimming pool, or the ocean obeys the same laws of fluid dynamics. However, the ocean has increased complexity and its own set of properties due to the scales involved. In general, the ocean has much larger horizontal scales compared to vertical. Vertically there are mixing that depends on the density profile of the water, and external perturbations caused by wind-driven currents can affect waters as deep as 100-200m below the surface. On the horizontal scale, there are eddies ranging from 10 to 100 km, and gyres on the scale of ocean basins circulating between the coast of continents [31]. The Coriolis force has a significant effect on the horizontal scale, as it deflects ocean currents at right angles to the velocity vector of the moving fluid, caused by the rotation of the Earth. The Coriolis effect and other external forces affecting the fluid acceleration will be explained in a subsequent section.

The seawater's properties are not uniform, and varying densities have a profound effect on ocean dynamics. We say that the water column is stratified if there are clear density, temperature, and salinity (salt content) gradients between layers within a parcel or area. The vertical layers are more stable with less transport between the water column if the density increases with depth (i.e., lighter water is above denser water), and more prone to vertical mixing if the density decreases with depth. Horizontally, the pressure gradients are much smaller than the vertical changes but lead to pressure-driven ocean currents.

In addition to the spatial scale, the ocean dynamics act on a wide range of temporal and velocity scales. The high-end edge case on the temporal scale is the thermocline circulation shown in figure 2.1. The word thermohaline is derived from the factors thermo, referring to temperature, and haline, referring to salinity, which together determines the density of water. The circulation of this global conveyor belt waters can last as long as thousands of years. In addition to the thermohaline circulatory currents, there are large ocean gyres formed by wind patterns that rotate in decade-long cycles. The five major gyres circulate the Indian, North Atlantic, South Atlantic, North Pacific, and South Pacific basins. At the lower end of the temporal scale are internal and external waves with varying velocities.

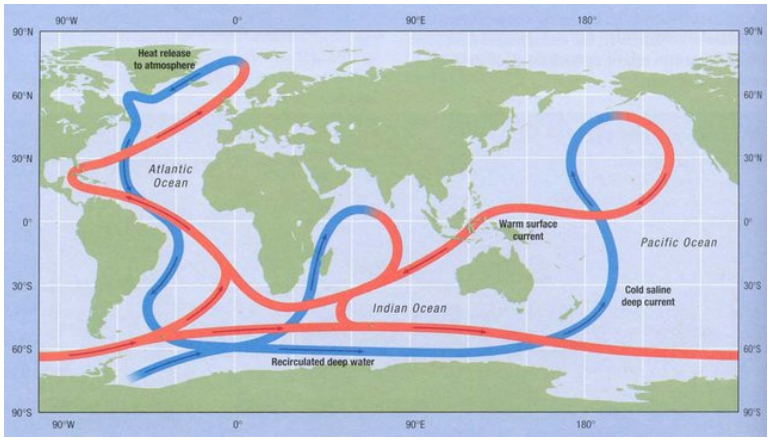


Figure 2.1: Image taken from [**?**]

Rossby waves are larger scale, smaller amplitude waves that gain momentum from the wind shear on the surface in combination with the vorticity caused by the Earth's rotation. External Rossby waves propagate at the lower end of the velocity scale at around 20 m/s.

2.1.3 Equations of motion

The ocean's eqiation of motion are derived from Newton's first law, in which the acceleration experienced by a parcel of fluid equals all forces acting upon that parcel, i.e.,

$$\Sigma \mathbf{F} = m\mathbf{a} \tag{2.1}$$

in which the force **F** and acceleration **a** are vectors with x, y, and z components. A fluid's acceleration from an inertial frame of reference is mainly determined by **Pressure**, **frictional**, and **gravitational** gradients.

- **Pressure gradients** arise from differences in sea level (*barotropic*) and differences in density (*baroclinic*).
- **Frictional gradients** is the *wind stress* caused by the friction caused in the interface between wind velocity and the sea surface, the deceleration caused by *bottom stress*, and the internal friction (*mixing*) among density layers of the water column.
- **Gravitational gradients** is caused by the rise in ocean levels (tides) due to the attraction between celestial bodies.

If we divide Newton's law in equation 2.1 by the mass, one can write the acceleration for a parcel of water as:

$$\Sigma acceleration = \frac{pressure + gravitation + friction}{mass} \tag{2.2}$$

After filling in the terms in the above equation, the Navier-Stokes equation assuming incompressible fluid can be derived:

$$\frac{\delta \mathbf{u}}{\delta t} + (\mathbf{u} \cdot \Delta)\mathbf{u} - \nu \Delta^2 \mathbf{u} = -\frac{1}{\rho}\Delta p + g, \tag{2.3}$$

in which **u** is the flow velocity vector, Δ is a vector differential operator (del), ν is the kinematic viscosity, p is the pressure, ρ is the density, and g is the acceleration due to gravity. The first term on the l.h.s. of the equation considers the local acceleration; the second term is due to advection of the flow field, which appears in a Eulerian flow field; the final term on the l.h.s. represents the diffusion of momentum caused by the viscosity. On the r.h.s., the first term represents the acceleration due to the pressure gradient.

Equation 2.3 assumes an inertial reference frame fixed in space, for which the velocity of a body of fluid equals the velocity of the particle plus the angular velocity of the Earth. Usually, however, it is desired that the coordinate system is attached to a specific location on Earth. By transforming the frame of reference to the Earth's surface, introduces an additional term representing the Coriolis effect. The Coriolis acceleration is perpendicular to the horizontal velocity vector of the parcel, whose direction depends on the hemisphere

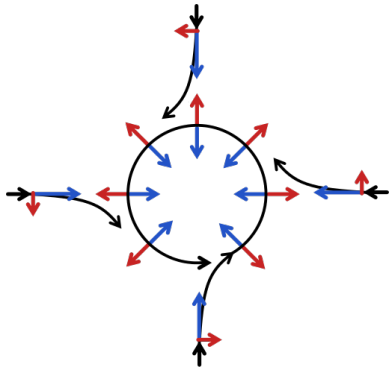


Figure 2.2: Representation of flow around a low-pressure cyclone in the norther hemisphere. The pressure gradient is represented by the blue arrows, and the Coriolis effect (always perpendicular to the velocity) by the red arrows.

and its magnitude depending on the latitude and velocity. A more detailed explanation of the Coriolis effect, and how it is one of the main contributors to the creation of mesoscale ocean eddies, will be given in the following section.

The Navier-Stokes momentum equation in equation 2.3 is accompanied by the continuity equation for an incompressible fluid, representing the conservation of mass, given by the equation

$$\Delta \cdot \mathbf{u} = \frac{\delta u}{\delta x} + \frac{\delta v}{\delta y} + \frac{\delta w}{\delta z} = 0 \tag{2.4}$$

All materials exhibit some variation in volume under compression. The degree of compressibility of water is so low that it can be assumed to be incompressible in many cases. On the scale of ocean models, this is a valid assumption, and it has been applied in equations 2.3 and 2.4. The conservation of momentum is essential with regards to the properties of ocean eddies and other phenomena, which will be further discussed in the subsequent introduction to ocean eddies.

2.1.4 Ocean eddy

The ocean dynamic is much more complicated than what is shown on maps and globes showing the ocean gyres or the thermohaline circulation like the one shown in figure 2.1. If one were to place buoys throughout the ocean at random locations, a few of them would move in a circulatory motion like a loop. These loops are known as eddies (eddy singular) and are common mesoscale patterns of the oceanic flow. Mesoscale is an intermediate scale smaller than the synoptic scale (ocean gyres and large weather fronts that areas above 1000 km) and larger than the microscale (about 1 km or less). Horizontal mesoscale phenomena generally range from about 5 km to several hundred kilometers. Mesoscale eddies range from about 10 to 500 km in diameter and are known to persist for days up to months [32].

Mesoscale eddies can either be static eddies caused by obstacles or the bathymetry (depth) profile, or transient eddies caused by the baroclinic instabilities. The baroclinic instabilities cause irregularities in mean flow, increasing turbulence, and the generation

of eddies [33]. To paraphrase: increased velocity cause a higher Reynolds number and increased turbulence, which promotes the formation of eddies. [34] shows that the energy flows are much more vigorous in winter than in summer, causing more activity in existing eddies and the formation of eddies.

The rotational properties of an eddy are caused by the interaction between the mentioned baroclinic disturbances and the Coriolis effect. The previous section introduced the notion that the Coriolis effect is an inertial force that acts upon objects moving in a moving frame of reference [35]. Figure 2.3 shows how winds and ocean currents moving across the Earth moves in a deflected path as a result of the Coriolis effect. The deflection is different between the direction of the wind and in what hemisphere it moves. Figure 2.2 illustrates how a body of water in the northern hemisphere with higher pressure moves towards an area of lower pressure trying to attain an equilibrium, as stated by the second law of thermodynamics [36]. The current moving north towards an observer placed at the center of the eddy in figure 2.2 will move faster because of the difference in Earth's rotation the further south in the northern hemisphere. This difference in rotational velocity is what causes the Coriolis effect, and is why the northward current in figure 2.2 deflect to the east and southward current to the west. Due to the Coriolis effect, cyclonic rotation is counterclockwise, and anticyclonic rotation is clockwise in the northern hemisphere (vice versa for the southern hemisphere).

The conservation of momentum equation and the baroclinic instabilities caused by

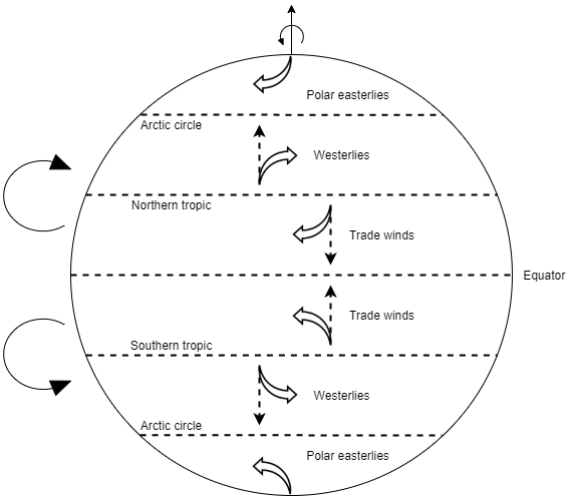


Figure 2.3: The image shows the impact of the Coriolis effect on winds. The Earth rotates towards the west with a counterclockwise spin when observed from the north pole. The Earth's spin causes the movement of the wind and ocean currents to appear as a curve instead of a straight line. For instance, the trade winds in the northern hemisphere blow originally towards the equator but are deflected by the Coriolis force. The original direction of the winds is shown as dashed lines, while the thick arrow demonstrates the deflection. For instance, the ocean gyres are mostly caused by the Earth's rotation; e.g., the gyres in the northern hemisphere spin clockwise, while the gyres in the southern hemisphere spin counterclockwise.

an ocean eddy suggests that there will be a smooth surge in sea surface height in areas of increased pressure and decrease in regions of lower pressure. Figure 2.2 illustrates how the interaction between the pressure and Coriolis gradients creates the geostrophic counterclockwise currents to a cyclone in the northern hemisphere. The decreased pressure around the core of the cyclone suggests decreased sea surface height and vice versa for the anticyclone.

Another noteworthy effect of the ocean eddy is its impact on ocean variables such as temperature and salinity. As with the phytoplankton in figure 1.2, eddies' advective effects between the wind speed of the sea surface and the sea surface temperature, causes cold and warm spirals of temperature. The converging cyclones in the northern hemisphere with decreased sea surface height contains spiraling rings of colder sea surface temperature, while the anticyclone has a warmer ring signature.

2.1.5 Vorticity

Vorticity is a measure of the instantaneous local spinning motion of a fluid parcel, as seen by a frame moving along with the flow. Vorticity in fluid dynamics is analogous to the angular velocity in solid body mechanics. The vorticity is a vector field ω defined as the curl of the flow velocity field Δv given by the following formula:

$$\underbrace{\Delta \times \vec{v}}_{\text{Curl}} = \left(\frac{\delta w}{\delta y} - \frac{\delta v}{\delta z}\right)\hat{i} + \left(\frac{\delta u}{\delta z} - \frac{\delta w}{\delta x}\right)\hat{j} + \left(\frac{\delta v}{\delta x} - \frac{\delta u}{\delta y}\right)\hat{k}, \tag{2.5}$$

which is a three dimensional vector function with eastward ($u(x, y, z)$), northward ($v(x, y, z)$), and vertical ($w(x, y, z)$) velocity components.

The absolute vorticity in the ocean is composed of both planetary vorticity (spin caused by the rotation of the Earth) and relative vorticity (local spin relative to Earth) [37]. If the observer were to look down onto the North Pole, the Earth would be rotating counterclockwise; however, if one were to look up towards the South Pole, Earth would rotate clockwise. Therefore, the planetary vorticity is always counterclockwise (positive) in the northern hemisphere and clockwise (negative) in the southern hemisphere. The planetary vorticity's contribution is at its maximum at the north pole (the full extent of the Earth's spin); however, at the equator, the contribution is zero because there is no added rotation to the horizontal velocities. In the context of large-scale oceanography, only the rotation of the horizontal flow field is used, i.e., the final term on the r.h.s. of equation 2.5, because the fluid layers are insignificant compared to Earth's radius [38]. By removing the vertical components (the two first terms) from equation 2.5, we see that the vorticity of the horizontal flow is always perpendicular to the two-dimensional flow creating a scalar value given as the angular velocity. The formula for absolute vorticity for the horizontal plane is given as a combination of both relative and planetary vorticity:

$$\eta = \zeta + f = (\frac{\delta v}{\delta x} - \frac{\delta u}{\delta y}) + 2\Omega sin\phi \tag{2.6}$$

The first of the first term of the final expression is the relative vorticity (same as the last term in equation 2.5), and the second term is the planetary vorticity. The vorticity is generally an important property to identify circulatory phenomena in the fluid flows of both the ocean

and atmosphere. The relative vorticity from equation 2.6 would take form as an rotation of the two-dimensional parcel shown in figure 2.4 which only illustrate the normal and shear strain of a parcel moving along a flow field.

2.1.6 The Okubo-Weiss parameter

The Okubo-Weiss parameter is a measure of the relative rotation and deformation of particles passing through a flow field. Both Okubo [6] and Weiss [7] analyzed how adjacent particles disperse and behave when passing through a complex flow field. Okubo analyzed singularities in oceanography, i.e., points of convergence (or divergence), and how they affect surrounding particles. Weiss found that the fluid in areas where the vorticity gradient exceeded the strain was in an elliptical mode of motion that advects the vorticity smoothly. In other words, a parcel moving through an area of an elliptical mode of motion will receive more vorticity than strain. They both came up with the same set of topological properties of particle trajectories making up the equation

$$W = S_n^2 + S_s^2 - \omega^2, \tag{2.7}$$

in which S_n and S_s are the normal and the shear components of strain, and ω the relative vorticity. Nevertheless, each of the kinematic properties given in equation 2.8 can

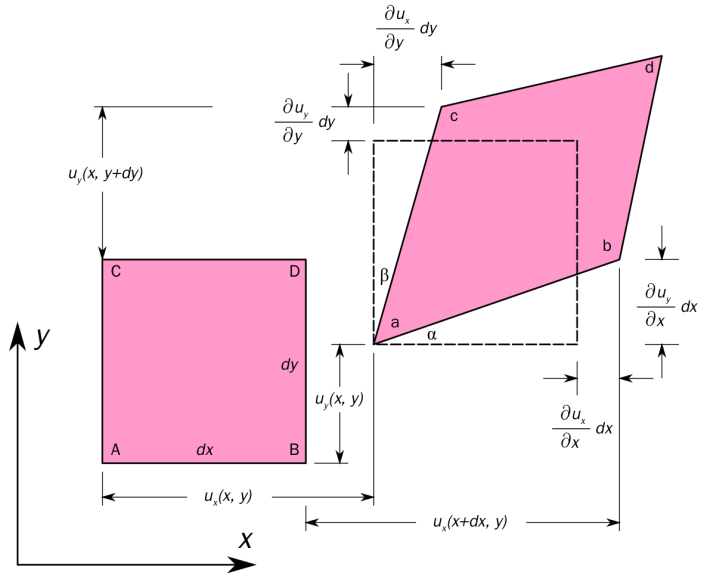


Figure 2.4: Two-dimensional geometric deformation of an infinitesimal parcel of water. The parcel is affected by both normal and shear strain while moving along the flow field (here given as u_x and u_y). There is no relative vorticity component active, as any rotation of the parcel is nonexistent.

respectively be expressed as

$$S_n = \frac{\delta u}{\delta x} - \frac{\delta v}{\delta y}, \quad S_s = \frac{\delta v}{\delta x} + \frac{\delta u}{\delta y}, \quad \omega^2 = \frac{\delta v}{\delta x} - \frac{\delta u}{\delta y}. \tag{2.8}$$

The variable W is named the Okubo-Weiss (OW) parameter after both Okubo and Weiss, and allows for seperation into three different types of flow field:

- $\boldsymbol{W > 0}$: Strain dominated area (particles or parcels with elevated OW values).
- $\boldsymbol{W = 0}$: Small positive and negative values denotes a background field with absolute form of advection.
- $\boldsymbol{W < 0}$: Vorticity dominated region.

The OW parameter provides an exceptional distinction between the different regimes of flow and has in multiple studies been used as a means of identifying and tracking the location of eddies. Figure 2.5 shows the calculated OW parameter from a sea surface flow field, the inner region of the circulatory motion of the eddy is shown to contain vorticity-dominated values, with converging (radial) gradients. As the velocity vectors start to show less circulatory behaviour, the outer rim of the eddy can be distinguished by lower absolute OW values ($W \leq 0.5$). As shown in the literature review of detection methods, the separation of the field domains has been proven to produce satisfactory results in identifying eddy cores from complex fluid flows.

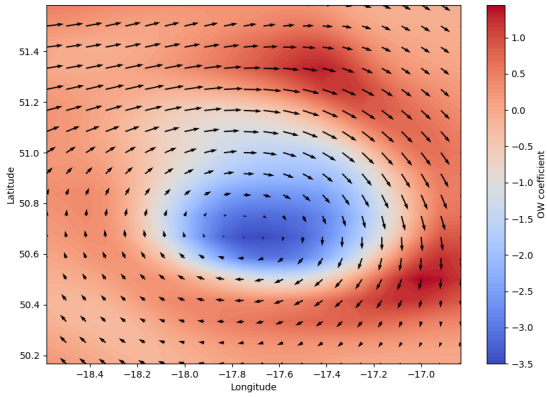


Figure 2.5: OW parameter calculated from the data assimilated northward and eastward sea surface current of one of the CMEMS datasets [3] centered around a large (~150 km) ocean eddy found in the North Atlantic. The arrows denote velocity magnitude and direction.

2.2 Artifical Intelligence / Machine Learning

The field of machine learning has taken a dramatic twist in recent times, with the rise of the Artificial Neural Networks (ANN). These biologically inspired computational models are able to exceed the performance of previous methods of artificial intelligence in common machine learning tasks. One of the most impressive forms of ANNs architecture is that of the Convolutional Neural Networks (CNN). CNNs are primarily used to solve difficult image-driven pattern recognition tasks and, with their well-defined architectures, offers a simple induction into ANNs.

2.2.1 Artificial Neural Network (ANN)

The definition of artificial is something made by human beings as a copy of something natural, while intelligence is the ability to understand, think, and learn. Although there are many definitions of intelligence, common denominators are learning, understanding, and applying previous experiences to achieve one or more goals. Artificial intelligence (AI) is a broad area of computer science committed to making machines that mimic human intelligence. Machine learning is a set of AI methods that are responsible for the ability of an AI system to learn. ANNs, such as CNN and SVM, and other methods, such as random forest, are a few common ML methods. Neural networks are inspired by biological processes and the ability to learn from experience. At a very high-level description, the neurons interoperate by activating one another through an interface of axon (nerve fiber) terminals that are connected to dendrites across a gap (synapse). In simpler terms, a neuron will pass a signal to another neuron across an interface if the sum of the signals (the energy potential) from one or more neurons at the interface is great enough. This build-up of input and the exceeding of a threshold activating the subsequently connected neuron is known as activation.

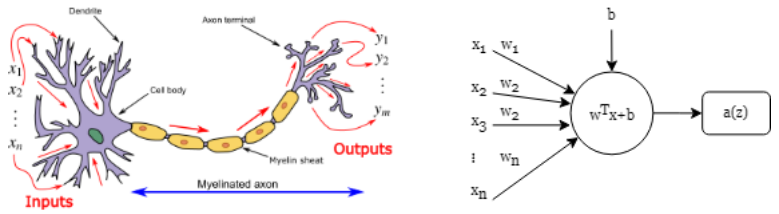


Figure 2.6: A neuron in the brain is activated when it receives signals that reaches the neuron's membrane potential causing the flow of ions. The flowchart to the right is an artificial representation of a biological neuron. Each input signal is given a weight decided by fortifying certain pathways and an activation function, much like the membrane potential determining if the neuron should be activated.

The activation of both biological and ANNs is more complex than simple summation, as all of the many signals passing between neurons have some synaptic weight. As a neuron in an emerging neural pathway is fortified depending on its importance for learning a specific problem. The neuron also applies a transformation function to the weighted inputs, which is then evaluated to see if it exceeds a given threshold, much like the action potential during

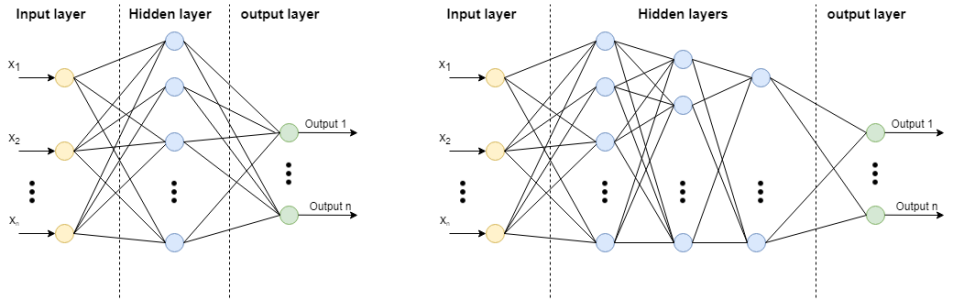


Figure 2.7: The left network depicts a simple (shallow) single-layer neural network. The network to the right is a more complex (deep) neural network. The deep neural network can process raw data through many layers of non-linear transformations, deriving, or constructing meaningful features to calculate a target output. On the other hand, a shallow network tends to be less complex and often requires up-front extraction and construction of meaningful features to enhance the model's ability to predict the correct outcome

the firing of a neuron in the brain [39]. Figure 2.6 shows an example of how each neuron accumulates (summate) weighted signals from the input vector with a bias, which is then transformed by an activation function that determines whether it should be activated and to what degree.

Much like an image classification problem use images with multiple RGB channels as inputs, the input signals in the brain can originate from many sensory systems, for instance, feeling, smelling, and seeing. The ultimate goal is the emergence of billions of activate neurons instructing actions, memory recollection, and so forth. The processing of our brain, also referred to as *thinking*, and the subsequent instructions to the neurological actuators (e.g., muscles and organs), are the ultimate result of a neural network in action. In addition to the processing, the brain's neural network is continuously modified as a result of experiences and learning.

In a simple ANN, as depicted in the first image of figure 2.7, we have the input layer, followed by one hidden layer, and lastly, an output layer. The models can be increasingly complex with increased problem-solving capabilities by increasing the number of neurons, hidden layers, and the number of pathways. Deep learning is often used to describe neural networks with an increased number of hidden layers (increased depth), such as the latter image in figure 2.7. Deep networks can process raw data through many layers of non-linear transformations, deriving or constructing meaningful features to calculate a target output. Shallow networks, on the other hand, tend to be less complex and often require up-front extraction and construction of meaningful features, to enhance the model's ability to predict the correct outcome.

2.2.2 SVM

Support-Vector Machines (SVM) is a supervised learning algorithm that was first theorized back in the 1960s and 1970s. Since then, the algorithm has been widely used in both industry and academia. The SVM has been extensively proven to deliver higher classification

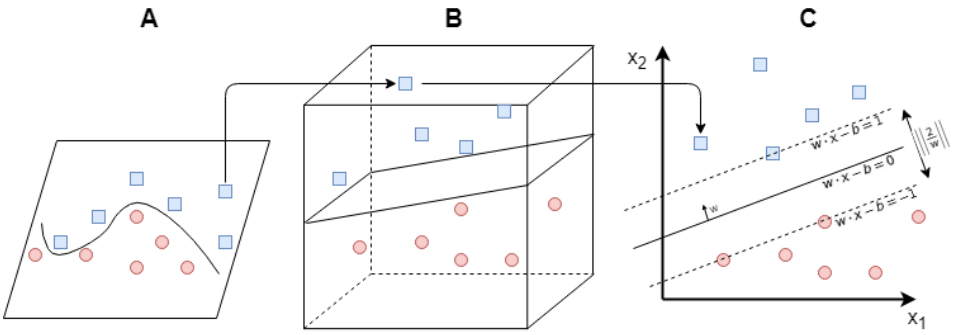


Figure 2.8: 1 does not separate the classes. H2 does, but only with a small margin. H3 separates them with the maximal margin. (Change this description)

accuracy performance than other classification algorithms. A particular SVM property compared to other classifiers is that it can simultaneously maximize the geometrical (classification) margin and minimize the empirical classification error; hence SVM can be called a *maximum margin classifier*. SVMs transforms the input vectors to a higher dimensional space in which it constructs a maximal separating hyperplane.

A SVM constructs a hyperplane or a set of hyperplanes in a higher dimension (in infinite dimensions), which best separates the data into classes. Simply put, the algorithm transforms the data into the higher dimension in which it can create a linear decision boundary that separates the data into classes. In figure 2.8, the SVM is presented initially with a non-linearly separable input vector, which is mapped to a higher-dimensional space, by use of a kernel function, in which a maximal separating hyperplane is constructed. On each side of that first hyperplane, it constructs two additional parallel hyperplanes. Classification in higher dimensional space is usually associated with over-fitting; however, the SVM minimizes the empirical risk of misclassification by maximizing the decision boundary [40]. The algorithm assumes that the larger the distance between these parallel lines, the better it predicts the class of previously unseen data [41].

Consider data points in the training set of the form:

$$\{(x_1, y_1), (x_2, y_2)...(x_n, y_n)\}$$

where y_i is either 1 or 0, denoting the class of the data point. The data x_i is an n-dimensional real and usually scaled vector. The importance of scaling to reduce the disproportional influence of variables with large variance, and feature engineering in general, will be introduced in a subsequent section. After pre-processing the data, the features are mapped to a higher-dimensional space, as illustrated in image B of figure 2.8. The most common kernel is the Gaussian radial basis function [42].

The SVM finds the largest margin hyperplane by choosing support vectors and tuning the weights and bias of the margin. The SVM has another set of parameters that are called hyper-parameters: The soft margin constant C and any other necessary parameters for the kernel function used (width of a Gaussian kernel or order of a polynomial kernel). A grid search method is common for supervised classifiers that rely on hyper-parameter selection.

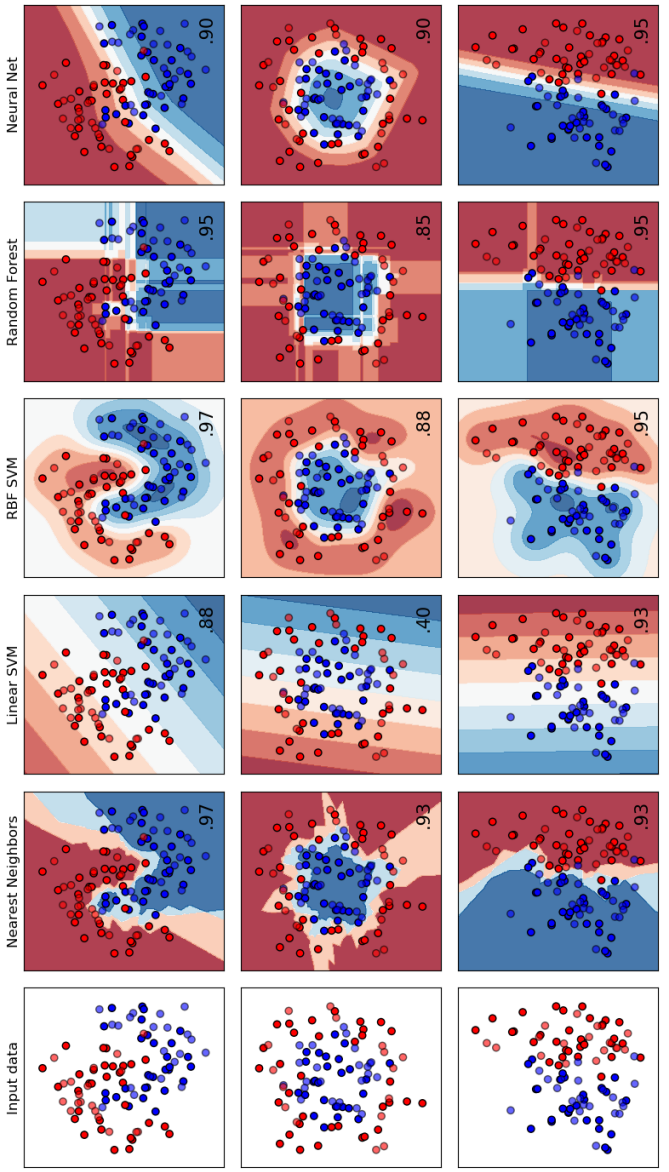


Figure 2.9: A comparison between a few key classifiers from the scikit-learn [43] on a synthetic dataset. The input data is synthetic, and the decision boundary intuition does not necessarily transfer over to real-life samples. The classification is a binary problem in which blue and red are two different classes that the classifier will learn to distinguish as effectively as possible. The training data is the solid colored data points, and the test data is the same color but semi-transparent.

The grid search will methodically build and evaluate a model for every combination of hyper-parameter specified in a grid.

If the SVM is implemented without using the kernel trick, the hyperplanes are linear and roughly equivalent to a simple feed-forward neural network, as the one shown in figure 2.7, without an activation function. Likewise, with the kernel trick, the SVM is roughly equivalent to a simple neural network using non-linear activation functions.

The SVM scales well with increased feature space, i.e., a lot of input features, such as high-resolution images. As an example, imagine that the input has a total of 1,000 features (dimensions), the kernel space could be its square (1,000,000) or even infinite. However, the only operation SVM is performing is calculating the distance between each data point, which is then used as input to the kernel machine. The trade off, however, is that number of distances severely increases as the number of samples (support vectors) increase. Figur 2.9) shows how the linear SVM's decision boundaries cannot cope with data following a non-linear class function but when employing the RBF (Radial Basis Function) it can create a linear boundary in a higher-dimensional space.

2.2.3 Random forest

The random forest algorithm has a hierarchical model structure, referring to the hierarchical splits between features in a tree structure that generate an optimal decision boundary, which can help classify even isolated groups of features. Hierarchical models are particularly suitable for modern systems of big data and distributed machine learning. They adopt both classification and regression strategies for a tree built as a sequence of decisions, also called a decision tree. There are two types of decision trees: regression trees and classification trees [44]. A decision tree employs a rule-based approach to create multiple linear splits to the domain and predict the response. The decision tree has a flowchart like structure in which each internal node represents a choice on a feature, each subsequent branch represents the outcome of the test, and the leaf nodes (last nodes of the tree) represents class labels. Figure 2.9 shows the decision boundaries created by several decision trees making horizontal and vertical cuts in the features. The ability to separate and isolate certain features makes the decision tree highly suitable for multiclass classification.

Random forest is categorized as an ensemble machine learning method called bagging [45]. Ensemble methods employ multiple machine learning models or methods to combine their predictive power and improve performance. Ensemble methods usually produce more accurate predictions than, for instance, a single decision tree would. **Bagging** is short for "bootstrap aggregating". Bootstrapping is a technique of statistical resampling that involves random sampling with replacement [46]. The idea is to create different instances of the original data by random sampling with replacement. After n numbers of decision trees are built using the bootstrap method, each tree's final decision is aggregated by using a majority vote to decide the final prediction. Figure 2.10 shows how the input data is split into n instances using the bootstrap method for which n number of decision trees are built, whose outcomes are aggregated into one single prediction.

As for the SVM, the random forest algorithm contains certain hyperparameters that need to be adjusted to optimize the performance. Hence, the optimization of hyperparameters using knowledge about the problem domain is a crucial task to increase the performance of the algorithm. The parameters for a random forest algorithm is the number of features

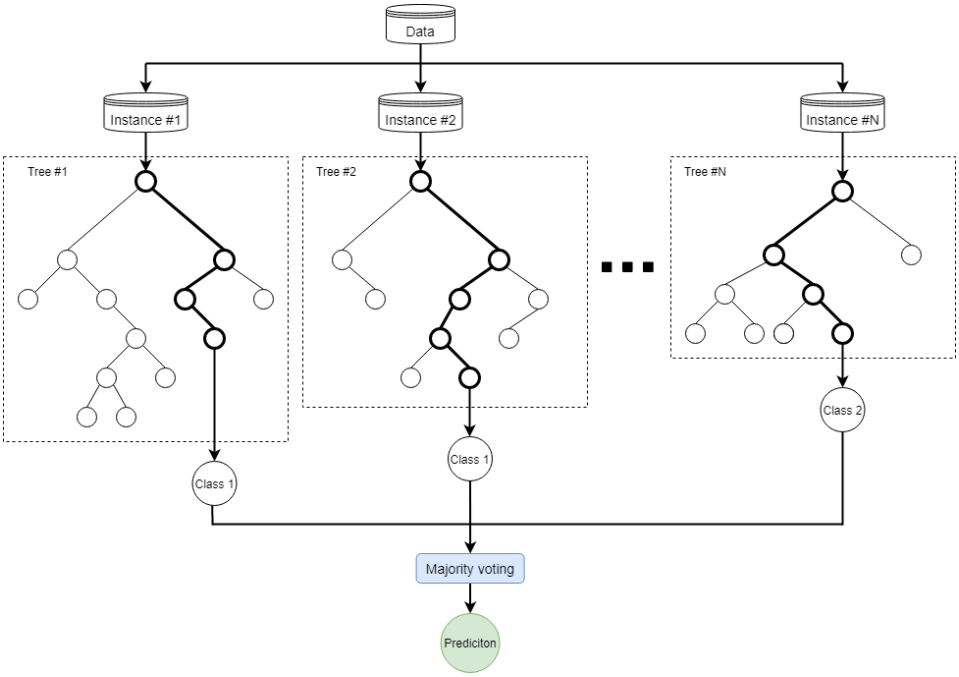


Figure 2.10: The random forest algorithm consists of multiple learnt decision trees which has a flowchart like structure in which each internal node represents a choice on a feature, each subsequent branch represents the outcome of the test, and the leaf nodes (last nodes of the tree) represents class labels. After n numbers of decision trees are built using the bootstrap method, each tree's final decision is aggregated by using a majority vote to decide the final prediction.

considered by each tree when splitting a node and the number of decision trees. As for the SVM, the hyperparameters are usually found using grid search methods that build and evaluate models from combinations of hyperparameters.

2.2.4 Convolutional Neural Network (CNN)

Prior to the development of deep neural networks, shallow networks, such as the SVM, relied heavily on extracting variables of interest (features). For most problems, the feature extraction procedure required a great deal of experience, especially for image processing. The invention of convolutional neural networks (CNN) has since revolutionized image and pattern recognition and eliminated most of the manual labor related to feature extraction. An essential improvement compared to an ANN is the number of parameters. If one were to use a fully connected deep neural network, much like the one shown in figure 2.7, it would take a considerable amount of parameters (weights and neurons) to characterize the network. CNN considerably reduces the number of parameters by using smaller convolutional filters to connect receptive fields to neurons in subsequent layers, as shown in figure 2.11. The increased depth scalability has prompted researchers and developers to experiment with larger and deeper networks to solve more complex tasks.

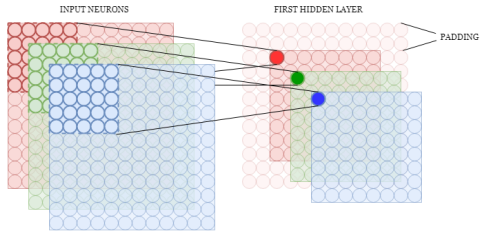


Figure 2.11: Visualization of a 5x5 filter convolving around an RGB input volume and producing an activation map. With a step size of one and padding at the edges of the output, the feature map remains the same size as the input.

CNNs takes the neural network's bionics to the next level by using biological inspiration from the visual cortex. The visual cortex (the part of the cerebral cortex that processes sensory nerve impulses from the eyes) engages with small regions of cells sensitive to specific regions of the visual field. The idea first appeared after Hubel and Wiesel showed that some individual cells in the brain fired only in the presence of edges with a certain orientation [47]. Imagine a flashlight sliding across the areas of an image or dataset, creating multiple smaller receptive areas. In machine learning, this flashlight represents a convolutional filter (sometimes referred to as a kernel), and the region of interest it creates represents a receptive field.

An important premise of CNNs is the spatial independence of features. In other words, in an object detection application, a CNN will identify features that belong to the object regardless of location. Another significant aspect of CNN is the exposure of more abstract and complex features as the input vector propagates towards the deeper layers. For instance, the first layers might detect the edges of the object, the shapes of the object in the second layers, and other higher-level features for subsequent layers.

There are four main layers in a conventional CNN: Convolutional, rectified linear unit (activation function), and fully connected.

2-D convolutional layer

The 2-D convolutional layer uses convolutional filters to the two-dimensional input vector. The layer looks at different receptive fields of the input by moving horizontally and vertically along the input while computing the dot product of learned weights. The convolutional layer's output is the most informative mapping of features computed by the input and the weights of the kernel. For instance, the first hidden layer in figure 2.12 applies five 3x3 convolutional filters of size 3x3 each (3x3x5 filter). The sides of the output can either be zero-padded or the same value as their neighbor to ensure that the size of the input and output are the same, otherwise the output matrix would be slightly smaller depending on the filter size[1]. The convolutional layer is connected to an activation function, the most common being the rectified linear unit (ReLu),

[1]Recent studies have argued whether it is more beneficial to use convolutional layers for downsampling instead of pooling layers, e.g., by omitting the padding step.

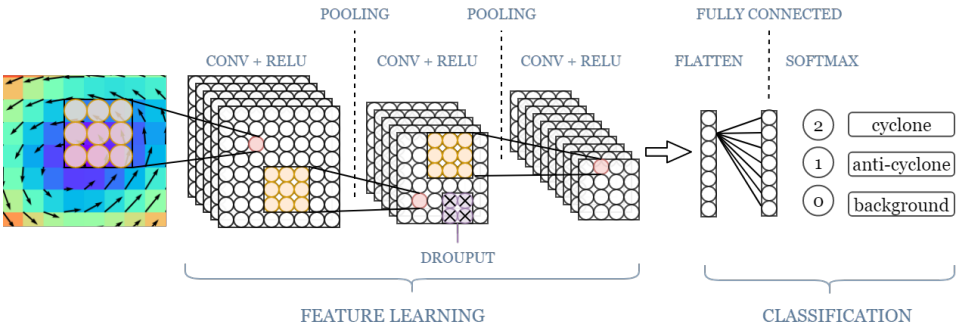


Figure 2.12: Visualization of how a CNN would handle classifying a grid of ocean variables as either having a cyclone, anticyclone, or nothing present in the grid. The first hidden layers is usually a sequence of convolutional layers combined with a ReLu (leading activation function for CNNs), followed by a pooling layer (downsampling). Each subsequent block of hidden layers extracts more complex and abstract attributes, and all feature maps are flattened to create a single long feature vector, which is then connected to one or more fully connected layers. Dropout is used to randomly ignore neurons after pooling for increased regularization. The fully connected layer is much like a SVM in that it tries to learn the (possibly non-linear) function that predicts the class from the feature maps.

$$\sigma(x) = max(0, x) \tag{2.9}$$

which rectifies negative filter outputs to zero, but positive values are the same. Although the ReLu seems like a linear activation function, which would uphold the property

$$f(x) + f(y) = f(x + y),$$

it can be shown that

$$f(-1) + f(1) \neq f(0),$$

which is a non-linear property. Much like for the SVM, the added non-linearity of the activation function allows us to build arbitrary shaped curves on the feature plane.

pooling layer

The pooling layer's objective is to downsample the input vector, reducing the dimensionality, and allowing for assumptions to be made about the feature map in the binned subregions. The pooling layer serves as one of many forms of regularization by progressively reducing the spatial size of the representations, reducing the number of computations, and also controlling overfitting. In short, regularization is the act of tuning or selecting model complexity to avoid unnecessary complexity and overfitting. Two of the most common pooling techniques are the average pooling, yielding the average presence of a feature, and max pooling, yielding the most activated presence of a feature. The pooling layers are also useful due to their ability to detect features regardless of location, i.e., the model's

invariance to local translation. Max pooling achieves partial invariance to small translations to a larger degree than average pooling, as the max of a region mainly depends on the largest element.

dropout layer

The term dropout refers to the act of neglecting the activation of a specified factor of random neurons during the training phase. These neurons turned off and not considered during a particular sequence of either forward (calculations) or backward (backpropagation) pass. Dropout layers are also a key contributor to regularization, preventing over-fitting and slightly reducing the number of computations. The fully connected layers towards the classification section of the CNN shown figure 2.12 tend to develop codependencies among each other during training, hindering the individual power, leading to overfitting of the training data.

Flattening layer

The flattening step the final link between the activations generated by the feature learning step and the fully connected layer(s). Each feature map channel (e.g., multichannel for RGB images and single-channel for greyscale) in the output of a CNN layer is flattened, creating one large one-dimensional layer.

Fully connected layer

The fully connected, also referred to as a dense (a densely connected) layer by TensorFlow ML platform, is placed between the flattened vector and the output prediction. All neurons from the previous layer are connected to all the neurons in the fully connected layer. Just as the convolutional layers, the fully connected layers often use activation functions to determine the output of the neurons.

2.2.5 State-of-the-art CNN architectures

ImageNet [48] is a large visual database used for visual object detection research, with more than 14 million hand-annotated images. Every year both the ImageNet project and other leading workshops run an annual software contest, where different detection software compete to classify and detect objects.

In 2012 a CNN called AlexNet [49] achieved 15.3% error in the ImageNet challenge, more than 10.8 percent than second place. Being the first architecture to push ImageNet classification accuracy by a significant stride and shock the AI community, it was a milestone for deep CNNs.

This project has implemented three prominent architectures from the ImageNet challenges throughout the years. The contestants are the VGG, ResNet, and Inception network architectures. The VGG [50] network is an improvement over AlexNet by replacing their large convolutional filter sizes (11 and 5) with multiple smaller 3x3 filters. The ResNet (residual neural network) [51] utilizes identity connections that shortcut parts of the network, enabling deeper network without over-fitting and barely any increase in computational

power. The final network architecture is the Inception architecture (GoogLeNet) [52], which uses a parallel set of layers to the same input instead of using conventional sequential convolutional filters.

2.2.6 VGG16

The VGG16 network architecture has a straightforward structure made of sequences of VGG blocks. Figure 2.13 shows a diagram of the original VGG16 network architecture. The convolutional filters have smaller receptive fields (3x3) made to capture the spatial perceptions, such as the direction or location of features. The convolutional stride is set to 1, preserving the spatial resolution as long as the input's edges are padded. Five max-pooling layers using 2x2 windows perform the spatial downsampling following the convolutional layers. Each of the five stages can be called a VGG block, which consists of two convolutional layers, followed by a pooling layer.

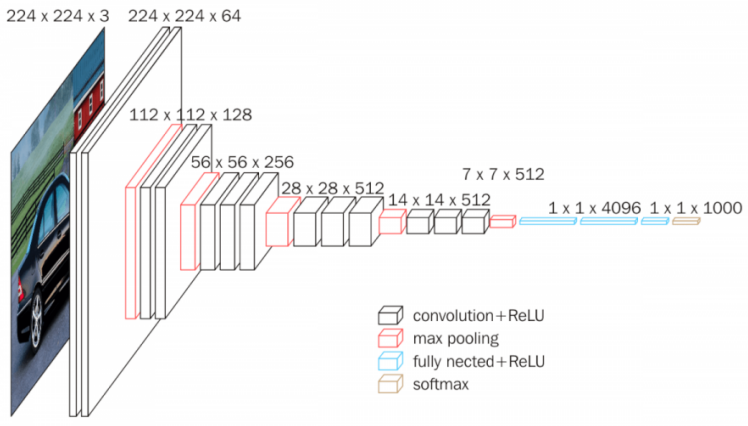


Figure 2.13: VGG network, modified image from [53].

Multiple modifications of the VGG16 architecture has been employed in different parts of this project because of its simplicity and computational efficiency. When comparing machine learning methods, a sequence of modified VGG blocks was used. Figure 2.14 shows the simple VGG network, which employs three modified VGG blocks, each consisting of two 3x3 convolutional layers, a 2x2 pooling layer, and a dropout layer. Following the three blocks is two fully connected layers, the last of them predicting one of three classes (cyclone, anticyclone, or background).

ResNet

AlexNet and VGG introduced the conventional convolutional layers, followed by pooling. The intuition being progressively increasing the depth enables the model to learn more complex features. The first layers detect the edges; the second layers detect shapes; the third detects objects, and so forth. Despite the AI community's persistence to go deeper, He et al. [51] shows that there is a threshold to error achieved with increased depth using

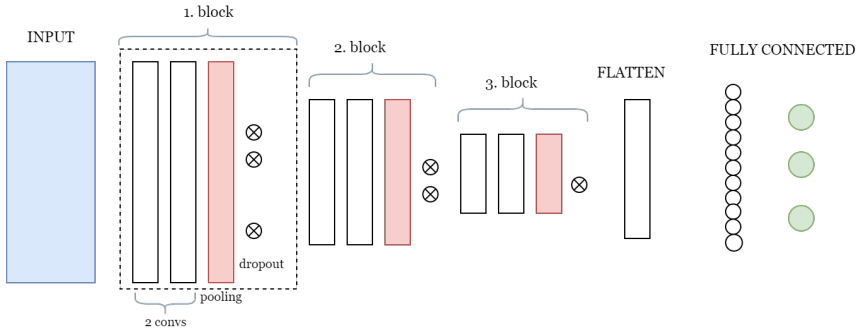


Figure 2.14: A simple network architecture made up of a sequence of three modified VGG blocks whose feature vectors are flattened and connected to a larger fully connected layer, before the final fully connected layer predicts one of three classifications.

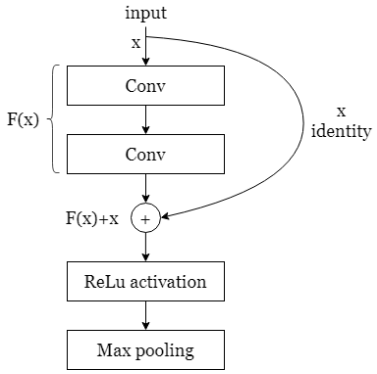


Figure 2.15: A residual block, consisting of the conventional convolutional layers with an added skip connection. The skip connection is an identity mapping that adds the input from the previous layer to the convolutional layers. The identity connection allows the network to discard the convolutional layers in that block if deemed inadequate.

the traditional networks, i.e., at some point, the deeper networks starts to perform worse than the more shallow ones. The vanishing gradients are especially to blame for the underperformance of deeper networks. To alleviate the poor performance of very deep neural networks, ResNet introduces a new layer to the introduced VGG block, a residual layer, renaming the block of layers as a residual block.

A residual neural network (ResNet) is a kind of ANN that is based on the structure of the pyramid cells found in the pyramidal neurons found in the cerebral cortex. The pyramidal neurons are shaped like a pyramid, in that it has multiple dendrites (receiving electric stimuli) connected to a single axon. Compared to normal neurons, the dendrites of pyramidal neurons perform some form of skip connections [54], which is what the ResNet mimics by identity mapping the input and bypassing the conventional convolutional layers as shown in figure 2.15. The identity connection allows the network to discard the

convolutional layers in that block if deemed inadequate.

To understand the intuition behind calling it a residual network, one has to look at the difference between the true value H(x) and the predicted value of the:

$$R(x) = Output - Input = F(x) - x \tag{2.10}$$

where R(x) is the residual of the difference between the input and the true distribution of the output f(x) and the input x. If we rearrange the equation, we get

$$F(x) = R(x) + x, \tag{2.11}$$

which tells us that the convolutional layers tries to approximate both the residual and the input x. However, since we feedforward the input x through an identity connection, the layers are only trying to learn the residual R(x). It has been shown that it is easier to learn the residual of an output [51].

The standard residual neural network consists of different types of residual blocks; however, like with the VGG block, this project employs a more simple modified version. The ResNet block is much like the VGG block shown in figure 2.14, only with a unity mapping bypassing the two convolutional layers.

Inception (GoogLeNet)

The GoogLeNet, also known as Inception V1, from google achieved an error rate of 6.67% in the 2014 ImageNet competition, which was very close to the average human performance. The network's primary innovation was the inception module. The main idea behind the

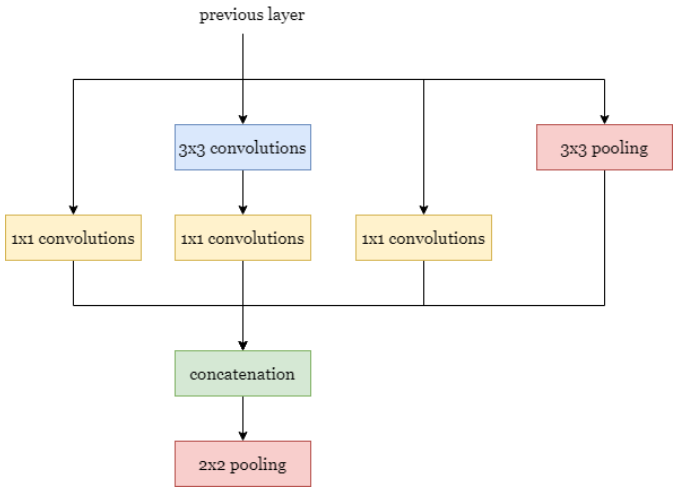


Figure 2.16: A residual block, consisting of the conventional convolutional layers with an added skip connection. The skip connection is an identity mapping that adds the input from the previous layer to the convolutional layers. The identity connection allows the network to discard the convolutional layers in that block if deemed inadequate.

module is to let multiple filter sizes operate on the same input level, opening possibilities to detect features only visible to a specific filter size. The most noticeable parts of an image usually have a considerable variation in size. Because of the variation in the location of important information, choosing the correct kernel size can prove a difficult task requiring a great deal of domain expertise. Usually, the larger kernels are preferred for more globally distributed information, while the smaller kernels are used for more local information. Furthermore, the very deep networks are notoriously prone to overfitting, and the effect of gradient descent starts to diminish.

Figure 2.16 shows the inception module (or block) used to create a modified inception network architecture. The module contains parallel convolutional layers with differing filter sizes: 1x1, 3x3, and 5x5, and a 3x3 max pooling layer, all of which use padding and a stride of one to maintain the vector size. The feature maps of each branch in the parallel structure are then concatenated before they are pooled for dimensionality reduction.

2.2.7 Partitioning training data

The first step in any machine learning project is obtaining knowledge about the problem domain and strategize on how to gather or generate relevant data. After the source of data and its appropriate structure is identified, and a sufficient amount of data is extracted, the engineer needs to train and evaluate a machine learning algorithm. The first step in training and evaluating a model is partitioning the dataset into a training, validation, and holdout set.

- Training set — Which you run your learning algorithm on.
- Dev (development) set — Which you use to tune parameters, select features, and make other decisions regarding the learning algorithm. Sometimes also called the validation set.
- Test set — which you use to evaluate the performance of the algorithm, but not to make any decisions regarding what learning algorithm or parameters to use.

Figure 2.17: Training data is partitioned into three data sets, training, validation, and test.

Training the algorithm, learning the hyper-parameters of the prediction function, and testing the model on the same data is a methodological mistake: The model would simply repeat the labels of the data it has already seen and get a perfect score but fail miserably when presented with unseen samples. To avoid this situation, it is common practice when performing a supervised machine learning project to hold out a part of the available data as a training and test set, usually a split of about 70-30 percent. The test split is used to evaluate the performance of the final model. The training set is then usually split into a training set and an evaluation set, which is used used to assess various models during training. Figure 2.18 shows how Scikit-learn's grid-search [43] algorithm evaluates different combinations of hyper-parameters (e.g., the soft margin C and other kernel dependent hyper-parameters for the SVM algorithm), and retains the best performing model.

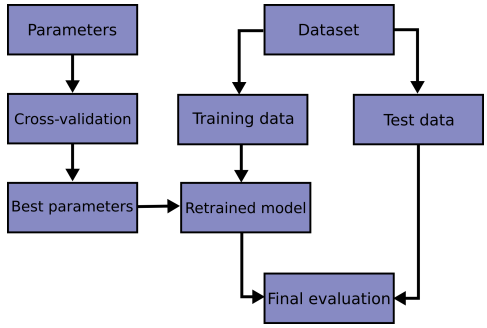


Figure 2.18: To avoid overfitting, certain machine learning techniques utilize cross-validation techniques to evaluate hyper-parameters and models during training and retain the best performing model. The image is taken from Scikit-learn's website about cross-validation [43].

2.2.8 Feature engineering and selection

A machine learning engineer will spend about 80 percent of their time performing feature engineering, and the remainder 20 percent is allocated for training the model and performing hyperparameter optimization. Performing the proper feature extraction is crucial for improving the performance of a machine learning algorithm. Feature engineering is the process of taking unrefined raw data and converting it into meaningful features that the machine learning algorithm can understand better decision boundaries.

A feature is a numerical representation of raw data [55]. Raw data is usually unstructured and messy, but there are several ways of transforming it into so-called features that hold relevant information with respect to what a machine learning algorithm is trying to predict or solve.

Feature selection

A common method of feature engineering is to either use the intuition of an expert or simple feature selection method to prune away less useful features. The training data might contain samples that are affected by noise or outliers, which will eventually mislead the machine learning algorithm and reduce its performance. By removing redundant and misleading features, the model will have reduced dimensionality, i.e., it is quicker to compute, and less degradation in its performance. There are three main classes of feature selection techniques: Filtering, Wrapper methods, and Embedded methods. In short, the filtering method filters out features on the count of correlation and mutual information, wrapper methods iteratively refines subsets of features, embedded methods perform feature selection as a part of the model training process. Decision trees inherently perform feature selection as they continiously create a split decision between two features at each step. CNN also performs some form of embedded feature selection by using convolutional kernels that can virtually shut down features (neurons) when training the model.

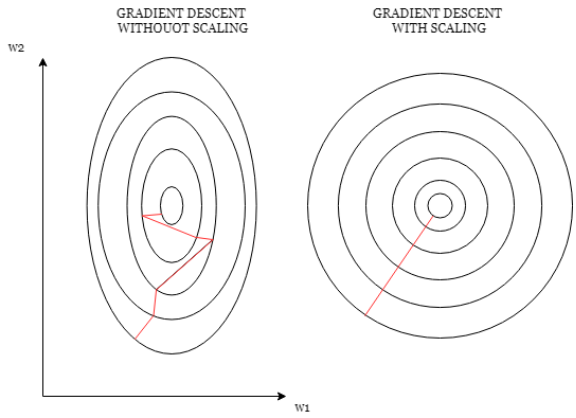


Figure 2.19: Gradient descent: gradient descent on unscaled vs scaled data.

Feature scaling

We cannot calculate the perfect weights for a neural network as there are too many unknowns. Instead, the problem of learning is cast as a search or optimization problem and an algorithm is used to navigate the space of possible sets of weights the model may use in order to make good or good enough predictions.

Most machine learning models that are smooth functions of the input, e.g., anything that involves a matrix, are sensitive to large variations in magnitudes, values, or units. Feature scaling is often an essential technique used to standardize the independent features in the training data. The first plot in figure 2.19 illustrates how a gradient descent algorithm [56] always points perpendicular to the contours of the cost function, which means a skewed feature distribution causing it to ”zig-zag” towards the optimum. By normalizing the features, as done in the second illustration of figure 2.19, the gradient descent will faster converge toward the optimum of the cost function. In addition to the faster convergence of the optimization algorithm, un-scaled features tend to be more influenced by outliers and features with values that are much larger than the rest of the dataset. In the feed-forward mapping between input and output in a neural network, the activation functions can cause larger feature values to blow up, which can cause undesired saturation effects.

The min-max is the simplest one, as it simply squeezes the features between a given range, usually between 0 and 1, preserving the features’ distribution. The min-max method is useful when the ML algorithm does not make any assumptions about the distribution; however, it is very sensitive to outliers.

The standardization method subtracts the features’ mean and divides it by its variance; the resulting scaled features have a mean of 0 and a variance of 1. The standardization method is often used when features have large variations in scale and units, and if the distribution is known and gaussian.

2.2.9 scoring functions

Once the model is built, its performance needs to be appropriately evaluated. The most common measure of performance is its accuracy, which provides a percentage based on the number of both correct positive and negative predictions for all classes divided by the total number of predictions made:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.12}$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative. However, the accuracy is inappropriate for classificaitons with an imbalanced dataset. If the majority of samples have the same class (or classes), it might overwhelm the samples with less frequent classes, meaning that even an inadequate or over-fitted model can achieve scores higher than 90%, depending on the severity of the class imbalance. As an improved examination of the model's performance, multiple scoring methods are used, such as precision and recall metrics.

precision

Precision is the sum of true positives across divided by the sum of both true positives and false positives for a given class:

$$Precision = \frac{TP}{TP + FP} \tag{2.13}$$

If the dataset contains samples with three different classes, the average precision would be the total accuracy of the model. If there is an imbalance in the training data, the precision for each class would quantify the ratio.

recall

The recall quantifies the number of correct predictions made from all positive predictions that could have been made for a given class. The recall is calculated as the sum of true positives divided by the sum of true positives and false negatives for a given class:

$$Recall = \frac{TP}{TP + FN} \tag{2.14}$$

Unlike precision, that only provides a measure on the correct positives out of all positive predictions made; the recall provides a measure of the missed positive predictions.

F1-score

The F1 score is the harmonic mean of both the precision and recall, i.e., it considers both false positives and false negatives:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.15}$$

The F1-score is simply a combined metric. The evaluation is often used if one struggles to determine the superior model based on recall and precision. F1-score can provide a means of assessment if two models outperform one another in precision and recall.

Chapter 3

Methods

3.1 Training data

Most of the resources spent developing a machine learning project are not designated towards designing and developing the hyperparameters or network structure but usually spent analyzing and extracting training data and modifying and selecting features. The main scope of this project is to investigate the possibility of creating a system that could provide a daily update on eddies' location to the SINMOD system. Daily means of ocean variables were downloaded from the CMEMS project, for which three applications of extracting annotated training data were created and evaluated.

The datasets offer a multitude of ocean variables with various temporal resolutions. Four standard variables were chosen as candidates for training a machine learning model: sea surface height (SSH) anomalies, sea surface temperature (SST), and eastward and northward sea surface current velocity (uvel and vvel). The final system should be able to use grids of either individual variables or an aggregate of variables to successfully predict and distinguish between cyclones, anti-cyclones, and background. The cyclones and anti-cyclones consist of well-defined coherent properties, making it easier to provide an adequate and representative set of training samples to train the machine learning models. The model's ability to distinguish between an eddy and the background, on the other hand, is more complicated. The algorithm requires a diverse and representative set of background samples to infer background from every situation besides those with an eddy present.

3.1.1 The datasets

The datasets used in this project originate from three different approaches to either measure or simulate the behavior of key ocean variables. One of them is the SINMOD model, which is a ocean modeling system developed at SINTEF. The other two are from the Copernicus Marine Environment Monitoring Service (CMEMS), one of which is a global ocean analysis and forecast system, which will be referred to as CMEMS-phys, the other is a combination of satellite and in-situ measurements, which will be referred to as CMEMS-multiobs (multi

observation).

SINMOD A numerical ocean modelling system created and operated by SINTEF. The main objective for this project is to investigate the possibility of of a system capable of working as a ocean eddy detection service for the SINMOD model, as discussed in section 1.3. The SINMOD datasets provides hourly variables on a 4 km polar stereographic projected grid. An example of the SSH from the dataset is shown in the right plot of figure 3.1

CMEMS-phys A analysis and forecast system provided by the CMEMS project, which use assimilations of global ocean observations in order to estimate oceanic variables. The CMEMS-phys datasets downloaded for this project contains daily mean values. The annotated training data generated to train the machine learning models were extracted from the CMEMS-phys datasets. The datasets provides daily mean values on a 0.083° regular grid [3]. An example of SSH from the dataset is shown in the right plot of figure 3.1.

CMEMS-multiobs A combination of satellite and in-situ measurements. These datasets were employed as a reasonableness test for the machine learning models, i.e., the models trained on the CMEMS-phys model data, as they should perform just as well on the CMEMS-multiobs measurement data. The CMEMS-multiobs datasets provides variables on a 0.25° regular grid at a weekly period [4].

The main scope of this project is to investigate the possibility of creating a system that could provide a daily update on eddies' location to the SINMOD system. The datasets used for this project are either downloaded as or modified to be daily means of the ocean variables. Another reason to use a daily mean is to avoid the unwanted noise produced by external ocean events and phenomena such as tides and other cyclic ocean currents. These external factors will, in most places, add undesired variability to the sea surface variables we will be using for this project. By either averaging from hourly samples or interpolating from weekly samples to a daily value, we accentuate the effects of ocean eddies, which can live from 10 to 100 days.

The annotated training data used for training the machine learning models were extracted from the CMEMS-phys datasets because of its availability, reliableness, and the fact that it offers a product with daily means of the variables of interest. All subsequent sections will only contain data and figures produced by employing the CMEMS-phys dataset.

3.1.2 Data collection and annotation

Most of the resources spent developing a machine learning project are not designated towards designing and developing its structure or hyperparameters but usually spent analyzing and extracting training data and modifying and selecting features. This project was no exception, and although multiple research projects have tried to tackle the eddy

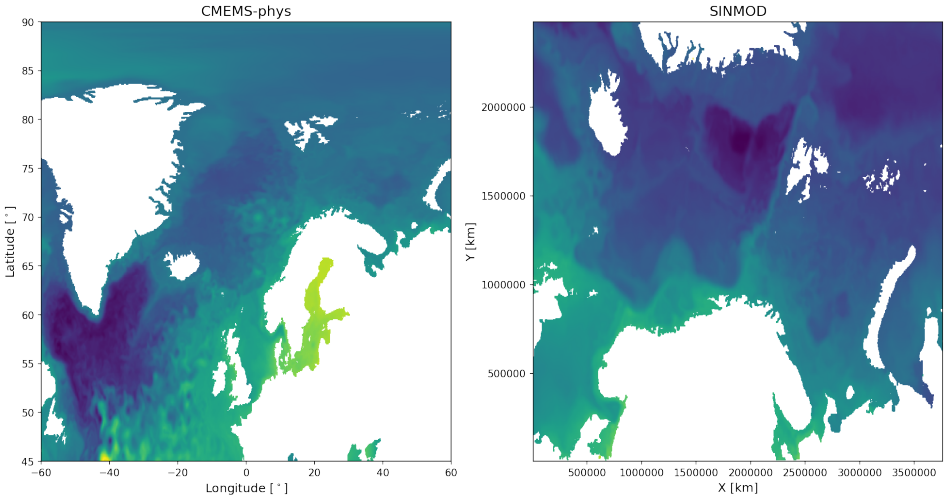


Figure 3.1: The left plot shows the SSH from the CMEMS-phys dataset [3] on a standard geodetic coordinate system while the right plot shows the SSH from the SINMOD dataset, which uses a polar-stereographic projection with a shifted north pole. Both plots show variables found for 2017-06-05, for which the CMEMS-phys has out of the box daily averages, while the SINMOD plot is averaged over 24 hours of hourly fields. Both grids cover mostly the same area, making them applicable for further comparisons. The white areas indicate the masked landmass from the respective models.

identification and tracking problem, there is no publicly available data, i.e., no annotated grid arrays or metadata such as eddy core coordinates. Most of the state-of-the-art methods seem to either employ experts that manually distinguish the classes or recognized methods such as the Okubo-Weiss method with a strict threshold to establish a few, yet precise areas that are very likely to contain an eddy.

Unlike an object detection algorithm [44], in which each object is segmented or captured within boundaries of the grid and everywhere else is labeled as background, this system should be able to classify cyclone, anti-cyclone, and background. The machine learning model will be a multiclass binary classifier [57], which means that the training and test samples will contain only one of the three classes. After searching through scientific papers and web sites for either out-of-the-box training samples or a simple tool for labeling any other metrological or oceanographical training data, there understanding was that the training data either had to be created manually or by creating some special-purpose tool. Manually extracting subgrids and labeling them is a viable option; however, machine learning algorithms generally require a substantial amount of training samples. If one were to create an automated collection system, there is a risk of spending more time building a successful application compared to time spent collecting data. Three concepts of data extraction were tested: A fully automated, semi-automated, and a more manual approach.

Automated data collection

The initial premise was to create a fully-automated process of collecting training data from the CMEMS data. The intention was for parameters such as Okubo Weiss (OW) or vorticity to discover the core and the neighboring cells that belong to an eddy. The appliance of OW parameter and vorticity to determine eddy properties was explained in section 2.1.6, and will play a key role in the post-processing introduced in a later section. The algorithm uses a user-defined threshold, for which cells with OW values below this threshold remain for consideration as eddy cells, as cells with a negative OW value, are dominated by relative vorticity. The algorithm accumulates the local OW minima of neighboring cells that are below the threshold. Each local OW minima is a potential eddy core, and the neighboring cells are accumulated until goodness of fit is below a predefined threshold. The cluster of cells is identified as an eddy if the number of cells is above predefined criteria. The eddy identification method is a modified version of a three-dimensional eddy detection method introduced in 2013, M. R. Petersen et al. [18]. This method worked well when identifying large and distinct eddies using strict criteria.

The algorithm depends on stringent criteria, such as the OW threshold, to decrease the chance of generating faulty training samples. However, this will make the algorithm favor the larger, long-lived eddies, creating undesired bias and over-fitting. Another obstacle would be collecting samples classified as background, as the machine learning algorithm requires a diverse and representative set of background samples to infer background from every situation besides those with an eddy present. A fully automated system would be the desired method for generating training data. However, a more manual approach is more reliable and much easier to realize.

semi-automatic data collection

The difficulties of creating a fully automated data collection system prompted the creation of a semi-automated system. The automated system proposes subgrids from which an expert can determine if the sample is suitable for training. Since all cells with an OW value below zero is considered to be a vorticial region, the remaining cells with OW value above zero are dominated by strain, which can potentially be used to generate samples classified as background. The system would generate training data by performing the following steps on a CMEMS-phys dataset:

- The system computes the most likely eddies, i.e., cyclones and anti-cyclones, from a netCDF file containing ocean variables.
- A simple GUI displays the variables in a grid determined by the automatic method from the most likely eddy core and its neighboring cells.
- The user can then either choose scrap the grid or store it as training data along with a given annotation.
- After examining the possible eddies, the system proposes random subgrids of strain dominated cells with an OW value above zero for the user until the proper amount of non-eddy samples are chosen.

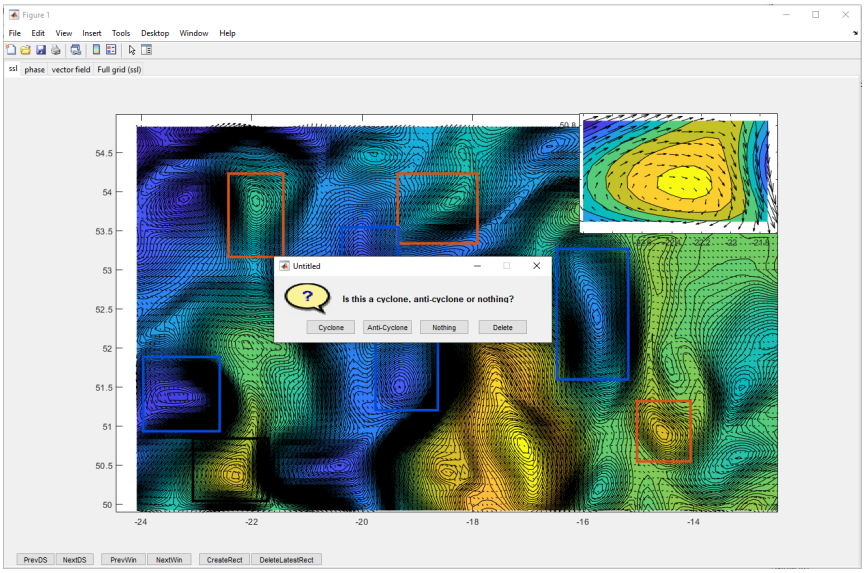


Figure 3.2: Data collection application.

This system was used to create 600 samples of ”cyclones,” ”anti-cyclones,” and ”background.” The 600 samples were then split into train and test subsets and used to train a simple CNN (in most following cases, the ”simple” CNN is referring to a 3-block VGG network with a single dense layer). As anticipated, the model performed well on the samples generated by the application but inadequately when shown more diverse unseen eddies and background samples. Because the system only suggests eddies and background samples according to the given criteria (OW parameter), it will most likely show the same or at least the most similar eddies. This concept will still struggle with some of the same problems as the fully automated concept and produce a less varied and heterogeneous data set.

Final data collection application

The final iteration of the training data collection application used a more manual approach for extracting training samples. A screenshot of the application created using MATLAB is shown in figure 3.2. The user can maneuver subgrids of a NetCDF dataset and mark a rectangular grid and decide whether the marked rectangular subgrid is a cyclone, anti-cyclone, or nothing (background). The subgrid’s ocean variables, along with a given label, are then appended to a compressed hdf5 file containing the training data. The user can choose a more diverse set of training samples, and fix the lack of diversity transpiring from the previously mentioned methods.

3.1.3 The training data

The variables considered as candidates for training data need to exhibit spatial attributes in the presence of an ocean eddy. Four fundamental variables included in all three datasets are the initial candidates as predictors: SSH, SST, and eastward and northward sea surface current velocity. The application for extracting training data stores a given grid comprised of the four variables affixed with a class annotation.

A representation of the training data is shown in figure 3.3. The heatmap in the plot shows elevated levels of SSH as a more red nuance, and more blue color indicates lower levels. The arrows represent the velocity vectors given by the east- and northward sea surface velocities. The anti-cyclone in the top left corner displays the counterclockwise motion and a raised sea surface height at the core. In contrast, the cyclone in the top right corner contains velocities with clockwise rotation and a lower sea surface height. Both the SSH and velocities displayed in the upper plots of the figure show the spatial ocean eddy properties introduced in section 2.1.4.

The bottom images show samples of the samples classified as background, in which no eddy is present within the grid. The velocities indicate little or no circulatory motion, and the SSH does not exhibit any strong local extremum. In both background plots, ocean current is shown as having as nearly monodirectional flow.

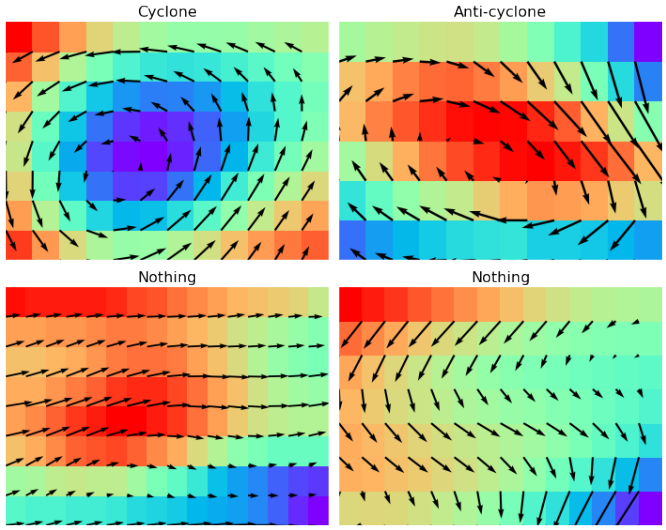


Figure 3.3: A few samples of the collected training data. The heatmap in the plot shows elevated levels of SSH as a more red nuance, and more blue color indicates lower levels. The arrows represent the velocity vectors given by the east- and northward sea surface velocities.

The most notable obstacle was creating an adequate number of samples to represent the background. The model will most likely be able to infer samples containing an eddy when trained on a smaller and not so diverse set of training samples, as they commonly exhibit low-level spatial features such as diverging SSH or SST gradients with local extrema and distinct circulatory geostrophic currents. On the other hand, the background class has to

be learned from a much more diverse set of samples. If the training samples classified as background are insufficiently diverse, the model will have increased uncertainty, which could, in some cases, affect the model's ability to distinguish the background from an eddy. Although the expert attempts to choose a diverse set of samples, he or she possesses a preference for what he or she believes to be the appropriate data to represent the background. In addition, the experts have their own conclusions of what is needed to further diversify the training data, based on the visual depiction of the GUI's data. Because of this selection preference, a simple CNN model and the manual training data extraction application were cooperatively used as a reinforcement system. After a simple initial CNN model was trained on a sufficient number of training samples, it was applied to a grid of unseen data. The same data was then applied to the GUI, much like the grid shown in the GUI from figure 3.2, and the incorrect predictions made by the model was stored as correctly annotated training data, performing a sort of reinforcement loop.

A total of 2045 samples was collected using the final data collection application and the procedures specified. A roughly equal number of samples of each class were obtained to avoid class imbalance: 725 non-eddies, 666 anti-cyclones, and 654 cyclones. Figure 3.5 shows that the accuracy of the training, validation, and test set when training a CNN model on an expanding set of training samples. Although the trend seems to be increased validation and test accuracy, it seems to be diminishing towards the end. This error margin between training and testing arises from one of two reasons; the model's bias or variance [58]. The variance is recognized as a margin of error between training and testing, but the testing accuracy seems to increase with increased training samples. The variance can be reduced by increasing the number of training samples, leading to improved accuracy. If the error margin does not reduce as one increases the number of training samples, the model suffers some bias. Bias is the product of erroneous assumptions in the learning algorithm. High bias is the product of an inadequate machine learning model missing relevant relations between features and target outputs (underfitting). From figure 3.5, the error between training and testing seems to result from a combination of bias and variance. The trend is slightly responding to an increasing number of training samples; however, the performance improvement rate is diminishing. 96% detection rate is excellent accuracy, and improving the model or increasing the training samples may only provide an insignificant accuracy boost.

3.1.4 Predictor selection

As was mentioned in section 3.1.2, multiple grids containing ocean variables were stored and used as training data: sea surface temperature, height, eastward velocity, and northward velocity. Until now, these have been the viable candidates for training a machine learning model based on the characteristics they exhibit. A machine learning algorithm can either use one or a combination of the variables as predictors. The initial assumption was that all variables provided some predictive power and would, as an aggregate of variables, further improve the data's predictability. However, after thoroughly testing all the possible combinations of variables, the combination that performed best was the ocean current velocities, i.e., the north- and eastward velocity.

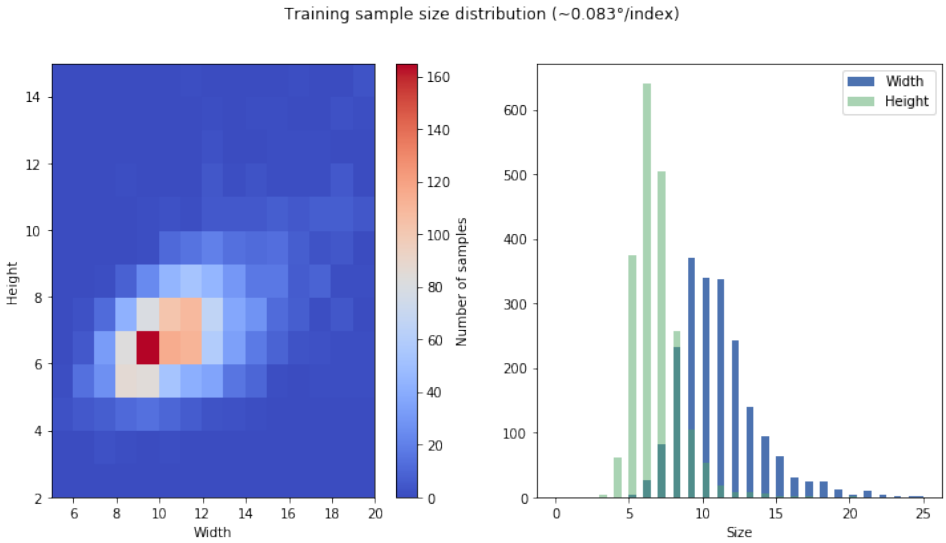


Figure 3.4: The size distribution of the collected training samples. The width and height are given as the number of indices in the grid. The resolution of the data used is 0.083°, which approximates to 9.2 km at a lower latitude projection (close to the equator).

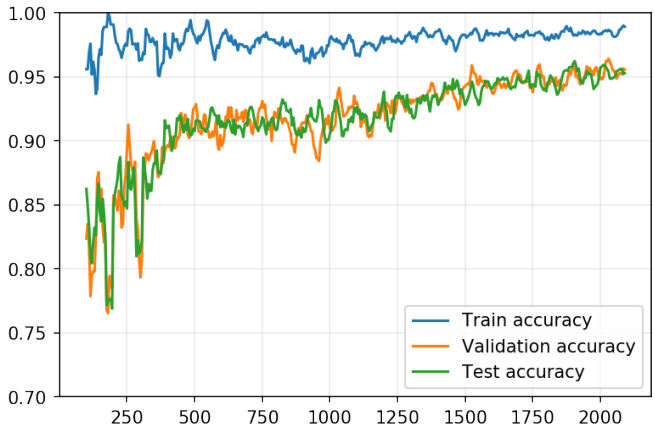


Figure 3.5: The training, validation and testing accuracy when increasing the amount of training data. You can see the accuracies diminishing returns start to kick in as the amount of training data increases

Initial predictor candidates

Figure 3.6 shows two samples of both cyclones and anti-cyclones. The first column of the figure displays the sea surface height, second is sea surface temperature, third is the eastward ocean current velocity, and the fourth is the northward ocean current velocity. As previously stated, the mesoscale eddies should manifest as diverging SSH and SST

gradients with local extrema [59, 60]. Figure 3.6 confirms elevated SSH for the anticyclone samples and lower SSH for the cyclone samples. However, the SST of the samples from figure 3.6 lacks the same distinct gradients towards local maxima or minima. It can be discussed whether the samples were found near the coast or affected by other external disruptions. However, this lack of spatial attributes in the proximity of an eddy was found to be true for most samples, rendering the SST as an inadequate predictor. The two last columns represent the south to north and the west to east velocity magnitude, respectively, and are used to reconstruct the ocean current velocity field. From the eastward velocity (uvel) in the third column and top row of the matrix of figures, we can see that values are going from negative to positive values indicating westward velocities in the bottom part of the grid, and eastward velocities in the top part. The northward velocity (vvel) in the top row indicates northward velocities to the left and southward velocities to the right of the grid. Combining these into a velocity field will create a characteristic anticlockwise rotation for a cyclone in the northern hemisphere, as displayed in figure 3.3. All of the samples gathered so far, showed the same distinct clockwise or anticlockwise rotation of the ocean current velocity field, making it a reliable and attractive candidate as a predictor.

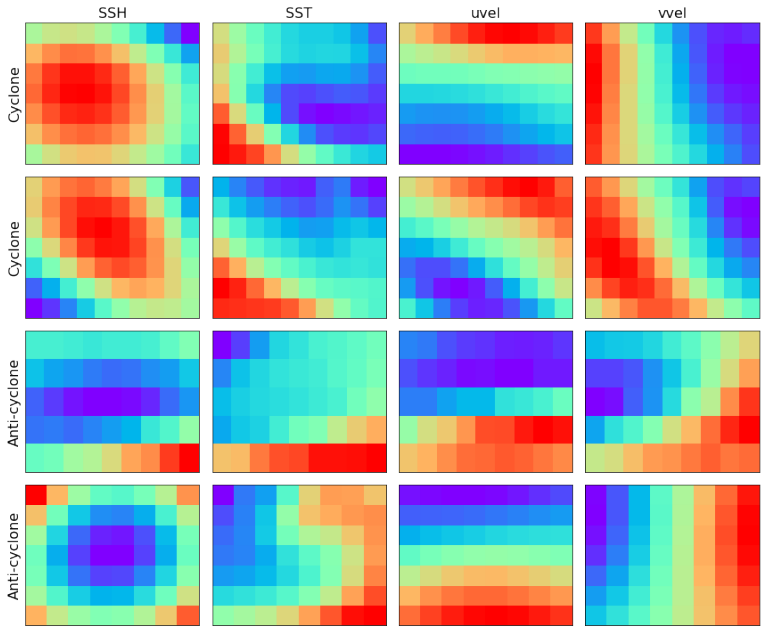


Figure 3.6: Pseudocolor plots of the variables Sea surface height (SSH), sea surface temperature (SST), eastward velocity (uvel), and northward velocity (vvel), stored in a cyclone and anti-cyclone training sample. Lower values are blue, while the higher values are red.

Final combination of predictors to be used

Although the theory supported all the variables' efficacy as predictors, further testing revealed certain combinations of variables to perform better. The combinations of predictors were evaluated using a simple CNN trained on the collected training data. A deeper dive into the machine learning pipeline, such as resizing, scaling, and other feature engineering techniques, will appear in the subsequent sections of this chapter. Because of the non-deterministic nature and variability of training a CNN, the average accuracy of several trained models is used to evaluate the candidate variables.

For the first test, the network was modified with a single-channel input. After the model had been trained and tested 20 times, they gave the average accuracy and standard deviation displayed in the first four columns of table 3.1. The most notable performance is that of the SST, which seems to be performing just slightly better than random guessing when predicting one of the three classes. Hence, the initial test quickly dismissed the use of SST as a predictor. The three remaining variables seemed to give promising results as single-channel inputs and could presumably perform better as an ensemble. Since uvel and vvel together represent the sea surface velocity, it is rather evident that they will perform better as an aggregate. The results of the aggregated variables as multi-channel input are displayed in the two last columns of table 3.1. Although the difference between the two ensembles is minuscule, the higher variance and lower predictive power suggest that excluding the SSH provides a better performing and robust model. In conclusion: the ocean current velocity variables were the optimal predictors for predicting ocean eddies.

Table 3.1: Table showing the accuracy and standard deviation after training a simple CNN 20 times using different combinations of variables as input channels. Top row are the variables used either as a single channel or multi-channel input.

Variable	SSL	SST	uvel	vvel	uvel and vvel	SSL, uvel, and vvel
Accuracy [%]	77.75	36.92	uvel	vvel	95.83	94.25
σ[%]	5.52	1.31	uvel	vvel	0.74	1.03

3.1.5 feature selection and engineering

Two of the most crucial elements of machine learning pipelines or models are feature selection and engineering. The feature engineering and selection process occupy the space between the data and the machine learning model in the machine learning pipeline. Features are numerical representations of the raw data or other features [55]. Feature selection and engineering is the act of using domain knowledge to modify raw data into representations that highlight the underlying problem to the machine learning model, improving the model's accuracy on unseen data.

SVM and Random Forest belongs to a branch of machine learning models for which the user is most of the time required to perform feature engineering and feature selection. On the other hand, conventional CNN produces feature maps by stacking multiple layers of non-linear functions that transform the input into feature maps while training the network. If the features are scarce and irrelevant, the model will struggle to predict the correct outcome. However, the model will likely be slow and tend to overfit if there are too many features.

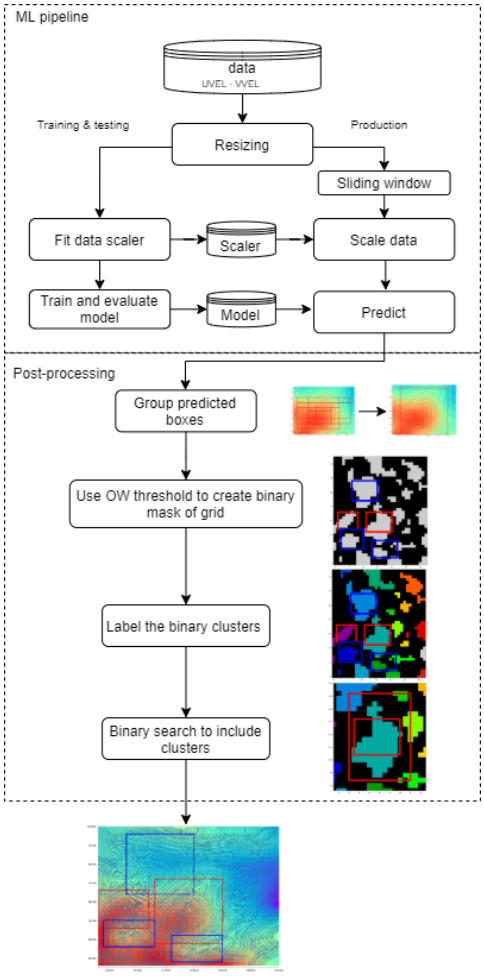


Figure 3.7: The train and prediction pipeline. Whenever a model is trained, the training data from the train-test split

In section 3.1.4, the ocean current velocities were determined to be the subset of the available variables in the training data to contain consistent and predictable features. Dissection of the available variables and choosing the best performing subset is a method of feature selection. The training data could initially use a combination of all variables, from which a smaller subset of variables was found to provide more information and, at the same time, reduce the overall dimensionality. Although multiple feature engineering techniques were tested, only a few simple methods seemed to improve the information gain of the ocean current velocity: resizing and scaling.

In the initial stages of the system's workflow shown in figure 3.7 depicts how the training data is prepared for both training and prediction. The data storage at the top of the diagram represents, in this case, 2045 training samples. Each sample contains both northward and

eastward ocean current velocity, as concluded in the previous section. Because the machine learning algorithms used in this project require a predefined input size, the first step in the machine learning pipeline is to resize the samples to uniform square grids. The size distribution in figure 3.4 suggests that the eddies have a more rectangular grid, which is caused by the CMEMS-phys use a geodetic reference system. The cells of a geodetic reference system extend further in the latitudinal (north-south) direction than in longitudinal (east-west) depending on the latitude of the cells (equal close to the equator and stretches as one move towards the poles). Nevertheless, the samples are resized as equal square grids for pragmatic reasons. The modified VGG16 network was used to provide intuition as to how the model behaves when presented inputs of different sizes. Figure 3.8 displays how the model performs when trained and tested on training data resized to a given square input size. It is important to note that a CNN's performance will behave differently to the input sizes according to its network structure. The test merely provides some insight into the influence of the size of the input. The optimal size for the CNN network used in figure 3.8 seems to be somewhere around 25x25 indices. Most likely, either a shallower or deeper network with more pooling layers (downsampling), and a larger or smaller sample size, could cause the best performing sample shape to shift. There is also a trade-off between the input size and the amount of processing time. However, since the detection of eddies only needs to happen once a day, the processing time is not one of the main concerns.

The next preprocessing step after resizing the variables to uniform and seemingly optimal sizes is another essential feature engineering procedure: feature scaling (feature normalization). Different methods of feature scaling were introduced in section 2.2.8, the two most popular techniques being the *min-max* and *standardization*. If the features' distribution is known and gaussian, the accord is to use normalization. Figure 3.9 shows the distribution when every value of all 2045 samples are put into 100 bins and counted. Because the training data are collected from a large area and within large time-span, the

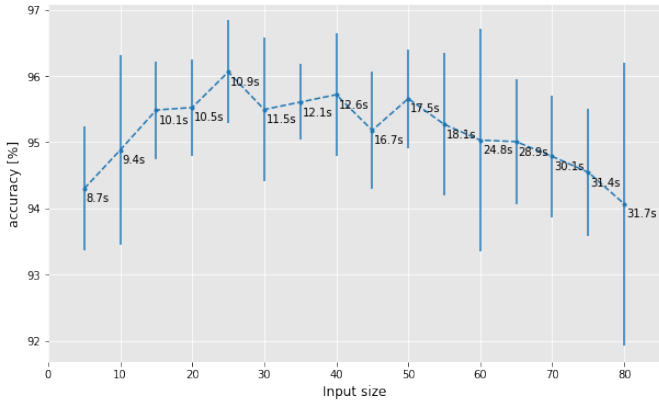


Figure 3.8: Testing different training data input sizes on a 3 blocks deep VGG network. The trend shows how the average accuracy and its standard deviation from training 20 models behaves to different square input sizes. Next to each accuracy is the time it took to train at the given size (with some common run-time overhead).

velocities will follow a gaussian distribution centered around zero. Both scaling techniques were applied and tested, and as expected, the normalization method gave much more promising results. Although multiple feature engineering techniques exist, resizing and scaling are the most essential and were the only ones found to improve the performance of the machine learning models.

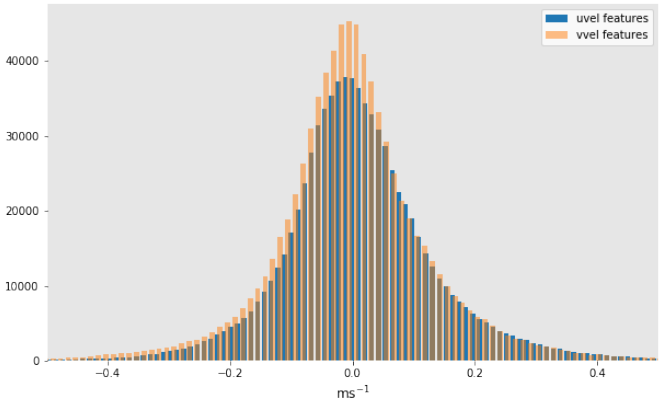


Figure 3.9: A histogram with 100 bins for both the eastward and westward ocean current velocity variables.

3.2 The machine learning algorithms

The predictors and features are now pruned, resized, and scaled to provide the best possible foundation for classification. However, there will never be a ”one fits all” machine learning algorithm because of the no free lunch theorem. There will always be domains in which one algorithm is better than another. Three of the most common supervised learning algorithms were evaluated using pre-processed training data to identify the best performing machine learning algorithm for our training data. The algorithms on trial were SVM, random forest, and CNN. The scoring strategies used to evaluate the algorithms are precision, recall (sensitivity), and f1-score. A small recap of the scoring functions:

Table 3.2 shows how each algorithm performed on the test set after training. The scoring functions provide a measure of how the model performs on a single class. The final row in table 3.2 provides a macro measure, which is an average of the scoring functions over all classes. It is worth noting that the macro measurement for precision is the overall accuracy of the model. The algorithms were tested using a simple train-test split, where a portion of the training data is used as a holdout set used to test the model after training. The holdout set was created using one-third of the training samples, which amounts to 664 samples.

3.2.1 Random forest and SVM

The main difference between random forest and SVM are the number of hyperparameters that needs to be tuned. Random forest only needs to specify the number of decision trees. On

Table 3.2: The scores of different algorithms using the test set of 664 samples with an even distribution of classes. The SVM scores have one less decimal place because of the scoring functions used at the time.

Class	Method	Precision	Recall	F1 score
SVM	Cyclone	0.94	0.98	0.96
	Anti-Cyclone	0.97	0.93	0.95
	Nothing	0.92	0.92	0.92
	Macro	0.94	0.94	0.94
RF	Cyclone	0.96	0.99	0.97
	Anti-Cyclone	0.971	0.964	0.972
	Nothing	0.948	0.942	0.953
	Macro	0.952	0.956	0.958
CNN	Cyclone	0.967	0.991	0.979
	Anti-Cyclone	0.968	0.990	0.981
	Nothing	0.986	0.947	0.966
	Macro	0.975	0.976	0.975

the other hand, SVM has several hyperparameters that need to be tuned to find the decision boundary that maximizes the margin between the classes. Another difference is that random forest is inherently suitable for multi-class problems, while the SVM is intrinsically a one-vs-one problem. Hence, the SVM has to be reduced to multiple classification problems to classify the three intended classes. The processing time of the algorithms is, within reason, of no significant concern for this project, and the performance of the algorithms will be evaluated based on their accuracy. *Grid-searching* strategies were employed to find the best possible combination of hyperparameters for both algorithms. The grid search method trains multiple models based on lists of hyperparameters that an expert suggests, and saves the best performing model for continued use.

By comparing the accuracy (the macro measure of precision) of both the final SVM and random forest models in table 3.2, the random forest outperforms the SVM by about one percentage point. It is hard to provide any specific reason as to why this is the case. One reason could be that the hyperparameter tuning for the SVM is much more complicated, making it harder to find the best performing combination. As the amount of resources allocated for tuning the hyperparameters and testing both SVM and random forest was limited, there might exist a permutation of SVM hyperparameters that outperforms the random forest for our data.

Another possible reason for the difference in performance is the lack of feature engineering for the SVM. Although the standard feature engineering methods such as Histogram of Oriented Gradients (HOG) and Principal Component Analysis (PCA) were tested (citation), they only seemed to reduce the performance of the SVM. Just like there is no free lunch when picking a machine learning algorithm, there exists a feature engineering procedure for the dataset that will improve the SVM's performance.

3.2.2 CNN

CNN outperformed both SVM and random forest in the initial trial from the results in table 3.2, which was the initial assessment of the analysis. It is disputable that a more elaborate testing scheme could have found a better performing algorithm for the dataset. The intrinsic ability of a CNN to transform the input into new and meaningful feature mappings simplifies certain aspects of the machine learning pipeline. The engineer is not required to investigate what he or she believes to be the most informative augmentation of features. In addition to being the more pragmatic approach, CNN was found to have significantly higher accuracy. However, as for all other aspects of machine learning, there is no one network structure fits all. Different schools of network architecture need to be tested to identify the best performing model for the training data.

Three of the most common CNN architectures were introduced in section 2.2.4: VGG (Visual Geometry Group), Inception (GoogLeNet), and ResNet (Residual Neural network). Each of these classic network structures was used as inspiration when creating and testing architectures for the final model. For instance, if one were to follow the VGG network trend, one would use a series of conv-conv-pool layers like the VGG block shown in figure 2.14.

A VGG network architecture was used to complete all tests prior to this final evaluation because of its simplicity and versatility. All three architectures are using modified blocks of fundamental hidden neural network layers. The blocks usually consist of an arrangement of convolutional layers, followed by regularization layers (pooling and dropout). All three

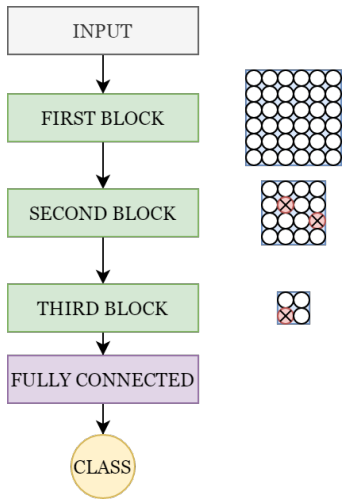


Figure 3.10: The three architecture varieties contain the same structure made up of a sequence of blocks. Each block contains filters with different receptive fields, either in a sequential or parallel manner or employing bypassing techniques. After the convolutional filters, there are layers of regularization, such as a pooling layer followed by a dropout layer, illustrated by reducing the size and the deactivation of neurons. The final layers are either one or more fully connected layers to predict a class from the previous layers.

Table 3.3: Accuracy of three of the most common CNN architectures: VGG(Visual Geometry Group), Inception (GoogLeNet), and ResNet (Residual Neural network).

Network	VGG	Inception	ResNet
Accuracy [%]	97.51	94.47	96.11

designs follow the same design pattern shown in figure 3.10: An input layer given by the geostrophic current vectors, Three modified blocks following different architecture conventions (e.g., VGG, residual network, and parallel), and finally the flattened feature maps are flattened and classified using one or more fully connected layers. All three networks seemed to perform well; however, the more modest VGG network architecture seemed to outperform the other conventions. The VGG network was henceforth used as the model of choice for further evaluation and refinement.

3.2.3 Other notable AI methods tested

A deep learning method that was tested, but omitted from the methods section and is worth mentioning is the Faster R-CNN object detection algorithm [61]. The main difference between object detection and classification is that the detection algorithm tries to draw a bounding box around the objects of interest and locate it in the image. The Faster R-CNN is trained by inputting images along with a parsed document, providing information about the bounding boxes' coordinates for the objects present in that image and its labels. The data collection GUI was slightly altered to fit the mentioned description of annotated data, and a total of 35 grids with a total of 253 eddies were extracted. The only problem was that Faster R-CNN only accepted images using RGB as input channels. Hence, the sea surface height (SSH) was encoded as a 0-255 value and used as the red channel, the eastward velocity (uvel) was encoded as the 0-255 green channel, and northward velocity (vvel) as the 0-255 blue channel. Figure 3.11 shows how the ocean variables make up an image using each variable as an RGB channel.

Figure 3.11 shows how the Faster R-CNN performed after training on 35 images for 20 epochs, gaining a total accuracy of 81.3. Twenty epochs of training are not sufficient for training a Faster R-CNN model; however, the resources needed for training the algorithm

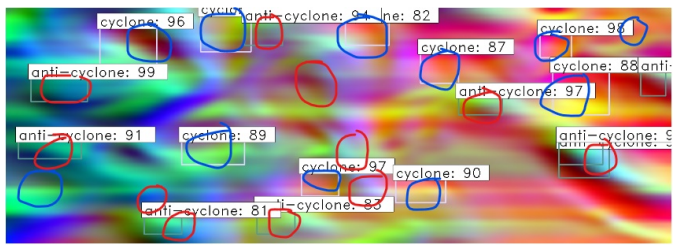


Figure 3.11: Because the original variables were lost, but the text file containing the boundary boxes were in tact, the eddies have been drawn on top of the image. Blue circles are the cyclones and red are the anti-cyclones.

were limited. The results of the Faster R-CNN demonstrated that even though a lot of information was lost when encoding the variables into RGB channels, it did manage to predict eddies to a certain degree. The red circles are the true anti-cyclones and the blue circles are the true cyclones. Although a few discrepancies, the algorithm seemed to correctly predict about 15 out of 22 eddies (about 68% accuracy).

3.3 Detection and post-processing

After the best performing CNN model has been trained and stored, it is ready for predicting ocean eddies on a larger ocean grid of interest. The model's application stage of the system is referred to as "production" in figure 3.7, and follows the path to the right in the pipeline flowchart. The first step is for the model to query multiple sliding windows performed on the grid and store the predictions made by the model. The predictions are then grouped according to how many predictions overlap according to some relative measure. The final predictions are then refined to encompass the full eddy, much like the final output of the flow diagram in figure 3.7.

3.3.1 Eddy detection

Sliding window

The data storage at the top of the flowchart is, in this case, a placeholder for a larger grid of ocean current velocities, like the one shown in figure 3.12. Instead of feeding the system with annotated training data, a scheme of sliding windows will feed the model with subgrids as candidates for prediction. As discussed in section 3.1.3; the height and width of the training samples from figure 3.4 span anywhere between 36 to 184 km. The range in sample sizes suggests that a single-window approach may not be able to capture either the smaller or larger eddies. However, one could also argue that the circulatory motion is constant across the eddy and that a smaller window should capture the eddy by detecting its core. The initial sliding window procedure used a single smaller window, did a satisfactory job of predicting positive observations. However, it also predicted a few false positives, as it was able to pick up minor circulatory sea-surface ocean currents. It is not faulty of the model to predict these smaller pseudo-eddies; nevertheless, it is an undesirable effect. Instead of using a single sliding window, the system employs sliding windows with various windows and step sizes.

To further facilitate the prediction procedure, the system combines the sliding windows with proper grouping techniques to merge the predictions into a further plausible prediction. As mentioned, eddies exhibit robust circulatory features that should be recognizable by two or more of the window sizes, and setting a minimum number of overlapping rectangles as a criterion for grouping could enhance the reliability of the system. Figure 3.12 highlights how sliding windows of varying sizes have moved across a smaller grid of ocean current velocities, and created numerous candidates. The predictions are consistent for all window sizes and accumulate when the presented window contains an eddy.

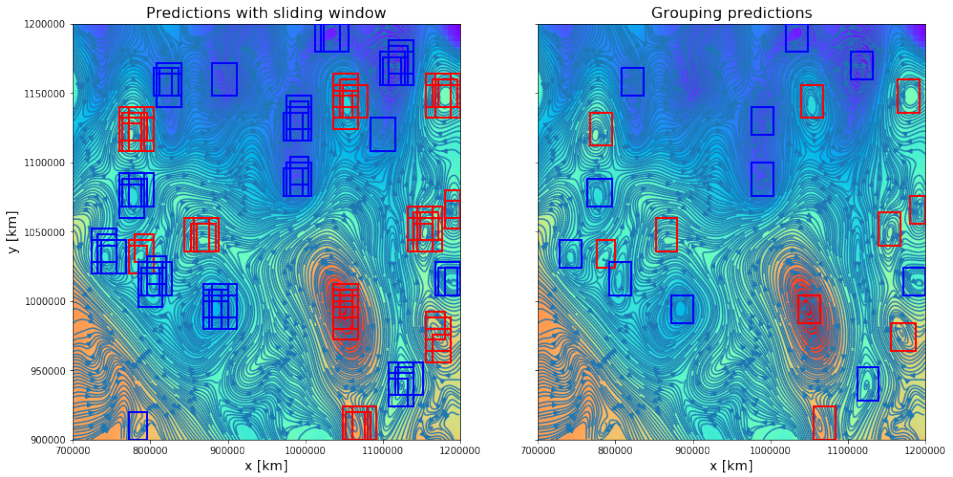


Figure 3.12: Both plots shows a streamline plot of the sea surface current on top of a heatmap of the sea surface height. The left plot shows the successful predictions as a result of running a sequence of sliding windows over the grid. The plot to the right is the remaining predictions after they have been grouped. The blue rectangles are the predicted cyclones with clockwise rotation, and the red rectangles are anti-cyclones with anti-clockwise rotation.

Grouping

After running the sliding windows on the grid of interest, the result is a seemingly messy map of predicted windows, as there are overlaps and, most likely, a few predicted pseudo-eddies. Thus a grouping technique is used as an ensuing refinement process. The grouping is achieved based on two criteria: A relative measure of the rectangles' overlap, and how many rectangles are needed for it to be valid. All the successful predictions after running the sequence of sliding windows are shown in the left plot of figure 3.12. There are a few unaccompanied predictions, which most likely are false positives, or at least less likely to be eddies of interest. The next plot in figure 3.12 shows the predictions that have met the grouping criteria, combining the overlapping predictions of eddies into a much clearer map of predictions, while removing the less likely candidates.

3.3.2 OW threshold clustering

Although the model predictions and the subsequent grouping of predictions have presented a clear set of cyclones and anti-cyclones, the rectangle boundaries do not necessarily incorporate the full size of the eddy, nor is it necessarily centered at the core of the eddy. This section will introduce using the OW parameter and vorticity to enhance the eddy detection system further. The steps of the OW threshold clustering method (finn et bedre navn) are:

1. Calculate the OW parameter from the ocean current velocities.
2. Create a binary mask of the grid cells dominated by relative vorticity (negative OW

values).

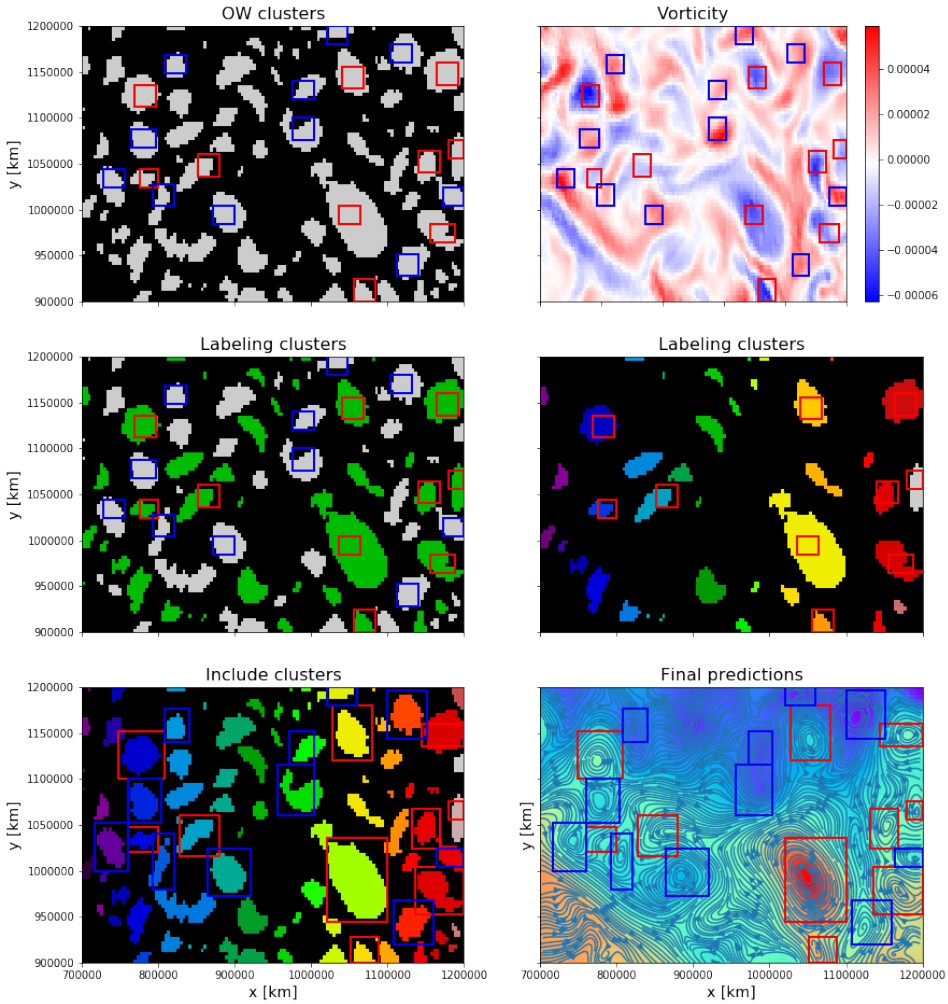


Figure 3.13: The figure shows all the stages of post-processing the predictions to include the entire eddy. In the first plot (top left), the OW values below a certain threshold are masked, forming a binary map of cells dominated by vorticity. The next plot (top right) shows the vorticity ($[s^{-1}]$), which is used to split the labeled clusters into cyclones (positive rotation) and anti-cyclones (negative rotation). In the middle left plot, the binary clusters are divided into cyclones and anti-cyclones according to their rotation's polarity. The next step is labeling each cluster for both cyclones and anti-cyclones. The middle right plot shows uniquely labeled cyclone clusters. The final step in the procedure searches through the labeled cluster that belongs to a given prediction and expands the rectangle until it covers the cluster. In the bottom left plot, both cyclone and anti-cyclone predictions include the full clusters. The last image shows the final predictions using the same heatmap and streamline plot as in figure 3.12

3. Use vorticity to partition the clusters formed by the binary mask into cyclones and anti-cyclones based on positive or negative vorticity.

4. Create an unique label for each cluster in both cyclone and anti-cyclone partitions.

5. A search algorithm finds all cells that belongs to a given eddy.

6. The predicted rectangle is expanded such that it covers the clustered cells of the eddy.

7. The final prediction should cover most of the eddy, giving a more accurate perception of the eddy's characteristics, e.g., size, average current velocity, etc.

OW binary mask

The first part of the post-processing shown in figure 3.7 is calculating the Okubo Weiss (OW) parameter. The procedure is somewhat similar to the concept behind the earlier iterations of the data collection application introduced in section 3.1.2, as we will be using a threshold to discriminate between eddy cells and non-eddy cells. A negative OW value indicates that relative vorticity dominates over the strain in a given cell, particularly in the inner part of the eddy [7, 6] (flytt siteringene til teoribiten). The idea is to use a less strict threshold to produce a binary mask separating the eddy cells from the rest. Although some cells do not belong to an eddy, the cells within and neighboring the prediction most likely does. The upper left plot of figure 3.13 shows the binary mask produced after calculating the OW parameter using the ocean current velocities from figure 3.12 and extracting the mask using an OW threshold of -0.5.

Seperating and labeling clusters

The post-processing's main goal is to expand the smaller rectangle around the eddy core such that it covers the full eddy using the binary mask. The procedure continues to find all neighboring masked cells, giving them a unique label. Each of the predictions will very likely be a part of one of the clusters, as can be seen in the top left plot of figure 3.13. The rectangle is expanded to include the full cluster by performing a search over the labeled cluster associated with a prediction. However, since the OW parameter does not discriminate between negative or positive vorticity, there are cases of cyclones and anti-cyclones merging into one large cluster. To prevent the inclusion of both types of eddies as the same prediction, the polarity of the vorticity is used to separate the clusters into cyclones and anti-cyclones. The vorticity, in this case, is the curl of the sea surface current velocity field, describing the local spinning motion of a parcel, or in our case, a two-dimensional. The top right plot shows the grid's vorticity, which is used to separate the clusters based on each cell's spin shown in the middle left plot.

Once the clusters are separated and labeled, a search algorithm are used to find all cells that belong to a given cyclone or anti-cyclone. The algorithm continuously expands the edges of the initially predicted rectangle as it finds cells belonging to its cluster until the new rectangle covers the full cluster. The middle right and bottom left plot in figure 3.13 shows the progression of the post-processing from a small rectangle with an eddy core within its bounds to it covering the full eddy cluster. By comparing the grouped predictions in

figure 3.12 with the final plot in figure 3.13, one can see how the post-processing procedure covers a much larger and more accurate portion of the eddies.

3.4 Converting datasets to a comparable basis

Once the predictions are grouped and refined, we have a fitting prediction and boundary box for the eddies in the given grid. An important outcome of the refinement process is a more accurate perception of the eddy's properties, such as size, velocity, and location. Because the datasets use different projections, spatial units, and resolution, they need to be transformed to a common and comparable footing.

So far, only the CMEMS-phys dataset has been used to showcase the application for extracting training data, the machine learning models, the predictions, and the post-processing. However, the primary motivation is not being able to detect eddies using the open-access assimilated data that CMEMS provides, but creating a system that can potentially improve SINMOD's numerical simulations by producing information about the eddy's position and characteristics. Besides being able to interface with both CMEMS-phys and SINMOD, the system needs to work on the in situ observations from the CMEMS-multiobs dataset.

The foremost challenge is that the datasets use different projections, resolutions, and grid coordinates. The SINMOD model uses cartesian coordinate x-y coordinates on a polar stereographic projection, while the other CMEMS models use geodic lon-lat on a regular grid (Get citation or reference theory section). Because the grid of interest is located near the north pole (arctic ocean), the CMEMS datasets' grid can be somewhat distorted as more of the ocean's surface falls into each cell. This effect gets more critical as the cells get closer to the north or south pole. On the other hand, the stereographic projection projects the regular spherical grid onto a plane with a one-to-one mapping betwen the ocean's surface and the grid. For this reason, the CMEMS dataset coordinates are converted to the same polar stereographic projection used by the SINMOD model.

Chapter 4

Results & Discussion

So far, a system capable of detecting ocean eddies have been proposed. The variables that, from a theoretical perspective, held relevant features and attributes related to an ocean eddy, were evaluated as candidates for training a machine learning model. The final set of variables that contained the most contextual information with regards to an eddy seemed to be the geostrophic velocities given by eastward and northward ocean current velocity. After predicting the eddies of a specific geostrophic velocity map using multiple sliding windows, the predictions were grouped to promote the most likely eddies by a measure of the density of predictions. The predictions were further refined by using supported measures of deformation and rotation (The relative vorticity and OW parameter), to provide a more segmented representation of the predictions, providing a more accurate presentation of the size and location of the eddy core.

4.1 Training data

4.1.1 Data collection

One of the significant problems proposed by this project has been the collection of training data. After a somewhat comprehensive study of existing data annotation methods, there was not anything applicable to the domain of two-dimensional grid variables, let alone marine and ocean data. Existing annotation tools are predominantly for inserting images and providing labels to an image's segmentations or boundary boxes.

The final image annotation tool, or application, created in MATLAB, was displayed in section 3.1.2 and figure 3.2. For the most part, the application was a temporary and modest tool for generating a sufficient amount of training data to start training and testing different machine learning algorithms and structures. The initial design was a Python application, but without previous experience in developing graphical interfaces and because MATLAB practically provides out-of-the-box graphical interfaces with straightforward integration of interactive buttons and other accessories for handling data, made MATLAB the most appealing alternative.

The number of samples extracted per hour marking, converting, and downloading two-dimensional fields of SSH, SST, and geostrophic currents to a compressed folder were about 400 samples pr. hour, which is a decent pace considering the substantial amount of data needed for training a machine learning algorithm, especially a deep neural network.

The cyclones and anti-cyclones that was stored marked and stored as training data were, for the most part, identified by looking at the SSH and ocean current flow field. The SST were only attached to the training data without genuinely examining the SST data for eddy features. It is arguable that by only looking at the sea surface current or any single layer for that matter, could deceive the user in some cases, and some other event caused the seemingly desired features. However, in practically every case, the exposed surface features found when visualizing is the trace of a real ocean eddy.

The model will most likely be able to infer samples containing an eddy when trained on a smaller and not so diverse set of training samples, as exhibit low-level spatial features such as diverging SSH or SST gradients with local extrema and distinct circulatory sea surface currents. On the other hand, the background class has to be learned from a much more diverse and complex combination of data. If the training samples classified as background are insufficiently diverse, the model will have increased uncertainty, which could, in some cases, affect the model's ability to distinguish the background from the two classifications. The model's performance seemed to increase steadily with the expansion of the training data set. However, the background data set will never be perfect, as new situations that are harder to recognize will continue to appear. Something along the lines of the reinforcement procedure introduced in section 3.1.3 could help pinpoint the weak points of the model and provide a more diverse set of samples.

4.1.2 The variable selection

As shown in section 3.1.3 and explained in section 2.1.4, the evaluated candidate variables for use as predictors in a machine learning model were SST, SSH, and the northward and eastward velocity vectors. There probably exists some other essential ocean variable that contains more informative features related to an ocean eddy. However, to the extent of this project's literature review and theory evaluation, these three were the most attractive candidates.

The initial belief was that the CMEMS-multiobs contained both remotely and in-situ observations only at the upper levels of the sea surface. However, all three datasets used, both observed and assimilated data, contained three-dimensional fields, which could have been utilized to improve the system's performance. There are uncertainties attached to only using the sea surface variables, as the geostrophic sea-surface currents could exhibit a circulatory behavior and illuding the system into concluding that an ocean eddy is present. The circulatory motion from the eddy persists through multiple layers of ocean current data, and it would most likely be beneficial to reinforce the algorithm by appending at least the upper layers of the sea-surface as channels for the training data.

4.2 Final Machine Learning algorithm

4.2.1 Machine learning algorithm selection

The SVM and random forest depend more on spatial consistency of features, i.e., the location of the informative features, such as the eddy's position in the two-dimensional grid. In contrast, CNN employs pooling layers that allow for spatial displacement of the object of interest. Not having to consider the object placement in the grid is a significant advocate for using a deep neural network.

Table 4.1: Repeat of the accuracies, or macro precision, given in table 3.2 from testing (using the test set) machine learning algorithms.

Algorithm	SVM	Random Forest	CNN
Accuracy [%]	94.0	95.2	97.5

From the final results are reiterated in table 4.1, it was a little surprising to see random forest performing slightly above the 95 percentile. The SVM had a more modest performance and was initially considered as a fitting contender to the CNN. The SVM is particularly vulnerable to variations in object placements when training the algorithm, and the training data were collected by creating boundary boxes with the eddy in different locations (i.e., not always centered). Another argument for why the SVM gave worse results is its dependence on proper feature engineering prior to training. Other than the feature scaling, neither random forest nor CNN need any feature scaling. The only feature engineering method examined for training and testing the SVM was the HoG (histogram of intensities), which only worsened the performance. The HoG is a feature descriptor mostly used in image processing for object detection, distinguishing the object's features from the background. Considering feature extraction methods such as HoG are intended for object recognition in images, it comes as no surprise that it lowered performance [62]. In the end, the results of the analysis found clear support towards CNN as the better machine learning technique for predicting eddies, given the geostrophic ocean currents as input vectors.

4.2.2 CNN architecture selection

A brief survey of three different architectures was performed in section **??**, for which the accuracy for each network structure, given in table 3.1, revealed the VGG network structure to be the best performing. Because of little experience concerning machine learning projects, the initial plan was to use CNN with very conventional architecture, i.e., the VGG16 structure. The other methods and architectures became appealing contenders after searching through the state of the art of AI.

Although the architectures had different structures in their blocks or modules introduced in the theory section 2.2.5, they still had the same three-block structure with the same regularization layers and fully connected layer at the end. After the final networks were created, they performed very well out of the box, with only slightly inferior results than the final model. The CNN that was shown in figure 2.14 in section theory section 2.2.4, gives a somewhat realistic representation of the final VGG network, only with one extra fully connected layer.

The answer to why any of the three architectures are better than the other is not trivial and cannot be fully covered in this report. Network architecture design is a complicated process. It demands a lot of trial and error mixed with previous experiences to find the model that best fits the specific structure or domain of the classification problem. When trying to find the best possible architecture, the CNN's employed were initially copies of popular architectures created for image classification with a specific size. After experimenting with different network structures and depths, the block-structured architectures introduced in section 3.2.2 were found to be a less complicated and adequate fit. Most likely, there is an architecture that fits better for this type of problem.

4.3 Analyzing final model's performance

This section will provide a review of the final model, and its performance on not just the CMEMS-phys assimilated model data, but also the CMEMS-multiobs containing variables observed by remote sensing instruments, and variables produced by the SINMOD model. First, a quick review of the model on the CMEMS-phys over a couple of days, to not only provide a spatial but also a temporal conception of the model's performance. The second part will evaluate the model's performance across the datasets in three different areas, with a presentation of the model's predictions accompanied by the system's post-processing and the area's bathymetry. The third and last evaluation will present the results of running a comparison of the model's performance on all three datasets over a full year, generating predictions and census such as the number of eddies per day, as well as anayzing their correlation.

4.3.1 Analyzing performance over a few days

The first analysis of the model will be a quick comparison of the model's performance on the CMEMS-phys and SINMOD over a couple of days (the CMEMS-phys is referred to as "satellite" in this figure). The intention is to look at both the horizontal correspondence in variables as the model's behavior over a smaller period.

Figure 4.1 contains the predictions made for the datasets in a smaller area that sits between Greenland, Iceland, and Norway. The system has predicted eddies over six days (the CMEMS-phys contains daily mean fields, and the SINMOD data have been averaged from hourly data to a 24-hour mean), from November 30, 2016, to December 05, 2016. A more extensive examination of the model's precision will be performed in the subsequent section, as the CNN used for this analysis was an earlier and inadequate model.

The first noticeable difference between the datasets is the crater of lower SSH present in the upper left corner of the SINMOD data. It seems to arise from the sharp increase in depth and the concave ridge forming at the upper-left edge, making the south-westward ocean current curl into a cyclonic anti-clockwise rotation. There are striking differences in how the CMEMS-phys and SINMOD models the geostrophic currents and SSH are modeled. Both datasets have a few consistent eddies throughout the analysis: The bottom anti-cyclone (red) for the CMEMS-phys predictions seems to persist (for the most part given the number 1. and sometimes 2.), and the upper right cyclone (blue) and anti-cyclone (red) in the middle. One could suspect that top right cyclone that sometimes appear in the

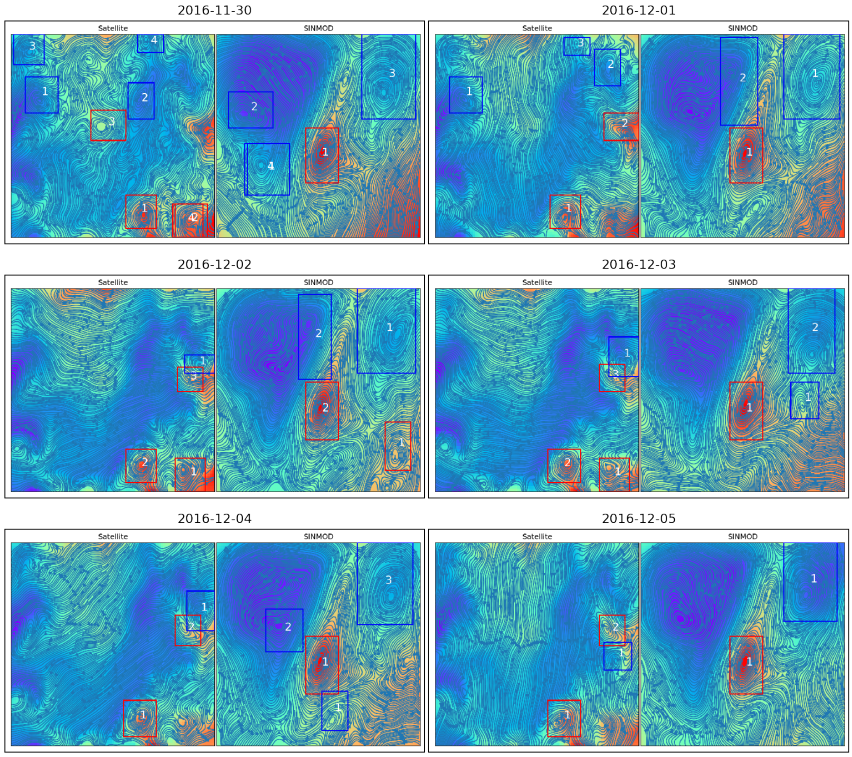


Figure 4.1: Six days worth of predictions somewhere in the intersection between the Greenland Sea and the Norwegian Sea. (skal forklare litt videre) the CMEMS-phys is referred to as "satellite" in this figure

upper right corner of the CMEMS-phys predictions and the one persisting at the same place for the SINMOD predictions are the same.

The final conclusion of this brief analysis, without looking at the system's performance, is that the CMEMS-phys and the SINMOD datasets have notably different variables, and there does not seem to be any strong similarities between the occurrence of eddies at this particular area.

4.3.2 Model's performance on all three datasets

The final CNN model will be applied to the three datasets (CMEMS-phys, CMEMS-multiobs, and SINMOD) mentioned in section 3.1.1. It will show the bathymetry, the predictions on top of the Okubo Weiss binary mask used for post-processing, and the predictions on top of a plot with a heatmap of the SSH and a streamplot map of the ocean current velocities. Three different areas surrounding the Norwegian shoreline with a polar stereographic projection, like the one explained in section 3.4, have been selected for analysis:

- Skagerrak: The rectangular arm of the North Sea, trending southwest to northeast between Norway and Denmark.
- The Norwegian Sea: Just outside the west coast of Norway.
- The Baltic Sea: Near the northern coast of Norway.

Each area will be analyzed three larger figures for each of the areas given int the list above. There will also be a plot showing a contour map of the bathymetry of a larger grid in the area, just to get some more insight into where on the map the area is located, the red rectangle denotes the area in which the eddy prediction system is applied.

Each row shows the output of the prediciton system when the sea surface current velocities from the different datasets are used as input. Each row shows a plot of the labeled OW masks to the left (explained in section 3.3.2). The right image shows a plot of both SSH and ocean current velocity, the SSH is indicated using a colormap with a more red color indicating elevated SSH and blue indicating a lower SSH, the ocean current velocity is shown using a streamplot where the density of lines indicates higher velocities and the arrows show the direction. First row shows the predictions on the CMEMS-phys assimilated data from the Copernicus service, second row on the CMEMS-multiobs remotely and in-situ observations from the copernicus service, and the last row on the SINMOD model data.

The Norwegian Sea

The first area in the middle of the Norwegian Sea, just outside the west coast of Norway. The white area at the bottom of figure 4.2 indicates the landmass of the west coast of Norway. The contour map displays how the depth increase towards the center of the Norwegian Sea. Figure 4.3 showcase the results of applying the proposed eddy prediction system on each of the datasets within the area of interest marked by the red rectangle in figure 4.2.

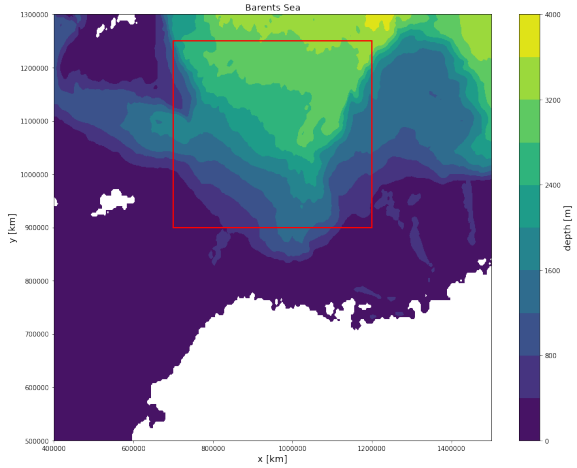


Figure 4.2: Contour map of the bathymetry of a larger grid in the Norwegian Sea along the west coast of Norway. The red rectangle indicates the area for which the prediction system is applied.

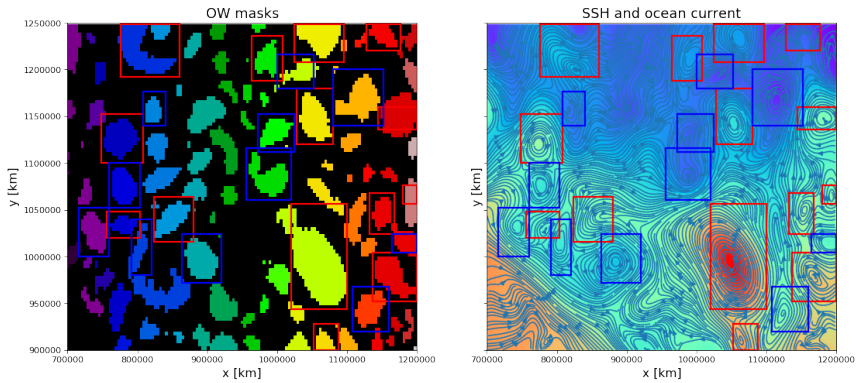
The overall finding was that the system has successfully detected all eddies present in the grids. After examining all three plots showing SSH and the sea surface currents, there are no undetected eddies. There can be eddies whose circulatory behavior is non-detectable on the sea surface current, but this is very unlikely. Another promising finding was that there is only one possible false-positive detected in all three datasets, the top right anti-cyclone (red rectangle) in the CMEMS-phys (top row) dataset. After looking closer at its streamplot, it does not seem to be fully circulatory. Although this might be because the streamplot is limited to the more strong velocities and the magnitude of the velocities making an enclosed circulatory flow is probably small. The system performed very well on the datasets in this area, showing good results.

The first impression after a closer inspection of the predictions from figure 4.3 was the similarities between the predictions from the CMEMS-phys model data (1. row) and CMEMS-multiobs observational (2. row) data. The larger anti-cyclone (red rectangle) to the bottom right seems to be virtually the same size, and its cores are at the same location in both plots. By looking at the bathymetry from figure 4.2, the eddy seems to be located just to the right in the concave depth formation to the bottom right of the red rectangle. Most likely, the reason for the eddy appearing in the same location is the Norwegian ocean current coming in from the left towards the right of the area, and hitting the upper ridge of the bathymetry formation, forcing the current to move down along the curved formation creating a clockwise rotation, i.e., an anti-cyclone. Studies have shown that the flow over the steepest part of the continental slope (known as the Lofoten Basin) contains a vigorous mesoscale eddy field around [63]. The formation of eddies can be highly dependent on the flow of fluid due to abrupt changes or other abnormalities in the bathymetry. The formation of cyclones (blue rectangles) along the left side of the same bottom concave ridge appears to be the same for all three datasets seems to reinforce this statement. The SINMOD ocean current and SSH seems to be much more dependent on the bathymetry than the data from the CMEMS datasets.

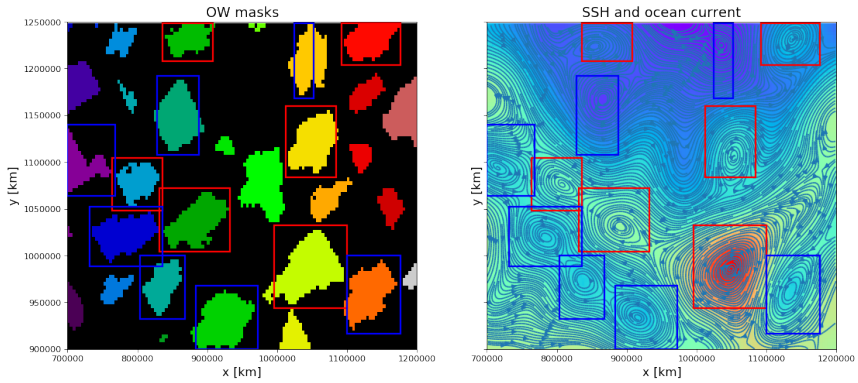
Another striking detail is that the CMEMS-phys seems to have a few more eddies than the CMEMS-multiobs, and the SINMOD model seems to have a few less. It is important to remember that the CMEMS-multiobs data are variables derived from remote and in-situ measurements and, hence, our most true variables[1].

The post-processing seems to be performing adequately. Most likely, the initial prediction system using sliding windows have made a lot more predictions, but the grouping method seems to have been handling it entirely. There is one spottable fallacy in the upper right cyclone (blue rectangle) in the CMEMS-multiobs dataset. The system is supposed to include most of the binary mask. However, the search algorithm seems to have stopped short, making the boundaries more narrow than they should be. This is most likely because the search algorithm hit the upper limit of the grid and stopped short, which is not a significant bug; however, it is a flaw.

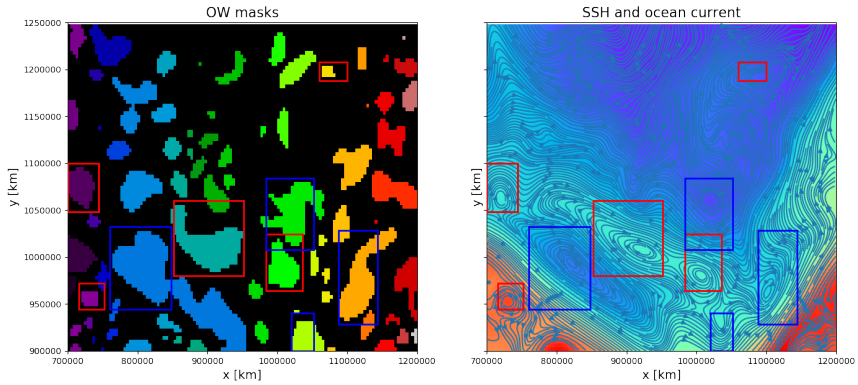
[1] The multiobs data is sampled every seven days, i.e., there are some inconsistencies.



(a) CMEMS-phys



(b) CMEMS-multiobs



(c) SINMOD

Figure 4.3: Each row shows the output of the prediciton system when the sea surface current velocities from the different datasets in the Norwegian Sea are used as input.Left image shows predictions on top of labeled OW masks, the right shows predictions on top of SSH and geostrophic currents. Red rectangles are predicted anti-cyclones and blue rectangles are predicted cyclones.

4.3.3 Skagerrak

The second area to be analyzed is Skagerrak, which is the the rectangular arm of the North Sea, trending southwest to northeast between Norway and Denmark. The area and the system will be dissected the same way as for the Norwegian Sea. Figure 4.2 shows how there is a steep pit between Norway and Denmark; hence the first thought is that there very likely is an eddy circulating this region. Whether it is a cyclonic or anti-cyclonic eddy, it depends on the source of the most influential current, which can either be the current from the Norwegian Sea, the Baltic Ocean, or the mainland.

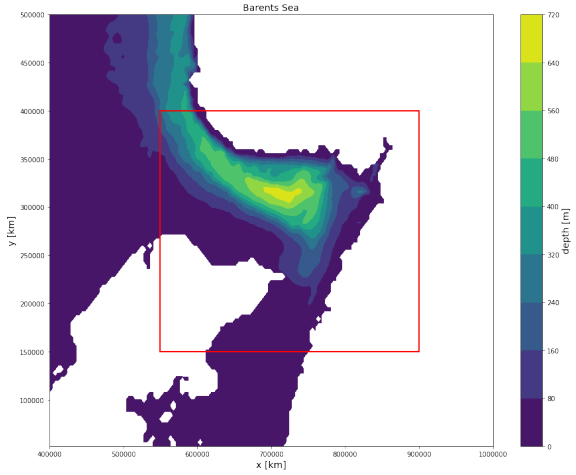
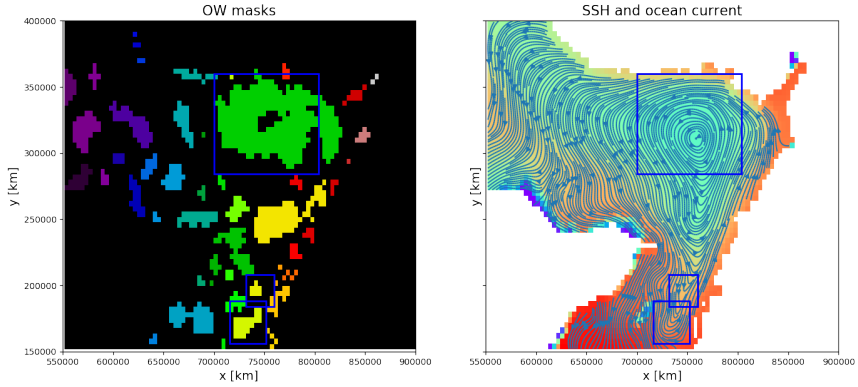


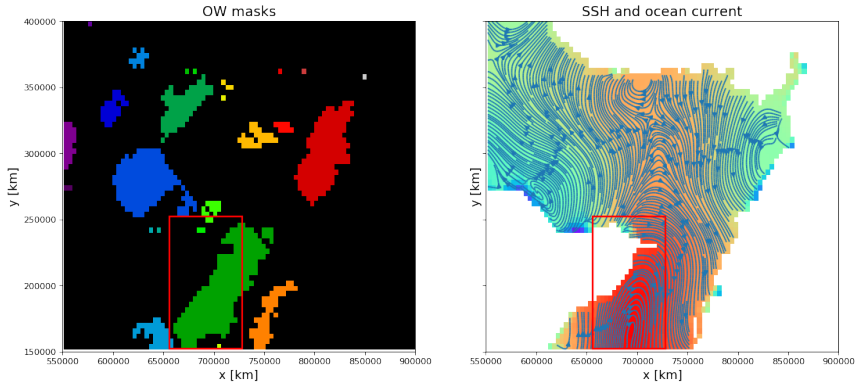
Figure 4.4: Contour map of the bathymetry of a larger grid in Skagerrak, the rectangular arm of the North Sea, trending southwest to northeast between Norway and Denmark. The red rectangle indicates the area for which the prediction system is applied.

The most unexpected sighting in figure 4.5 is the SSH and geostrophic currents of the CMEMS-multiobs dataset. There seem to be many contrarieties in the flow field: there is no circulatory behavior even close to the steep cavity of the bathymetry, and the majority of ocean currents seem to be between landmasses instead of along the sea. There might be some fallacies of the CMEMS-multiobs dataset in certain areas, for which this area most likely is one of the more critical.

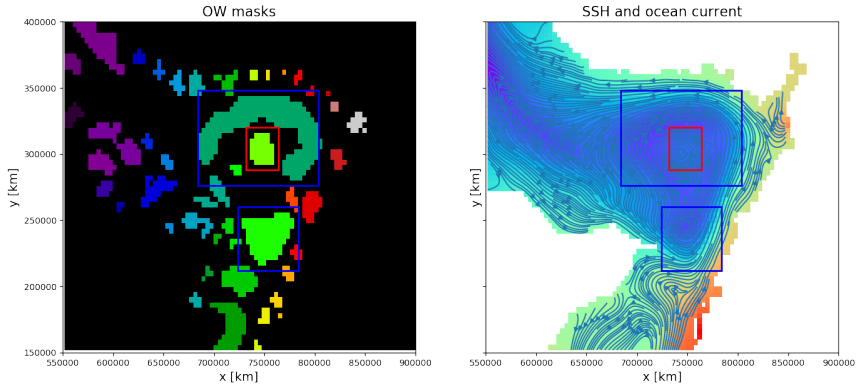
Except for the CMEMS-multiobs data, both the CMEMS-phys and the SINMOD datasets seem to successfully predict a large cyclone in the middle of the bay. The SINMOD dataset also seems to contain a smaller anti-cyclone at the center of the larger cyclone. The reason is unclear, but most likely the SINMOD allows for the bathymetry to produce a smaller cyclone within the larger cyclone.



(a) CMEMS-phys



(b) CMEMS-multiobs



(c) SINMOD

Figure 4.5: Each row shows the output of the prediciton system when the sea surface current velocities from the different datasets in Skagerrak are used as input. Left image shows predictions on top of labeled OW masks, the right shows predictions on top of SSH and geostrophic currents. Red rectangles are predicted anti-cyclones and blue rectangles are predicted cyclones.

The overall predictions seem to be correct, although the bottom predictions of the CMEMS-phys dataset unveil a problem discussed in section 3.3.1, which brought up the problem concerning predicting pseudo-eddies. The pseudo-eddy term means eddies that might either have non-significant circulatory behaviors or contain arching ocean currents, which are essentially half the eddy's circulatory feature. For the most part, this is an error originating from the model's inability to infer the background class on samples such as the ones wrongfully predicted. The training data most likely lacks background samples that represent similar situations, mistakenly increasing the model's confidence that those predictions are correct.

Barents Sea

The third and last area to be analyzed in this section is located in the Barents Sea close to the northern coast of Norway. Figure 4.6 indicates that there are no sharp differences in depth, and the overall depth of the area is much smaller than for the area used in the analysis of the Norwegian Sea in figure 4.2. Ergo, eddies occurring from uneven bathymetry are not to be expected.

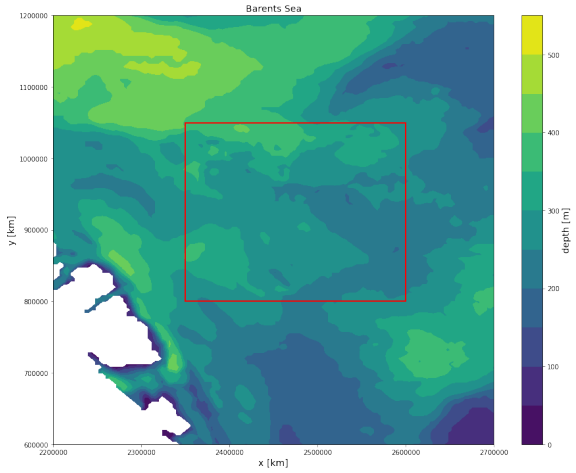
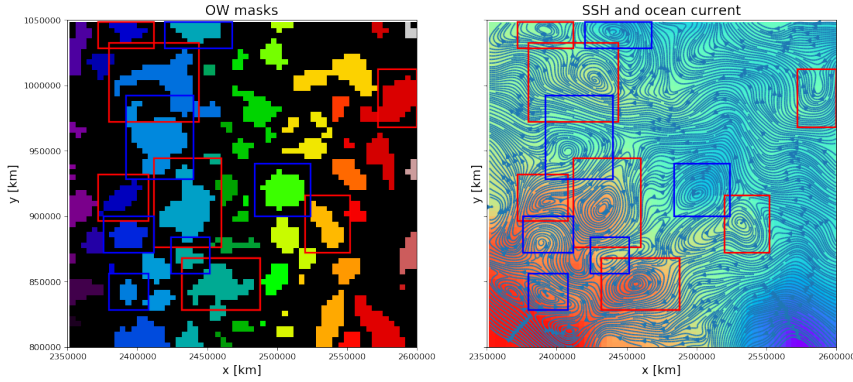
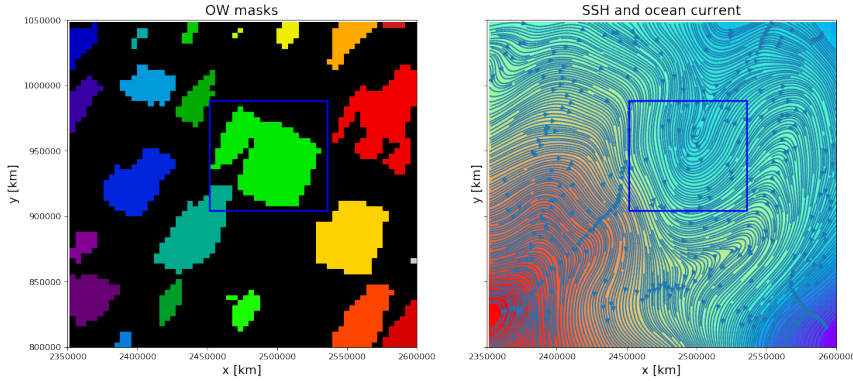


Figure 4.6: Contour map of the bathymetry of a larger grid in the Barents Sea, close to Norway (the patch of land in white is the northern coast of Norway). The red rectangle indicates the area for which the prediction system is applied.

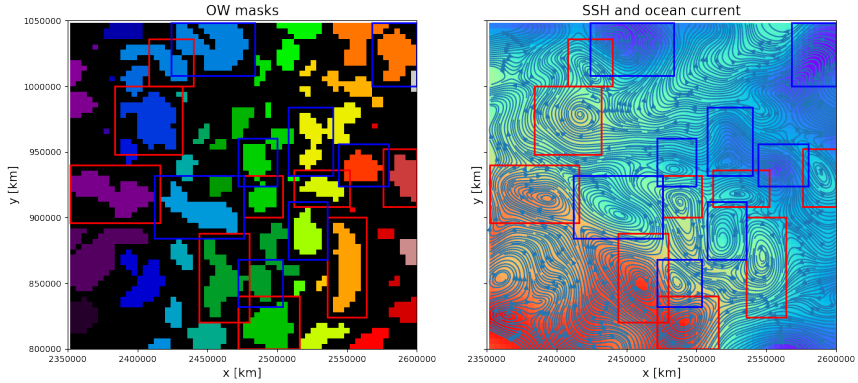
The CMEMS-multiobs data in figure 4.7 b) only seems to contain one minor cyclone connected to a larger meandering ocean current. The smaller enclosed streamline at the center tells us that there is a circulatory motion, although a minor one compared to the more distinct ocean current moving across the grid. The image pinpoints another minor fallacy of the detection system: the Okubo Weiss parameter is susceptible to those powerful meandering and horizontally oscillatory ocean currents. Some ocean currents are dominated by relative vorticity, even though they do not belong to an eddy.



(a) CMEMS-phys



(b) CMEMS-multiobs



(c) SINMOD

Figure 4.7: Each row shows the output of the prediciton system when the sea surface current velocities from the different datasets in the Barents Sea close to the northern coast of Norway are used as input. Left image shows predictions on top of labeled OW masks, the right shows predictions on top of SSH and geostrophic currents. Red rectangles are predicted anti-cyclones and blue rectangles are predicted cyclones.

The two other datasets, i.e., a) and c) in figure 4.7, appear to have a larger collection of eddies present in the grid. There are a few similarities in position and the type of eddy, but the overall correlation seems to be very low.

By looking at the CMEMS-phys dataset in a), the algorithm seems to find most eddies, the only false negatives seem to be two anti-cyclones in the bottom left corner; one to the left of the detected cyclone at x=850000 and y=2400000, and one underneath (the clockwise motion of the eddies is noticeable by zooming in).

The same seems to apply for the SINMOD dataset in c) – all eddies except one cyclone at x=850000 and y=2360000 and anti-cyclone at x=820000 and y=2360000. The false-negative (positives not detected) in both a) and c) seem to be close to the edge of the grid, supporting the assumption that the step of the windows will not enable the system to generate enough predictions for the grouping mechanism, making it unable to recognize some of the eddies at the edges of the grid. However, the bottom left corner does not seem to have any OW-parameter values above the threshold, which means that either the area is a simple background field in OW terms ($W \approx 0$) or dominated by strain ($W > 0$). The terminology of vorticity, background, or strain dominated areas was introduced in the theory section about Okubo Weiss in section 2.1.6.

An additional error is the predicted cyclone to the top right of the SINMOD predictions in a). There seems to be a cyclone close to its bottom left corner (hard to tell on the streamlines in areas dominated by lower SSH), which means that some of the contributing ungrouped predictions could have originated there and then merged. Nonetheless, it is a misclassification.

4.3.4 Exstensive spatial and temporal analysis

The final part of the results includes a full-year analysis of a larger area of ocean. The primary purpose is to research the correlation between the number of eddies detected across the three datasets. To reemphasize the motivation of this report: it is in the interest of the SINMOD model to be able to localize mesoscale eddies, as they have a significant impact on the general dynamics of the ocean as well as the distribution of biomass. The previous sections have proven that the CNN model has given satisfactory detection accuracies across the datasets. However, the datasets do not seem to produce cyclones and anticyclones in the same locations, except when they associated with the formation and slope of the bathymetry. The full-year analysis of the number of eddies can provide a reasonableness check for both the datasets and the model. The reason being that the number of eddies should follow a seasonal trend. Section 2.1.4 gave a short explanation that there is more activity in the number of active eddies during winter [34]. Also, it is conceivable that the amount of eddies for each day is the same across datasets even though there is a mismatch in their location. The area used for this analysis is the Norwegian and Greenland sea, off the coast of Greenland and Norway. The land mass and contour of the bathymetry is shown in figure 4.10, in which a cyan rectangle marks the analyzed area. It is important to note that there is a smaller white spot in the middle of the images, which is the Jan Mayen Island.

Figure 4.9 shows that the occurrence of eddies for all three datasets seems to follow the seasonal activity. The number of eddies is more than doubled during late autumn and early winter for both assimilated data models (CMEMS-phys and SINMOD). CMEMS-multiobs, however, seem to follow a similar trend only with a much smaller variation between the

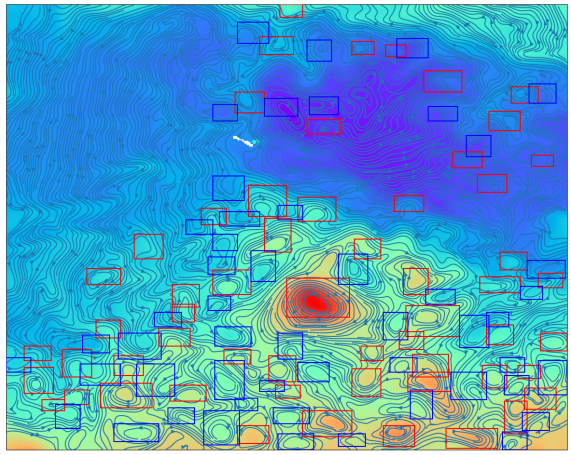


Figure 4.8: An example of a predicted grid made by the comparison system on CMEMS-phys.

seasons. Arguably the figure shows the filtered eddy counts, and the raw data had a much more stochastic day-to-day fluctuation in the number of eddies. However, for this analysis we were interested in seeing the correlation between the datasets and the seasonal activity. By establishing the model predicts eddies with a seasonal trend according to theory, we demonstrate the validity of the model. The model seems to predict more eddies in the CMEMS-phys dataset as opposed to the other two, which was somewhat visible in the analysis from the previous chapter. Most likely, the CMEMS-phys modeling use configurations that allow for a more turbulent ocean, creating more eddies. However, the mismatch in the number of eddies detected could also be a product of over-fitting toward the CMEMS-phys datasets, as they were used as training data.

Figure 4.10 to 4.12 contains visualizations of how the eddies over the full 15 months are distributed within the area. The distribution heatmap has been filtered using a Gaussian kernel to better show where the majority of the eddies appear. The steep bathymetry formation found in the Lofoten Basin in the bottom central part of the cyan rectangle is the same as used in Norwegian Sea analysis from the previous chapter (figure 4.3). As discussed, there should be increased eddy activity in this area; however, the SINMOD model does not seem to show the same activity as both CMEMS-phys and CMEMS-multiobs. It is most likely because of the modeled eddy viscosity in SINMOD, which could contribute to the smooth velocity fields and affect the occurrence of eddies depending on how certain parameters are tuned. Both CMEMS-phys and SINMOD appears to have higher numbers of eddies along the raised area to the right of the Jan Mayen Islands; however, the CMEMS-obs distribution does not show the same results. Concluding remarks for the distribution plots is that the CMEMS-phys and CMEMS-multiobs datasets have very similar eddy distributions during the 15 months, and the SINMOD dataset has a lower eddy activity at less correlated locations.

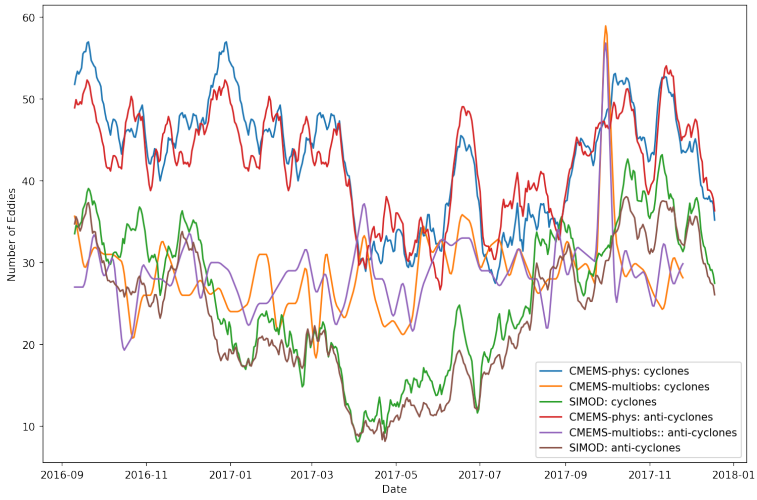


Figure 4.9: The plot shows number of cyclones and anti-cyclones detected for every day during the 15 months for all three dataset. A symmetric FIR filter using 11 coefficients have been applied to smooth out some of the noisy variations to depict the more seasonal trend.

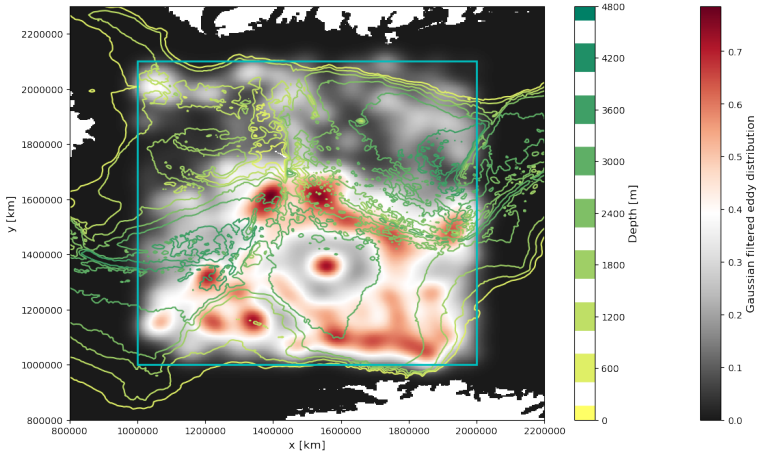


Figure 4.10: A heatmap showing the distribution of where eddies are located during the full year analysis for CMEMS-phys. The heatmap is filtered using Gaussian kernel with a relatively high standard deviation to spread the signal making it easier to visualize accumulations. The kernel used a standard deviation of 8 to properly show the distribution of the detected eddies. The system was applied to the area within the cyan rectangle. The white cells above are landmass that belongs to Greenland, and at the bottom is Norway's.

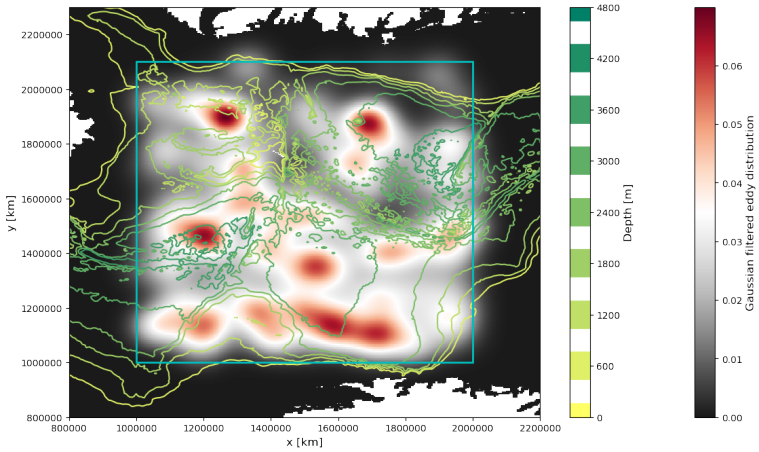


Figure 4.11: A heatmap showing the distribution of where eddies are located during the full year analysis for CMEMS-multiobs. The heatmap is filtered using Gaussian kernel with a relatively high standard deviation to spread the signal making it easier to visualize accumulations. The kernel used a standard deviation of 12, which is higher than for the CMEMS-phys dataset because it found higher numbers of eddies. The system was applied to the area within the cyan rectangle. The white cells above are landmass that belongs to Greenland, and at the bottom is Norway's.

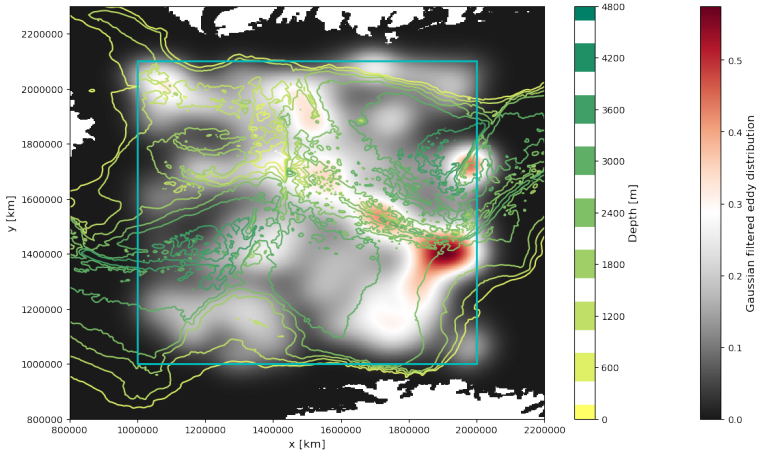


Figure 4.12: A heatmap showing the distribution of where eddies are located during the full year analysis for SINMOD. The heatmap is filtered using Gaussian kernel with a relatively high standard deviation to spread the signal making it easier to visualize accumulations. The kernel used a standard deviation of 12, which is higher than for the CMEMS-phys dataset because it found higher numbers of eddies. The system was applied to the area within the cyan rectangle. The white cells above are landmass that belongs to Greenland, and at the bottom is Norway's.

4.4 General assessment of results

The proposed eddy prediction system provided satisfactory results for SINMOD and other models and observational datasets. The machine learning model used to predict the results are accompanied by post-processing procedures, which I have found to improve collective systems performance significantly. There were negligible similarities between the assimilated and observed ocean dynamics after analyzing the system's performance across the datasets. However, a year-long comparison seemed to find a seemingly comparable seasonal trend in the eddy activity with significant correlations. As expected, the eddy activity seems to be more vigorous during winter. The activity also seems to be highly associated with the formation and slope of the bathymetry. Compared to other methods studied before this project, the model has provided promising results. I believe that the system can be used experimentally, but some further post-processing would refine the process and eliminate most of the errors. It is also worth mentioning that the results seem to indicate that CMEMS-multiobs should be used with caution if it were to be used to predict eddies for a model. Although it is supposed to represent the true image of the ocean, there seem to be discrepancies in its flow field and resulting eddies.

4.5 Future work

4.5.1 Training data

The fact that there are no public applications for annotating two-dimensional grids of arbitrary measurements (at least to our knowledge after researching the domain) is restricting for the AI community within most areas of research, apart from image classification. Since machine learning techniques have proven to be excellent tools for handling a huge variety of difficult problems that arise in atmospheric, climate and marine science, there most definitely is a market for developing a simple two- (horizontal) or even three-dimensional (and vertical) data annotation tool. The problems are usually data-driven problems concerning optimization, classification, or prediction, among others, and annotated data is scarce. The application should be able to provide layers of either heatmaps of direct measurements of temperature, spatial, or density, or the combination of measurements such as velocity vector fields.

It is arguable that by only looking at the sea surface current or any single layer for that matter, could deceive the user in some cases, and some other event caused the seemingly desired features. By employing all vertical layers of a dataset, the model would be more robust as the features will persist throughout the depths for genuine eddies. However, this would significantly increase the input size. The training data collection system should also allow for scanning through layers for eddies instead of only the surface data, making the user more confident when labeling samples.

4.5.2 Machine learning methods

Section 3.2.3 shortly introduced the notion that the Faster R-CNN [61] object detection algorithm had been tested. However, the current R-CNN algorithms are explicitly created

for object detection in images and required too much modification. It is foreseeable that a special-purpose eddy detection system based on the existing attributes of R-CNN algorithms. For that matter, any two- or three-dimensional measurement data should be able to use an open-source object detection algorithm, which only would need the measurements and a document describing all bounding boxes as training input.

The algorithms used for this project provided satisfactory results. CNNs has only been introduced into atmospheric and oceanic sciences in the last couple of years, and a more comprehensive study of what best structure best captures ocean variables' features should be used not only for features such as eddies and fronts.

4.5.3 General system

Although the accuracy of the model was high, there were a few false positives most likely due to the missing representation in the background relative to the straightforward features inherent to the eddies. The post-processing undoubtedly helped avoid misclassifications to create a more modularized and improved system. Most likely, there exist better solutions, and improvements to the refinement process used. For instance, the OW and vorticity procedures of separating and clustering can cause amalgamating (fusing) eddies of the same rotation to be classified as the same eddy because the OW threshold is too low (more cells believed to be OW cells and hence merging clusters). Parameters such as the OW threshold should not be hard-coded, but follow some measure related to the average ocean current in a particular area.

Chapter 5

Conclusion

The state-of-the-art detection of ocean features seems to be predominantly algorithms comprising of more filtering models and signal processing techniques of classification using flow field analysis mechanisms such as the Okubo-Weiss (OW) parameter. In the more recent decade, there appears to be an influx of academic interest in using a more pattern recognition and learning approach, as we are entering an era dominated by computer vision and deep learning.

The goal of this project was to analyze relevant machine learning algorithms for recognition of ocean features, such as the ocean eddy, and comparing the trained model's performance on assimilated data from SINMOD and observational data from the Copernicus Marine Service. After testing SVM, random forest, and CNN on collected training data consisting of northward and eastward ocean current velocity, a CNN using a modified VGG architecture achieved the best performance with an accuracy above 96% on a hold-out set. The model was combined with post-processing procedures utilizing vorticity and OW-parameter to produce more accurate and refined boundary boxes. The system has been proven to produce satisfactory results when applied to any of the three assimilated or observatory datasets. However, there is likely room for improvement and refinement as the algorithms and architectures tested were simple modifications. Further investigation of algorithms was not prioritized as a part of the limited resources allocated for this project. After analyzing the system's performance across the datasets, there were negligible similarities between the assimilated and observed ocean dynamics. However, a year-long comparison seemed to find a seemingly comparable seasonal trend in the eddy activity with significant correlations.

Bibliography

[1] Dag Slagstad and Thomas A. McClimans. Modeling the ecosystem dynamics of the barents sea including the marginal ice zone: I. physical and chemical oceanography. *Journal of Marine Systems*, 58(1):1–18, 2005. ISSN 0924-7963. doi: https://doi.org/10.1016/j.jmarsys.2005.05.005. URL http://www.sciencedirect.com/science/article/pii/S0924796305001296.

[2] Pierre-Yves Traon, Alfatih Ali, Enrique Alvarez Fanjul, L. Aouf, Lars Axell, Roland Aznar, Maxime Ballarotta, Arno Behrens, Benkiran Mounir, A. Bentamy, Laurent Bertino, P. Bowyer, Vittorio Brando, L. Breivik, Bruno Buongiorno Nardelli, S. Cailleau, Stefania Ciliberti, Emanuela Clementi, S. Colella, and Hao Zuo. The copernicus marine environmental monitoring service: Main scientific achievements and future prospects. *Mercator Ocean Journal*, 56, 2017.

[3] 2016. URL https://resources.marine.copernicus.eu/?option=com_csw&task=results?option=com_csw&view=details&product_id=GLOBAL_ANALYSIS_FORECAST_PHY_001_024.

[4] URL https://resources.marine.copernicus.eu/?option=com_csw&task=results?option=com_csw&view=details&product_id=MULTIOBS_GLO_PHY_NRT_015_001.

[5] J. M. A. C. Souza, C. de Boyer Montégut, and P. Y. Le Traon. Comparison between three implementations of automatic identification algorithms for the quantification and characterization of mesoscale eddies in the south atlantic ocean. *Ocean Sci.*, 7 (3):317–334, 2011. ISSN 1812-0792. doi: 10.5194/os-7-317-2011. URL https://www.ocean-sci.net/7/317/2011/.

[6] Akira Okubo. Horizontal dispersion of floatable particles in the vicinity of velocity singularities such as convergences. *Deep Sea Research and Oceanographic Abstracts*, 17(3):445–454, 1970. ISSN 0011-7471. doi: https://doi.org/10.1016/0011-7471(70)90059-8. URL http://www.sciencedirect.com/science/article/pii/0011747170900598.

[7] John Weiss. The dynamics of enstrophy transfer in two-dimensional hydrodynamics. *Physica D: Nonlinear Phenomena*, 48(2):273–294, 1991. ISSN 0167-2789. doi: https://doi.org/10.1016/0167-2789(91)90088-Q. URL http://www.sciencedirect.com/science/article/pii/016727899190088Q.

[8] Dudley B. Chelton, Michael G. Schlax, Michael H. Freilich, and Ralph F. Milliff. Satellite measurements reveal persistent small-scale features in ocean winds. *Science*, 303(5660):978, 2004. doi: 10.1126/science.1091901. URL http://science.sciencemag.org/content/303/5660/978.abstract.

[9] I. Frenger, N. Gruber, R. Knutti, and M. Münnich. Imprint of southern ocean eddies on winds, clouds and rainfall. *Nature Geoscience*, 6(8):608–612, 2013. ISSN 1752-0908. doi: 10.1038/ngeo1863. URL https://doi.org/10.1038/ngeo1863.

[10] Dudley B. Chelton, Peter Gaube, Michael G. Schlax, Jeffrey J. Early, and Roger M. Samelson. The influence of nonlinear mesoscale eddies on near-surface oceanic chlorophyll. *Science*, 334(6054):328, 2011. doi: 10.1126/science.1208897. URL http://science.sciencemag.org/content/334/6054/328.abstract.

[11] P. Gaube, D. B. Chelton, P. G. Strutton, and M. J. Behrenfeld. Satellite observations of chlorophyll, phytoplankton biomass, and ekman pumping in nonlinear mesoscale eddies. *Journal of Geophysical Research: Oceans*, 118(12):6349–6370, 2013. ISSN 2169-9275. doi: 10.1002/2013JC009027. URL https://doi.org/10.1002/2013JC009027.

[12] Stephanie A. Henson and Andrew C. Thomas. A census of oceanic anticyclonic eddies in the gulf of alaska. *Deep Sea Research Part I: Oceanographic Research Papers*, 55(2):163–176, 2008. ISSN 0967-0637. doi: https://doi.org/10.1016/j.dsr.2007.11.005. URL http://www.sciencedirect.com/science/article/pii/S0967063707002592.

[13] Sachihiko Itoh and Ichiro Yasuda. Characteristics of mesoscale eddies in the kuroshio–oyashio extension region detected from the distribution of the sea surface height anomaly. *Journal of Physical Oceanography*, 40(5):1018–1034, 2009. ISSN 0022-3670. doi: 10.1175/2009JPO4265.1. URL https://doi.org/10.1175/2009JPO4265.1.

[14] Changming Dong, James C. McWilliams, Yu Liu, and Dake Chen. Global heat and salt transports by eddy movement. *Nature Communications*, 5(1):3294, 2014. ISSN 2041-1723. doi: 10.1038/ncomms4294. URL https://doi.org/10.1038/ncomms4294.

[15] R. M. Samelson, M. G. Schlax, and D. B. Chelton. Randomness, symmetry, and scaling of mesoscale eddy life cycles. *Journal of Physical Oceanography*, 44(3):1012–1029, 2013. ISSN 0022-3670. doi: 10.1175/JPO-D-13-0161.1. URL https://doi.org/10.1175/JPO-D-13-0161.1.

[16] Zhengguang Zhang, Wei Wang, and Bo Qiu. Oceanic mass transport by mesoscale eddies. *Science*, 345(6194):322, 2014. doi: 10.1126/science.1252418. URL http://science.sciencemag.org/content/345/6194/322.abstract.

[17] Ute Hausmann and Arnaud Czaja. The observed signature of mesoscale eddies in sea surface temperature and the associated heat transport. *Deep Sea Research Part I: Oceanographic Research Papers*, 70:60–72, 2012. ISSN 0967-0637. doi: https://doi.org/10.1016/j.dsr.2012.08.005. URL http://www.sciencedirect.com/science/article/pii/S0967063712001720.

[18] Mark Petersen, Sean Williams, Mathew Maltrud, Matthew Hecht, and Bernd Hamann. A three-dimensional eddy census of a high-resolution global ocean simulation. *Journal of Geophysical Research (Oceans)*, 118:1759–1774, 2013. doi: 10.1002/jgrc.20155.

[19] Jordi Isern-Fontanet, E. Garc'ia-Ladona, and J. Font. Identification of marine eddies from altimetry. *J. Atmos. Oceanic Technol.*, 20:772–778, 2003.

[20] Jordi Isern-Fontanet, Jordi Font, Emilio García-Ladona, Mikhail Emelianov, Claude Millot, and Isabelle Taupier-Letage. Spatial structure of anticyclonic eddies in the algerian basin mediterranean sea) analyzed using the okubo–weiss parameter. *Deep Sea Research Part II: Topical Studies in Oceanography*, 51:3009–3028, 2004. doi: 10.1016/j.dsr2.2004.09.013.

[21] Jordi Isern-Fontanet, Emilio García-Ladona, Jordi Font, and Antonio Garcia-Olivares. Non-gaussian velocity probability density functions: An altimetric perspective of the mediterranean sea. *Journal of Physical Oceanography*, 36:2153, 2006. doi: 10.1175/JPO2971.1.

[22] Evan Mason, Ananda Pascual, and James C. McWilliams. A new sea surface height–based code for oceanic mesoscale eddy tracking. *Journal of Atmospheric and Oceanic Technology*, 31(5):1181–1188, 2014. ISSN 0739-0572. doi: 10.1175/JTECH-D-14-00019.1. URL https://doi.org/10.1175/JTECH-D-14-00019.1.

[23] J. M. Lilly, R. K. Scott, and S. C. Olhede. Extracting waves and vortices from lagrangian trajectories. *Geophysical Research Letters*, 38(23), 2011. ISSN 0094-8276. doi: 10.1029/2011GL049727. URL https://doi.org/10.1029/2011GL049727.

[24] Mohammad D. Ashkezari, Christopher N. Hill, Christopher N. Follett, Gaël Forget, and Michael J. Follows. Oceanic eddy detection and lifetime forecast using machine learning methods. *Geophysical Research Letters*, 43(23):12,234–12,241, 2016. ISSN 0094-8276. doi: 10.1002/2016GL071269. URL https://doi.org/10.1002/2016GL071269.

[25] Marco Castellani. Identification of eddies from sea surface temperature maps with neural networks. *International Journal of Remote Sensing*, 27:1601–1618, 2006. doi: 10.1080/01431160500462170.

[26] Ribana Roscher, Andres Milioto, Susanne Wenzel, and Jürgen Kusche. Ocean eddy identification and tracking using neural networks. 2018.

[27] A. Durmagambetov and L. Fazilova. Navier-stokes equations—millennium prize problems. *Natural Science*, pages 88–99, 2015. doi: 10.4236/ns.2015.72010.

[28] Sea-viewing wide field-of-view sensor (seawifs) ocean color data.

[29] Detlef Stammer. Global characteristics of ocean variability estimated from regional topex/poseidon altimeter measurements. *Journal of Physical Oceanography*, 27 (8):1743–1769, 1997. ISSN 0022-3670. doi: 10.1175/1520-0485(1997)027⟨1743:Gcoove⟩2.0.Co;2. URL https://doi.org/10.1175/1520-0485(1997)027<1743:GCOOVE>2.0.CO;2.

[30] Detlef Stammer. On eddy characteristics, eddy transports, and mean flow properties. *Journal of Physical Oceanography*, 28(4):727–739, 1998. ISSN 0022-3670. doi: 10.1175/1520-0485(1998)028⟨0727:Oeceta⟩2.0.Co;2. URL https://doi.org/10.1175/1520-0485(1998)028<0727:OECETA>2.0.CO;2.

[31] Robert N. Miller. *Numerical Modeling of Ocean Circulation*. Cambridge University Press, Cambridge, 2007. ISBN 9780521781824. doi: DOI:10.1017/CBO9780511618512. URL https://www.cambridge.org/core/books/numerical-modeling-of-ocean-circulation/644C657062CE1E8BBDC8622E1386AD7C.

[32] Zhengguang Zhang, Wei Wang, and Bo Qiu. Oceanic mass transport by mesoscale eddies. *Science (New York, N.Y.)*, 345(6194):322–324, 2014. ISSN 00368075. doi: 10.1126/science.1252418.

[33] Adrian E. Gill, editor. *Chapter Thirteen - Instabilities, Fronts, and the General Circulation*, volume 30, pages 549–593. Academic Press, 1982. ISBN 0074-6142. doi: https://doi.org/10.1016/S0074-6142(08)60038-8. URL http://www.sciencedirect.com/science/article/pii/S0074614208600388.

[34] D. Randall. *Introduction to the Global Circulation of the Atmosphere (ARC)*. Princeton University Press. ISBN 9781400897445. URL https://books.google.no/books?id=3MZHAQAACAAJ.

[35] May 30, 2018 . URL https://www.britannica.com/science/ocean-current.

[36] Yunus A. Çengel. *Thermodynamics : an engineering approach*. Sixth edition. Boston : McGraw-Hill Higher Education, [2008] ©2008, 2008. URL https://search.library.wisc.edu/catalog/9910062108002121.

[37] Matthieu Roy-Barman and Catherine Jeandel. *Marine Geochemistry: Ocean Circulation, Carbon Cycle and Climate Change*. 2016. ISBN ISBN-13: 978-0198787495.

[38] Lynne Talley, George Pickard, William Emery, and James Swift. *Chapter 7. Dynamical Processes for Descriptive Ocean Circulation*. 2011. doi: 10.1016/B978-0-7506-4552-2.10007-1.

[39] Christof Koch, Öjvind Bernander, and Rodney J. Douglas. Do neurons have a voltage or a current threshold for action potential initiation? *Journal of Computational Neuroscience*, 2(1):63–82, 1995. ISSN 1573-6873. doi: 10.1007/BF00962708. URL https://doi.org/10.1007/BF00962708.

[40] Anthony Gidudu, Hulley Greg, and Tshilidzi Marwala. Classification of images using support vector machines. 2007.

[41] Vladimir N. Vapnik. *Nature of Statistical Learning Theory*. New York: Springer New York, 1st ed.. edition, 1995. ISBN 9781475724400 9781475724424 9781475732641. doi: 10.1007/978-1-4757-3264-1.

[42] Martin D. Buhmann and M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2003. ISBN 0521633389.

[43] scikit learn, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn machine learning in python. *Journal of Machine Learning Research*, 12, 2011.

[44] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, and Xindong Wu. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–21, 2019. doi: 10.1109/TNNLS.2018.2876865.

[45] Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman Hall/CRC, 1st edition, 2012. ISBN 1439830037.

[46] B. Efron. Bootstrap methods: Another look at the jackknife. *Ann. Statist.*, 7(1): 1–26, 1979. ISSN 0090-5364. doi: 10.1214/aos/1176344552. URL `https://projecteuclid.org:443/euclid.aos/1176344552`.

[47] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1): 106–154, 1962. ISSN 0022-3751 1469-7793. doi: 10.1113/jphysiol.1962.sp006837. URL `https://pubmed.ncbi.nlm.nih.gov/14449617https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1359523/`.

[48] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vision*, 115(3):211–252, 2015. ISSN 0920-5691. doi: 10.1007/s11263-015-0816-y. URL `https://doi.org/10.1007/s11263-015-0816-y`.

[49] Krizhevsky Alex, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. pages 1097–1105, 2012. URL `http://www.cs.toronto.edu/~hinton/absps/imagenet.pdf`.

[50] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 2014.

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2016. doi: 10.1109/CVPR.2016.90.

[52] C. Szegedy, Liu Wei, Jia Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9. ISBN 1063-6919. doi: 10.1109/CVPR.2015.7298594.

[53] Manolis Loukadakis, José Cano, and Michael O'Boyle. *Accelerating Deep Neural Networks on Low Power Heterogeneous Architectures*. 2018.

[54] Georgescu Mariana-Iuliana, Ionescu Radu Tudor, Ristea Nicolae-Catalin, and Sebe Nicu. *Non-linear Neurons with Human-like Apical Dendrite Activations*. 2020. doi: 10.36227/techrxiv.11830761.v1. URL https://www.techrxiv.org/articles/Non-linear_Neurons_with_Human-like_Apical_Dendrite_Activations/11830761.

[55] Alice Zheng and Amanda Casari. *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*. O'Reilly Media, Inc., 2018. ISBN 1491953241.

[56] Sebastian Ruder. An overview of gradient descent optimization algorithms. 2016.

[57] Andre de Carvalho and Alex Freitas. *A Tutorial on Multi-label Classification Techniques*, volume 205, pages 177–195. 2009. doi: 10.1007/978-3-642-01536-6_8.

[58] Andrew Ng. *Machine Learning Yearning*. deeplearning.ai, 2018.

[59] Yeping Yuan and Renato M. Castelao. Eddy-induced sea surface temperature gradients in eastern boundary current systems. *Journal of Geophysical Research: Oceans*, 122(6):4791–4801, 2017. ISSN 2169-9275. doi: 10.1002/2017JC012735. URL https://doi.org/10.1002/2017JC012735.

[60] Stephanie Henson and Andrew Thomas. A census of oceanic anticyclonic eddies in the gulf of alaska. *Deep Sea Research Part I: Oceanographic Research Papers*, 55: 163–176, 2008. doi: 10.1016/j.dsr.2007.11.005.

[61] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017. ISSN 1939-3539. doi: 10.1109/TPAMI.2016.2577031.

[62] Luo Juan and Oubong Gwun. A comparison of sift, pca-sift and surf.

[63] Pål Erik Isachsen. Baroclinic instability and the mesoscale eddy field around the lofoten basin. *Journal of Geophysical Research: Oceans*, 120(4):2884–2903, 2015. ISSN 2169-9275. doi: 10.1002/2014JC010448. URL https://doi.org/10.1002/2014JC010448.