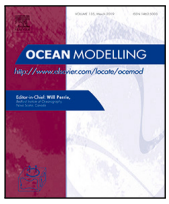
Ocean Modelling 196 (2025) 102543

Contents lists available at ScienceDirect

Ocean Modelling

journal homepage: www.elsevier.com/locate/ocemod



Surface drifter trajectory prediction in the Gulf of Mexico using neural networks

Matthew D. Grossi [a,*], Stefanie Jegelka [b], Pierre F.J. Lermusiaux [b], Tamay M. Özgökmen [a]

[a] Rosenstiel School of Marine, Atmospheric, and Earth Science, University of Miami, Florida, United States of America
[b] Massachusetts Institute of Technology, Cambridge, Massachusetts, United States of America

ARTICLE INFO

Keywords:
Advection
Current velocity
Mass transport
Lagrangian motion
Current prediction
Machine learning

ABSTRACT

Machine learning techniques are applied to Lagrangian trajectory reconstructions, which are important in oceanography for providing guidance to search and rescue efforts, forecasting the spread of harmful algal blooms, and tracking pollutants and marine debris. This study evaluates the ability of two types of neural networks for learning ocean trajectories from nearly 250 surface drifters released during the Grand Lagrangian Deployment in the Gulf of Mexico from Jul-Oct 2012. First, simple fully connected neural networks were trained to predict an individual drifter's trajectory over 24 h and 5 d time windows using only that drifter's previous velocity time series. These networks, despite having successfully learned modeled trajectories in a previous study, failed to outperform common autoregressive models in any of the tests conducted. This was true even when drifters were pre-sorted into geospatial groups based on past trajectories and different networks were trained on each group to reduce the variability that each network had to learn. In contrast, a more sophisticated social spatio-temporal graph convolutional neural network (STN), originally developed for learning pedestrian trajectories, demonstrated greater potential due to two important features: learning spatial and temporal patterns simultaneously, and sharing information between similarly-behaving drifters to facilitate the prediction of any particular drifter. Position prediction errors averaged around 60 km at day 5, roughly 20 km lower than autoregression, and even better for certain subsets of drifters. The passage of Tropical Cyclone Isaac over the drifter array as a tropical storm and Category 1 hurricane provided a unique opportunity to also explore whether these models would benefit from adding wind as a predictor when making short 24 h predictions. The STNs were found to not benefit from wind on average, though certain subsets of drifters exhibited slightly lower reconstruction errors at hour 24 with the addition of wind.

1. Introduction

Ocean trajectory prediction is a notoriously difficult problem. Most notably, the unsteady nature of oceanic flows often leads to chaotic advection (*e.g.*, Aref, 1984; Yang and Liu, 1997; Özgökmen et al., 2001; Koshel' and Prants, 2006), requiring that initial conditions be known with considerable accuracy in order to properly initialize forecast models. At the same time, the minimum number of points in time and space that must simultaneously be sampled or numerically resolved in order to capture the full complexity of 3-D ocean dynamics far exceeds the technical capabilities of modern observing systems. With observational data density nowhere near this requirement and with model resolutions restricted by computational resource limits, existing prediction tools lack the fidelity necessary for predicting chaotic ocean behavior (Özgökmen et al., 2009; Bolton and Zanna, 2019). Nevertheless, many high-stakes applications such as oil spill response (Poje et al., 2014; Özgökmen et al., 2016), search and rescue operations (Isaji et al., 2005; Serra et al., 2020), and forecasting the spread of harmful algae blooms, pollutants, and marine debris (Enriquez et al., 2010; Olascoaga, 2010; Olascoaga and Haller, 2012; Normile, 2014; Coulin et al., 2017; Lermusiaux et al., 2019) rely on ocean forecasting.

Existing approaches to ocean forecasting include data-assimilating ocean models (Coelho et al., 2015; Wei et al., 2016; van Sebille et al., 2018) and statistical stochastic models (Griffa, 1996; Berloff and McWilliams, 2003; Lermusiaux and Lekien, 2005; Haza et al., 2016; Feppon and Lermusiaux, 2018; Lu and Lermusiaux, 2021). The problem of sparse ocean data plagues both techniques. Ocean general circulation models (OGCMs) such as the Hybrid Coordinate Ocean Model (HYCOM; Chassignet et al., 2003, 2007) are challenging to initialize without

* Corresponding author.
E-mail addresses: matthew.grossi@earth.miami.edu (M.D. Grossi), stefje@mit.edu (S. Jegelka), pierrel@mit.edu (P.F.J. Lermusiaux), tozgokmen@earth.miami.edu (T.M. Özgökmen).

https://doi.org/10.1016/j.ocemod.2025.102543
Received 31 January 2024; Received in revised form 28 January 2025; Accepted 26 March 2025
Available online 11 April 2025
1463-5003/Published by Elsevier Ltd.

adequate data, especially in three dimensions (Lermusiaux, 2001; Jacobs et al., 2014). Stochastic models contain statistical parameters that must be tuned with data if the flow field is to be faithfully replicated. Improper tuning hinders performance (Beron-Vera and LaCasce, 2016; Mariano et al., 2016). Both of these families of models use data only to establish the starting conditions from which the model estimates future states using either equations of motion or statistics.

Machine learning (ML) provides a new and different approach to ocean modeling wherein rich nonlinear regression models are optimized using existing data to map known information to expected output. Such data-driven frameworks allow for the possibility that observational ocean data, though sparse, may still contain enough information to generalize the problem and train a ML model. Artificial neural networks are a family of biologically-inspired ML models that map input features to output target values and form the foundation of sophisticated ML and deep learning architectures (Qamar and Zardari, 2023). Artificial neural networks have been used for a variety of geophysical and oceanographic problems including weather forecasting (Dueben and Bauer, 2018), hurricane track prediction (Moradi Kordmahalleh et al., 2016), oil spill prediction (Kubat et al., 1998), and eddy tracking (Franz et al., 2018). Hybrid modeling approaches integrating theory with machine learning have shown potential for ocean forecasting and improving trajectory predictions. Aksamit et al. (2020) used long short-term memory recurrent neural networks to learn motion corresponding to higher order terms in a reduced-order Maxey–Riley drifter model. Nam et al. (2020) took a conventional ML approach by using model-derived wind and flow velocity to predict drifter location. Deep neural operator models have also been evaluated for realistic surface ocean forecasting in diverse dynamical regimes (*e.g.*, Rajagopal et al., 2023). Others address the problems of sub-grid scale model closure schemes (Gupta and Lermusiaux, 2021) and model parameterizations (Santos Gutiérrez et al., 2021). Grossi et al. (2020) approached the trajectory prediction problem in a data-only context and showed that fully connected artificial neural networks could predict future velocities of hypothetical ocean particles using only the previous velocity time series with errors averaging nearly half those of traditional autoregressive techniques.

The fully connected artificial neural network (FNN) used by Grossi et al. (2020) (hereafter, G20) was an initial experimentation at applying ML to the Lagrangian prediction problem and therefore had a few notable shortcomings. Though those authors tested trajectories in a flow field generated by HYCOM and characterized by interacting submesoscale and mesoscale dynamics in the Gulf of Mexico (GoM), their simulation did not consider any observed Lagrangian trajectory data. Further, they predicted only 24 h out, a time period known for being dominated by predictable inertial oscillations, while many material transport applications would benefit from having forecasts on the order of days. The FNN, which only "saw" one trajectory at a time, was unable to learn any underlying flow dynamics because observing 2D surface dispersion requires $N \geq 3$ Lagrangian drifters simultaneously (Pumir et al., 2000; Berta et al., 2016). Compared to traditional OGCMs and stochastic models built upon clean mathematical formulation of equations of motion or statistical parameterizations, classic neural networks provide little to no transparency into what the model is learning. This can be at least partially overcome by incorporating human intuition and domain knowledge – *i.e.*, human expertise of the physical system being modeled – into the ML model, but the simple FNN by G20 lacked such sophistication. Finally, their training set contained few examples by ML standards, making it difficult to learn the chaotic nature of ocean dynamics.

Here we address the shortcomings of G20 and build upon that previous work in several ways. We first test their FNNs on observed ocean drifter trajectories from the GoM and then modify the networks to predict five days out instead of 24 h. We then present a more physically intuitive approach by utilizing a graph neural network (GNN) that accepts multiple drifters at a time as input and allows behavioral information to be shared between drifters to facilitate the prediction of any particular drifter. GNNs were first proposed by Gori et al. (2005) for problems that can be described using mathematical graphs consisting of nodes and edges, where edges heuristically quantify relationships between data points represented as graph nodes. GNNs receive graphs as inputs and are well-suited for problems that can be formulated such that nodes contain observations and edges represent relationships or connectivity between the observations, as will be described in Section 4. Since then, a family of problem-specific GNNs has evolved (*e.g.*, Merkwirth and Lengauer, 2005; Scarselli et al., 2009; Zhou et al., 2018; Bianchi et al., 2019; Wu et al., 2020). Among these is the graph convolutional neural network (GCNN) (Gilmer et al., 2017; Hamilton et al., 2017; Kipf and Welling, 2017), which operates on the node and edge information of graphs in a similar fashion to how convolutional neural networks learn features within multispectral images in computer vision (Schlichtkrull et al., 2017). Mohamed et al. (2020) advanced these concepts further by developing a Social Spatio-Temporal Graph Convolutional Neural Network (STN) for predicting pedestrian trajectories in video scenes by learning both spatial and temporal patterns in pedestrians' previous trajectories while also accounting for non-verbal exchanges of information between people in social settings. These authors demonstrated performance improvement over common trajectory prediction models for multiple benchmark pedestrian data sets. The inspirations behind this STN – learning patterns in both space and time, sharing information between elements – make this type of model a compelling option for other trajectory prediction problems as well. We apply STNs to the drifter prediction problem according to the premise that the motion of adjacent drifters can be similar or dissimilar due to the underlying flow field, and learning these (dis)similarities may facilitate trajectory prediction.

Section 2 describes the drifter data used in this study. Altogether, six neural network architectures were evaluated: two FNNs, described in Section 3, and four STNs (Section 4). All are summarized in Table 1 and a side-by-side comparison of the two types of neural networks is provided in Table 2. Section 5 contrasts the two methods and discusses implications of each. Finally, conclusions are drawn in Section 6 along with suggestions for further advances that could be made.

2. Drifter data

The Grand LAgrangian Deployment (GLAD) experiment consisted of nearly 300 CODE-style GPS-equipped drifters with 1-m drogue (Davis, 1985) released near the former *Deepwater Horizon* site in the northern GoM in summer 2012 (Fig. 1a). Drifter positions were reported every 5 min and trajectories were later low-pass filtered using a 1 h cutoff period and interpolated to 15 min intervals starting on whole hours (Özgökmen, 2013). Initially deployed on the order of 100 m apart starting 20 Jul 2012, the drifters covered the entire eastern half of the GoM basin by 22 Oct, with the majority of drifters staying within the northeast quadrant (Fig. 1b). Time snapshots of surface mesoscale geostrophic velocities derived from NOAA CoastWatch 0.25° satellite altimetry throughout the experiment are shown in Fig. 2.[1]

We utilized 243 drifters from three deployments: two "*S*" formations consisting of 90 drifters – one near the former oil well (blue in Fig. 1a, hereafter *S*1) and a second release east of the well (orange, *S*2) – and a pair of "*L*"-shaped deployments to the southwest of *S*1 and *S*2 containing 63 drifters (red, hereafter "*L*"). The formations were organized in nodes containing three groups of three drifter triplicates in order to capture a range of scales of surface flow dynamics (Berta et al., 2016). Fig. 1c zooms in on *S*2 to show the 10 nodes of 3 groups (dots), while Fig. 1d zooms in on a group to show the triplicate of drifters

[1] Sea Surface Height Anomalies, Altimetry (S-3A/B, CryoSat2, Jason-2/3, SARAL), Delayed, Global 0.25°, 2012–2019. Retrieved from NOAA CoastWatch ERDDAP, 20 March 2023.

Table 1
Summary of neural networks developed and tested in this study, ordered as presented in the text. $d(\ldots)$ indicates the change in location (lon, lat) between observations, a proxy for velocity which was not normalized by dt because dt was constant throughout the entire dataset. Similarly, $d^2(\ldots)$ is a non-normalized proxy for acceleration: change in $d(\ldots)$ between observations. All outputs are $[d(\text{lon}), d(\text{lat})]$ pairs.

Model	INPUT TIME SERIES			PREDICTION	
	Variables	Length	Increment	Length	Increment
FNN	$d(\text{lon}), d(\text{lat})$	24 h	3 h	24 h	6 h
	$d(\text{lon}), d(\text{lat})$	7 d	1 d	5 d	1 d
STN	$d(\text{lon}), d(\text{lat}),\ d^2(\text{lon}), d^2(\text{lat})$	7 d	1 d	5 d	1 d
STN with wind	$d(\text{lon}), d(\text{lat}),\ d^2(\text{lon}), d^2(\text{lat}),\ u_{\text{wind}}, v_{\text{wind}}$	7 d	1 d	5 d	1 d
	$d(\text{lon}), d(\text{lat}),\ d^2(\text{lon}), d^2(\text{lat}),\ u_{\text{wind}}, v_{\text{wind}}$	24 h	3 h	24 h	6 h
STN, advanced configuration	$d(\text{lon}), d(\text{lat}),\ d^2(\text{lon}), d^2(\text{lat})$; see text	7 d	1 d	5 d	1 d

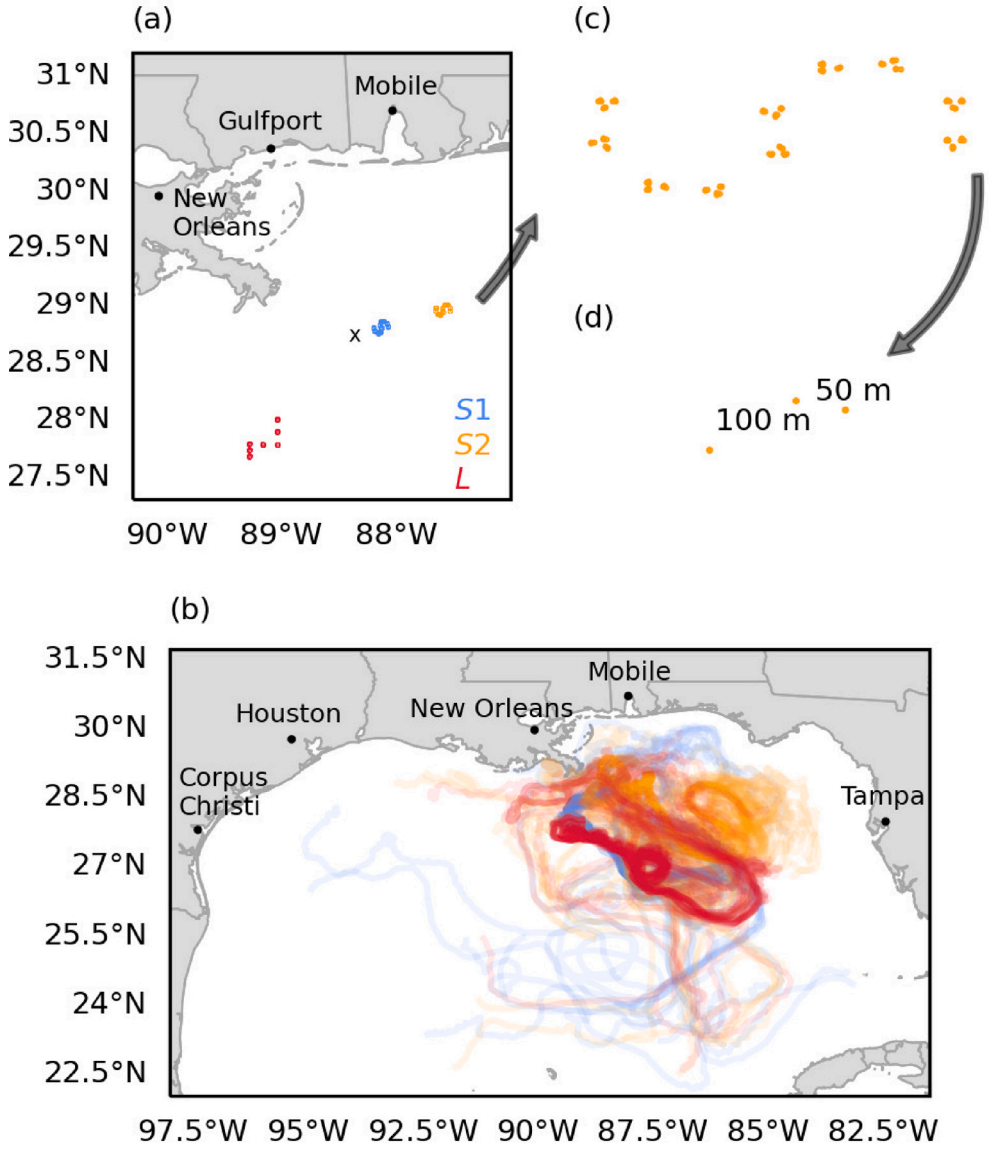


Fig. 1. Deployment locations and trajectories of all drifters released during the Grand Lagrangian Deployment (GLAD) experiment, colored according to deployment location. Drifters were released between 22–31 Jul 2012 (a) and tracked through 22 Oct (b). Two "*S*" deployment configurations (blue and orange) consisted of 90 drifters arranged in ten nodes (c), with each node containing three triplets of drifters spaced either 100 m or 50 m apart (d). A double "*L*" configuration (red) contained seven nodes of three triplets, totaling 63 drifters. Black "×" in (a) marks the Deepwater Horizon wellhead location.

released 50-100 m apart ($3 \times 3 \times 10 = 90$ drifters.) The *L* deployment was configured similarly but consisted of seven nodes ($3 \times 3 \times 7 = 63$ drifters.)

These three deployments captured a variety of multiscale surface dynamics (Poje et al., 2014). Surface geostrophic velocities in the deployment area during the first month of the experiment (Fig. 3) reveal that *S*1 and *S*2 were released on the edge of an anticyclonic mesoscale Lagrangian coherent structure (LCS) (Fig. 3a), but shortly thereafter these two groups of drifters ended up in two adjacent mesoscale LCSs (Fig. 3b). While mesoscale flows became increasingly relevant as the drifters dispersed to cover a larger spatial area, the deployment strategy was designed to capture submesoscale dynamics. Trajectories for the

Table 2
Comparison of the fully connected neural networks (FNN) and social spatio-temporal graph convolutional neural networks (STN). The number of trainable parameters is a function of network size and the lengths of the input and output time series.

FNN	STN
Sizes:	**Sizes:**
1 hidden layer, 20 neurons (24 h)	See Table 3
1 hidden layer, 40 neurons (5 d)	
Number of trainable parameters:	**Number of trainable parameters:**
508 (24 h); 1,090 (5 d)	1,637–31,435 (see Table 3)
$\mathbf{y} = f\left(\sum_j \mathbf{W} f\left(\sum_k \mathbf{W}\mathbf{x} + \mathbf{b}\right) + \mathbf{B}\right)$	$f(V^{(l)}, E) = g(D^{-\frac{1}{2}} \hat{E} D^{-\frac{1}{2}} V^{(l)} \mathbf{W}^{(l)})$
output vector $\mathbf{y} = (y_1, y_2, \ldots, y_i)$	$V^{(l)}$ = array of drifter observations
input feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$	l = network layer
n = number of observed features	D_t = diagonal node degree matrix of $\hat{E}_t$
j, k = indices of hidden neurons	$\hat{E}_t = E_t + I$
i = number of predicted values	$E_t = D_t^{-\frac{1}{2}} \hat{E}_t D_t^{-\frac{1}{2}}$
$f(x) = \frac{1}{1+e^{-\Sigma}}$ = activation function	with elements e at time t given by
Σ = weighted sum from previous network layer (sigmoid activation)	$e_t^{ij} = \begin{cases} \sum_{k=1}^{N_f} \frac{1}{\sqrt{(\mathbf{x}_t^{[k],i} - \mathbf{x}_t^{[k],j})^2}}, & (\mathbf{x}_t^{[k],i} - \mathbf{x}_t^{[k],j})^2 \neq 0 \\ 0, & \text{otherwise} \end{cases}$
$\mathbf{W}$ = trainable weight parameters	I = identity matrix
$\mathbf{B}, \mathbf{b}$ = trainable bias parameters	N_f = number of observed features
	$\mathbf{x}$ = feature vector of observations
	i, j = index of drifters
	g = PReLU activation function
	$\mathbf{W}^{(l)}$ = trainable parameters at layer l

Table 3
Summary of STNs used in this study. "STG" is the number of spatio-temporal graph CNN layers, "TXP" is the number of time-extrapolator CNN layers, and η is the learning rate. The number of trainable parameters is a function of both the size of the network and the lengths of the input and output time series.

Experiment	# STG	# TXP	η	# Trainable Parameters
5 d reconstruction	1	1	0.02	1,673
5 d reconstruction w/wind	2	5	0.02	1,789
5 d reconstruction, advanced config.	1	1	0.02	1,717
24 h reconstruction	1	1	0.03	31,415
24 h reconstruction w/wind	1	3	0.01	31,435

first 5 and 30 days, low-pass filtered to remove inertial signals and shown in Fig. 4, reveal the dominant submesoscale LCSs early in the experiment. The L drifters, released about 100 km southwest of $S1$, were advected by cyclonic submesoscale flow (which some escaped early on), while $S2$ drifters were released at a confluence of at least two submesoscale LCSs, causing the drifters to bifurcate several times shortly after deployment. $S1$ was least influenced by strong submesoscale motions, with mesoscale flows dominating throughout the experiment.

3. Fully connected neural networks

3.1. Neural network configurations

The first objective of this study was to test the simple FNNs developed by G20 on observed GLAD drifters to determine whether similar success could be achieved. In that study, the authors' intent was to develop the simplest possible neural network, rather than the best-performing one, in order to minimize the so-called "black box" abstractness of the model, to reduce the risk of overfitting, and to allow ample room for improvement with more sophisticated architectures. Those FNNs are described briefly below but the reader is referred to G20 for a more thorough exposition.

To replicate that previous study, we started with FNNs containing a single hidden layer with 20 neurons, sigmoid activation functions for all hidden and output neurons, gradient descent, and mean-squared-error cost functions. These FNNs received as input the previous 24 h time series of zonal and meridional velocity components, upsampled to 3 h observations for consistency with the previous study, to produce 24 h zonal and meridional drifter velocities in 6 h increments, from which position predictions were derived. Separate FNNs were trained on $S1$ and $S2$ drifters using 3-fold cross validation wherein 60 random drifters from the deployment were used for training and the remaining 30 were used for testing each time. We also employed a rolling window technique designed for domains that change in time (Kubat, 1989), illustrated in Fig. 5. Each network was initially trained on the first days' worth of data, after which the training set was updated every 3 h by replacing the oldest observations with the most recent. The models were trained further every time the training set was updated. Since the test drifter trajectories were predicted over the same time window as the training data targets, the model output is more akin to velocity reconstructions than forecasts. In a real-world application, this would be analogous to using deployed drifters to predict non-existent drifters within the same vicinity, as discussed in Section 5.

Velocity predictions were made every day at midnight and compared to both rudimentary persistence, the assumption that the most recently observed velocity persists indefinitely until a new observation informs otherwise, and to autoregressive integrated moving average (ARIMA) models using root-mean-squared error (RMSE):

$$\mathrm{RMSE}_{i,j} = \left[\frac{1}{KL}\sum_{k=1}^{K}\sum_{l=1}^{L}\left(Y_{ijkl} - \hat{Y}_{ijkl}\right)^2\right]^{1/2}, j = 1, \ldots, J \quad (1)$$

where Y is the attribute vector, (ˆ) indicates predicted values for prediction time i, J daily reconstructions issued throughout the deployment, K predicted attributes, and L test drifters. Since ARIMA models are best suited for long time series, these were fit to cumulative velocity time series (using the *auto.arima* routine from the "forecast" package for **R** 3.4.2) without discarding old observations. We exclude j from

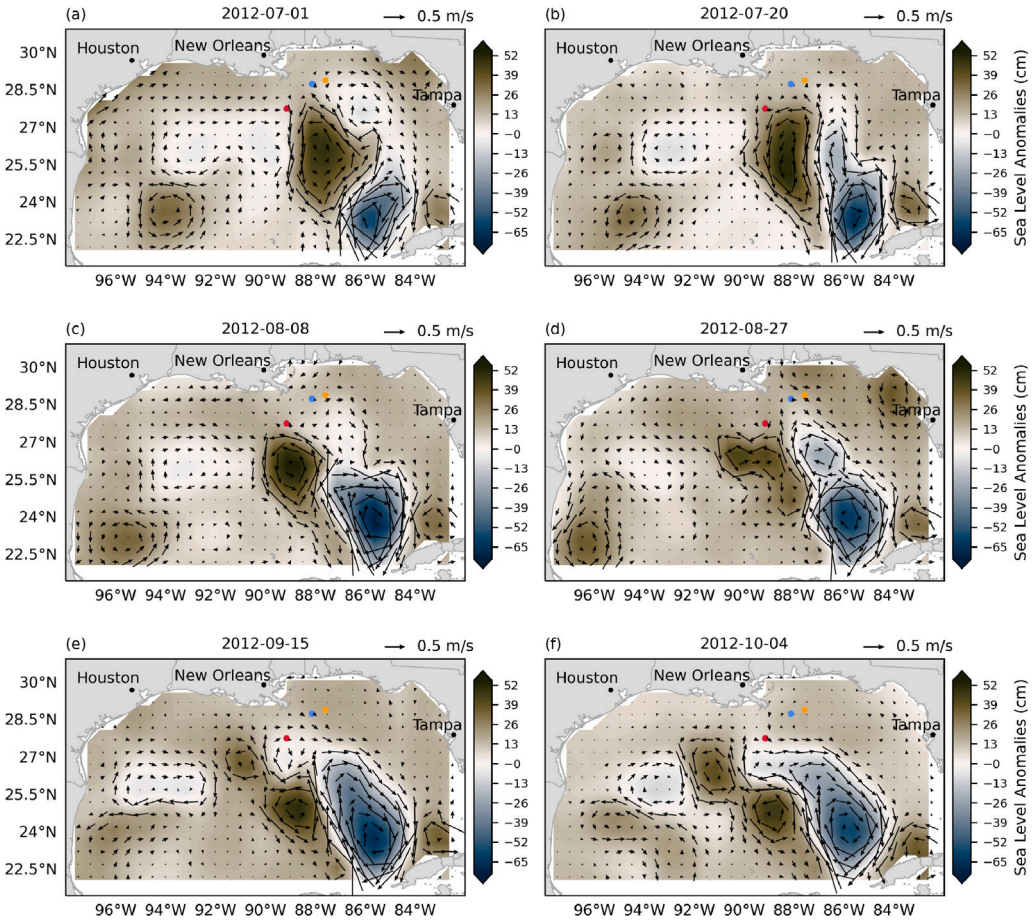


Fig. 2. Snapshots of geostrophic surface velocities (vectors, ms^{-1}) derived from satellite altimetry (colors, cm) throughout the entire GLAD experiment. Drifter deployment locations added for reference: *S*1, blue; *S*2, orange; *L*, red.

the RMSE averaging because each daily reconstruction was issued by a different version of the FNN by virtue of the additional training that took place between each reconstruction.

A second objective of this study was to explore enhancements to these FNNs that might improve their practicality and performance. To accomplish this, the experiment was repeated with three methodological changes. First, recognizing that predicting trajectories on hourly scales is considerably easier yet less operationally useful than several days out, we modified the FNN to consist of a single hidden layer containing 40 neurons designed to predict 5 d out using the previous 7 d of observations in order to better explore the practical utility of such an FNN. This meant that each training trajectory for the FNN spanned 12 days (see Fig. 5), which introduced a logistical dilemma. ML algorithms, as statistical models, require many statistically independent examples from which to learn. With that in mind, on one hand, advancing the sliding window by 15 min (the sampling frequency of the dataset) to generate new examples as the experiment progressed would result in significant temporal overlap between examples, since each 12-day time series would differ only by 15 min on either end. On the other hand, upsampling the data from 15 min (96 observations/day per drifter) to daily observations to minimize (though not eliminate) the interdependence problem would result in nearly 99% fewer coveted training examples and would sacrifice time series resolution. We compromised by upsampling the data to hourly observations and advancing the rolling window in 1 d increments. This also filtered out inertial oscillations that, due to their regular periodicity, were easy to learn (see, *e.g.*, Case 2 in G20), although we note that inertial signals can be associated with areas of convergence or divergence and heavily influenced by strong submesocale vorticity fields.

Another modification was to implement a divide-and-conquer approach to learning trajectories by systematically sorting drifters according to behavior. Rather than train a single FNN on all GLAD drifters that collectively captured a wide range of dynamics, we integrated an unsupervised hierarchical clustering routine from the Python library "scipy" (version 1.5.4 for Python 3.8.3) to objectively group drifters based on their historic trajectories and then induced separate FNNs for each group. At each observation time, the latitudes and longitudes of all active drifters for the past 7 d were passed to the clustering algorithm, which started with every drifter in its own group and sorted using spatial Euclidean distances. A sample clustering for *S*2 drifters within the vicinity of a surface divergence zone is shown in Fig. 6. A threshold of 20 drifters per cluster was set to ensure that each group had a sufficient number of drifters for training and testing; if any cluster had fewer than 20 drifters, we reverted back to a single cluster and induced a single FNN. We now present the results of these first two FNN experiments.

3.2. Results: Fully connected neural networks

Averaging Eq. (1) over the J reconstructions generated throughout the experiment quantifies RMSE for each time step in the 24 h reconstruction window (*i.e.*, RMSE for hour 6, 12, …, 24):

$$\overline{\mathrm{RMSE}}_i = \frac{1}{J}\sum_{j=1}^{J} \mathrm{RMSE}_{ij} \tag{2}$$

where $J = 90$ and 86 reconstructions for *S*1 and *S*2, respectively, from 20 Jul to 22 Oct 2012. Fig. 7 summarizes drifter position error for the

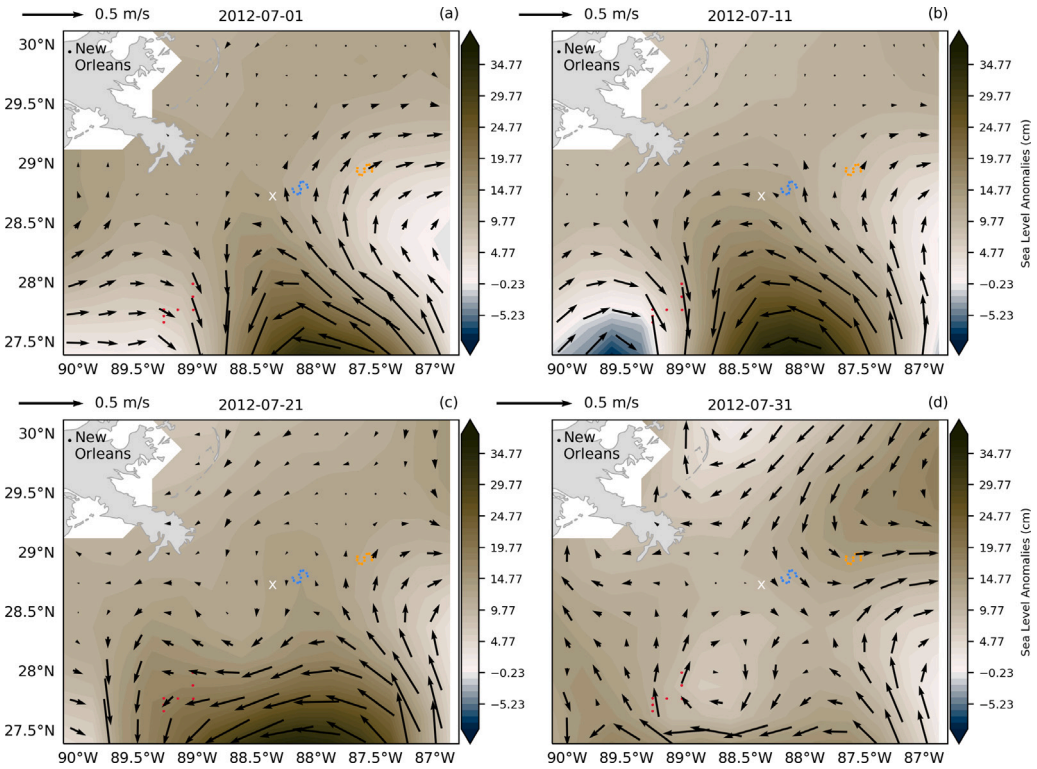


Fig. 3. Snapshots of geostrophic surface velocities (vectors, ms^{-1}) derived from satellite altimetry (colors, cm) throughout the first 30 days of drifter deployment. Deepwater horizon (white "x") and drifter deployment locations added for reference: *S*1, blue; *S*2, orange; *L*, red.

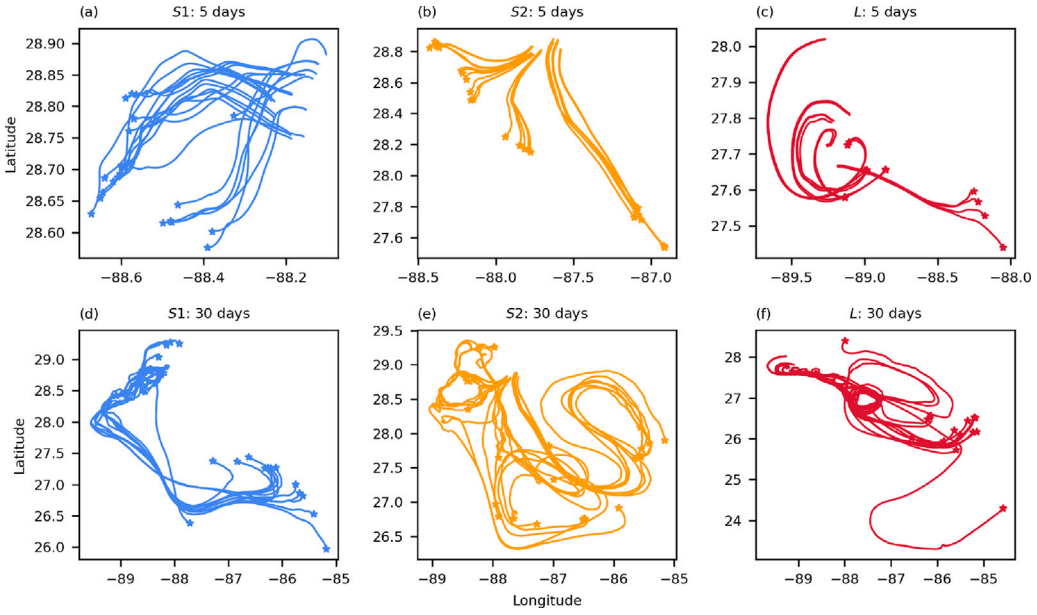


Fig. 4. Selected trajectories of *S*1, *S*2, and *L* drifters for the first five days (panels (a)–(c), respectively; *cf.* Fig. 2a and Fig. 3a) and the first 30 days (panels (d)–(f); *cf.* Fig. 2c and Fig. 3d). Inertial oscillations have been removed with a 24 h low pass filter. Dots indicate end locations.

FNNs (blue), ARIMA (orange dot-dashes), and persistence (red dashes) models for *S*1 and *S*2 drifters.

The FNN performed comparably to ARIMA on the *S*1 drifters with prediction error of ≈12 km at hour 24. Both models outperformed persistence due to the latter always being tangent to the inertial spiral trajectories (Fig. 7a). In contrast, the dynamically variable trajectories of *S*2 were harder for the FNN to learn (Fig. 7b). ARIMA performed best on these drifters while the FNN performed similarly to persistence but with smaller standard deviation (error bars) throughout the 24 h prediction window, with maximum mean RMSE ≈15 km. ARIMA's performance on these drifters can be understood conceptually by recalling that training the FNN sought to find an optimal regression for all examples, while separate ARIMA regressions were fit to each trajectory time series separately. This means that each ARIMA regression was completely independent of the others and represented a best-fit regression for any given time series, while the FNN was affected by every time series it was presented with during training. Thus, given the variability of *S*2 trajectories and the relatively small number of examples from

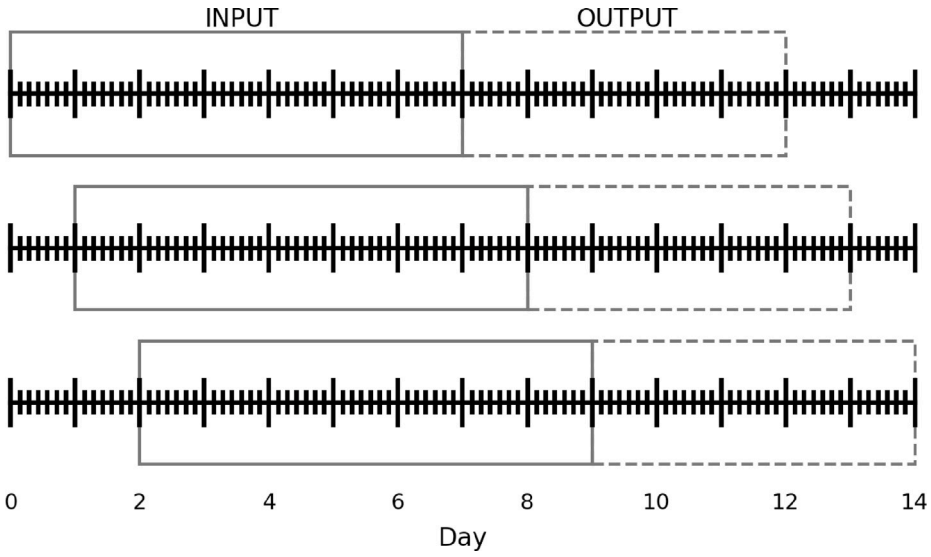


Fig. 5. Illustration of the rolling window implementation applied to the FNNs with 7 d input (solid box) and 5 d output (dashed box). Drifter data were upsampled to hourly observations (*e.g.*, small ticks) and the time series passed to the FNNs were upsampled again to daily increments (large ticks). See text for full explanation.

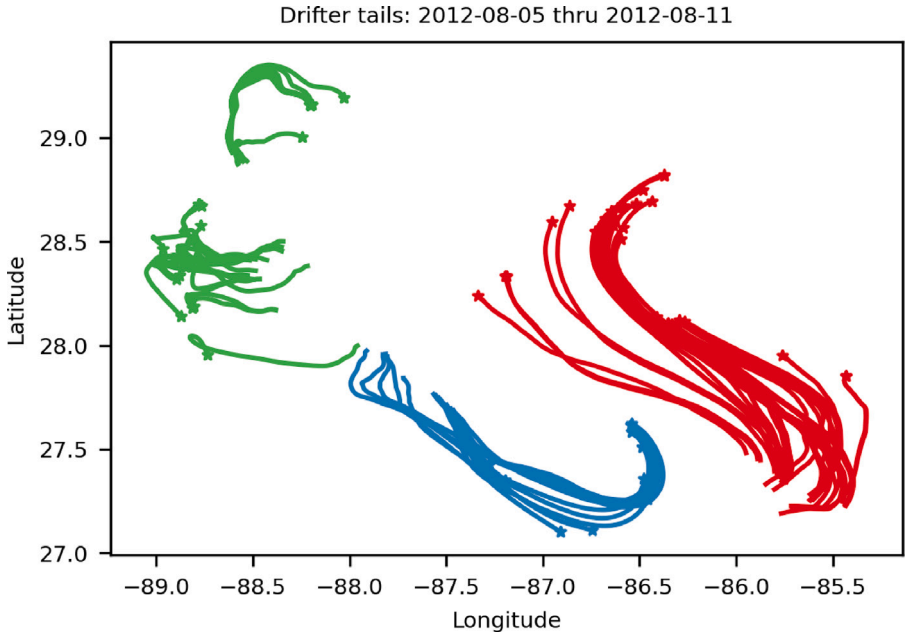


Fig. 6. Sample output of the hierarchical clustering algorithm applied to *S*2 drifters in the vicinity of a divergence zone. All drifters began in their own clusters and were then grouped according to spatial distances.

which to learn, the final FNN arguably underperformed on any given time series compared to the corresponding ARIMA model.

Fig. 8 summarizes the velocity and position prediction error (Eq. (2)) for the FNN trained on the GLAD drifters for a 5 d reconstruction window instead of 24 h, along with ARIMA and persistence models. The difference in performance between the three models was negligible, with all three producing position error ≈83 km on Day 5. We also note that the FNN performance on Day 1 was comparable to the *S*2 24-h FNN (Fig. 7b). This mean position error is in agreement with 5-day dispersion analyses of GLAD drifters (Poje et al., 2014; Haza et al., 2014; Huntley et al., 2019), but our error spread suggests the FNNs did not learn the velocity field structure well, if at all. Model error cascade caused ARIMA and persistence velocity errors to increase recursively with prediction day: error from Day 1 influenced the next prediction for Day 2 and so on. In contrast, the FNN was trained to predict all days simultaneously with minimal error; thus, velocity error was ≈26 m/s for Days 1–5.

Averaging Eq. (1) over I prediction times quantifies RMSE for each reconstruction and evaluates how the models performed over the three-month experiment:

$$\overline{\mathrm{RMSE}}_j = \frac{1}{I}\sum_{i=1}^{I} \mathrm{RMSE}_{ij} \tag{3}$$

This is shown in Fig. 9 for velocity and position with error bars indicating one standard deviation from the mean given by Eq. (3). Tropical Cyclone (TC) Isaac passed over the domain from 27–30 Aug 2012; this period is indicated by the gray shading. The reconstructions before the storm from all three models exhibited greater variability

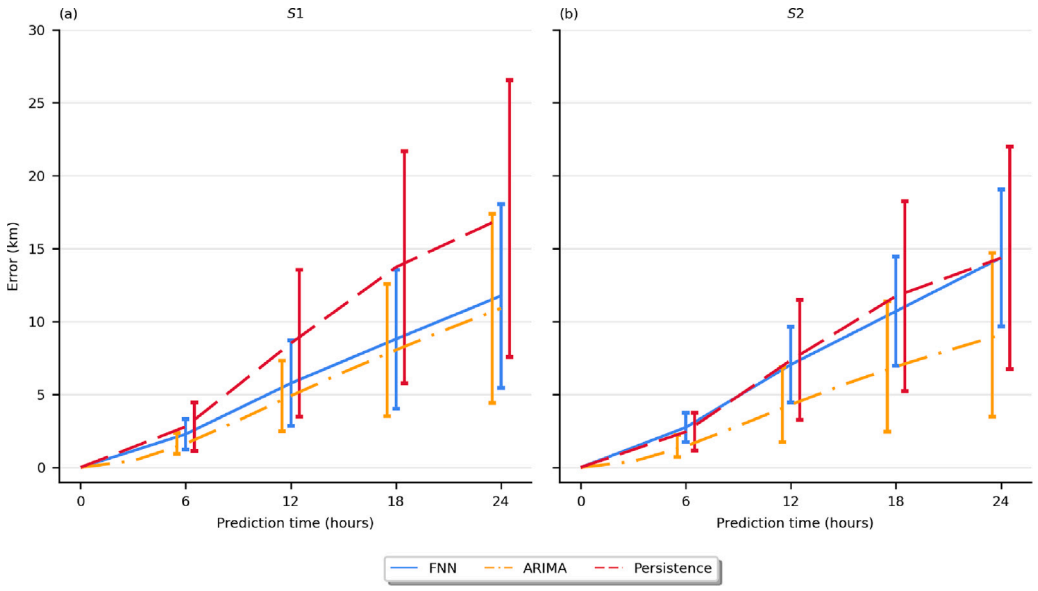


Fig. 7. Position error (km) for simple FNNs trained on GLAD *S*1 drifters (a) and *S*2 drifters (b). FNN error is shown in blue while ARIMA and persistence model errors are shown in orange dot-dashes and red dashes, respectively. Error bars indicate one standard deviation from the mean calculated using Eq. (2).

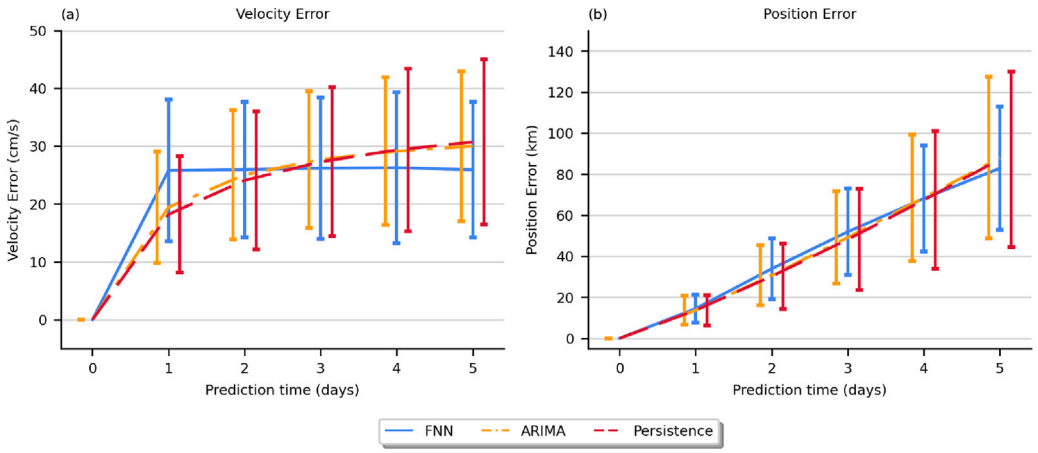


Fig. 8. Average velocity (a) and position (b) errors (cm/s and km, respectively) versus prediction time for 5 d predictions issued by a simple FNN (blue), ARIMA (orange dot-dashes), and persistence (red dashes) models. RMSE is calculated using Eq. (2) with error bars showing one standard deviation from the mean.

and larger mean error than after the storm; this is discussed further in Section 5.

4. Social spatio-temporal graph convolutional neural network (STN)

4.1. STN configuration

The STN code was downloaded from GitHub[2] under an open-source MIT license granting express permission to use and modify the software free of charge. The model consists of a spatio-temporal graph (STG) network that learns the spatial and temporal patterns in the data and a time-extrapolator (TXP) network that extends the time series to make predictions. Compared to the FNNs, the STN contained anywhere from 3 to 28 times the number of trainable parameters. A high-level overview of the STN is provided in Appendix A but the reader is referred to Mohamed et al. (2020) for illustrations and detailed descriptions of the model and its implementation. Optimal topologies of the STNs, determined by hyperparameter grid searches, are summarized in Table 3.

Input attribute vectors for each drifter were defined as $\mathbf{x} = [d(\text{lon}), d(\text{lat}), d^2(\text{lon}), d^2(\text{lat})]$ to help the model learn highly variable flows by including proxies for both velocity, $d(...)$, and acceleration, $d^2(...)$, where d indicates the difference between consecutive observations. (As with the original pedestrian model, dividing by change in time – dt and d^2t – was unnecessary given the constant time interval of the drifter data.) Qualitative inspections of histograms of all zonal and meridional velocities and accelerations for each deployment suggested that the distributions of velocity and acceleration were roughly Gaussian, with the exception of a more bimodal L meridional velocity distribution. Shapiro–Wilks tests for normality confirmed this assessment, indicating that predicting Gaussian distribution parameters was reasonable. We used the Adam optimization algorithm (Kingma and Ba, 2017) and trained the STN to learn the bivariate Gaussian probability density

[2] https://github.com/abduallahmohamed/Social-STGCNN, retrieved November 2020.

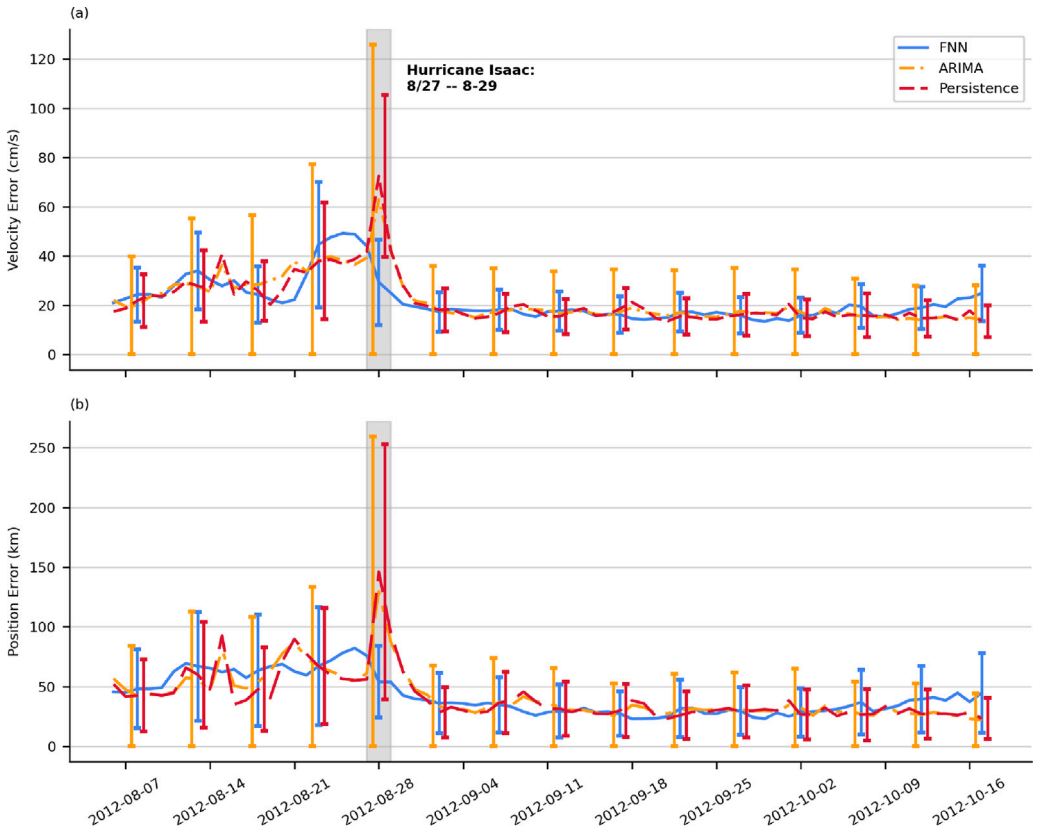


Fig. 9. Average velocity (a) and position (b) errors (cm/s and km, respectively) versus prediction initiation time for the simple FNN (blue), ARIMA (orange dot-dashes), and persistence (red dashes) models, averaged over the 5-day prediction window. RMSE is calculated using Eq. (3) with error bars showing one standard deviation from the mean. Error bars for ARIMA and persistence are offset slightly to prevent overlapping. The passage of Tropical Cyclone Isaac is shaded in gray.

function (PDF; see Wilks, 2006) given by:

$$f(x,y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 + \left(\frac{y-\mu_y}{\sigma_y}\right)^2 - 2\rho\left(\frac{x-\mu_x}{\sigma_x}\right)\left(\frac{y-\mu_y}{\sigma_y}\right)\right]\right] \tag{4}$$

where the five parameters $\mu_x, \mu_y, \sigma_x, \sigma_y, \rho$, corresponding to the means, standard deviations, and correlation between x and y, respectively. The STN used the previous 7 days of drifter velocity observations as input and generated 5 days of separate sets of PDF parameters for zonal and meridional velocity components of each drifter at each prediction time (Appendix A.)

We again employed the 3-fold cross-validation technique, this time reserving drifters from one full deployment ($S1$, $S2$, L) for validation while splitting the remaining trajectories so that the first 70% of the time series, corresponding to the first ≈2 months of the experiment, became the training data set while the last month of data became the test set. This setup reflects an operationally realistic scenario in which one would induce a ML model from existing data in hopes of using it on new drifter data from the region. While this is a common technique for learning time series, it presented unique challenges in this application that will be discussed in Section 5. The STNs were initialized using the first 7 days' worth of velocity and acceleration data from the training subset of drifters, upscaled to daily observations as discussed earlier. As before, the training set was updated every hour by replacing the oldest data with newer observations as they became available (Fig. 5), the models were further trained on the latest set of training examples, and reconstructions were created every midnight.

The drifter data were represented in the form of fully-connected and undirected mathematical graphs, G (Fig. 10):

$$G \equiv (V, E) \tag{5}$$

In Eq. (5), V contained time-stacked arrays of drifter observations and had dimensions $(N_f \times T \times N_d)$, where N_f is the length of the feature vector $\mathbf{x}$ (*e.g.*, velocity, acceleration), T is the number of time snapshots, and N_d is the number of drifters. Similarly, E contained time-stacked adjacency matrices quantifying the connectivity between each drifter over the same time window as V and had dimensions $(T \times N_d \times N_d)$. The connectivity between the ith and jth drifter at time t was given by:

$$e_t^{ij} = \begin{cases} \sum_{k=1}^{N_f} \frac{1}{\sqrt{(\mathbf{x}_t^{k,i} - \mathbf{x}_t^{k,j})^2}}, & (\mathbf{x}_t^{k,i} - \mathbf{x}_t^{k,j})^2 \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Thus, each example consisted of a pair of time-stack arrays containing information from every active drifter during the time window $[t_0, t_1, \ldots, T]$, where $T = 7$ for the input data and $T = 5$ for the output.

The passage of TC Isaac through the GLAD drifter array from 27–30 Aug 2012 as a tropical storm and Category 1 hurricane (Figs. 11 and 12) provided an opportunity to test whether the STN would benefit from knowing wind information as well as drifter behavior (*e.g.*, Curcic et al., 2016). We time- and space-matched each drifter observation with 0.25° Level 3 gridded 1-day surface (10 m) winds from the MetOp-A polar orbiting meteorological satellite retrieved from NOAA CoastWatch/OceanWatch[3] (Fig. 13). Missing data between successive

[3] https://coastwatch.pfeg.noaa.gov/erddap/griddap/erdQAwind1day_LonPM180.html

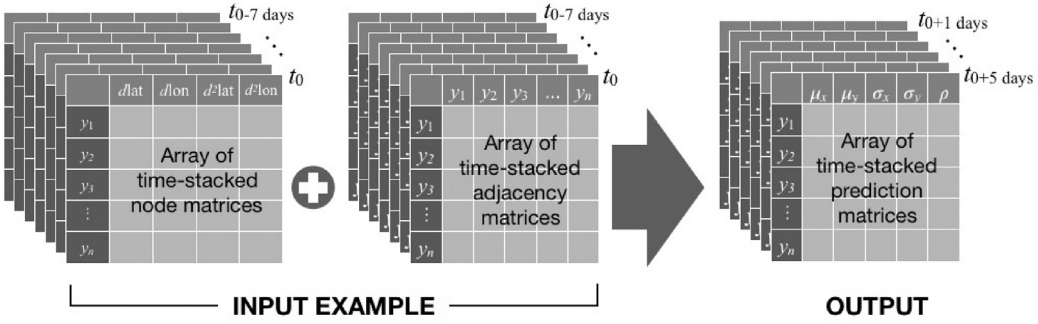


Fig. 10. Visualization of the STN: Input consists of an array of historical time-stacked node and adjacency matrices for drifters $y_1, y_2, \ldots, y_n$; output contains parameters for the bivariate Gaussian probability distribution function $\mu_{d\text{lon}}, \mu_{d\text{lat}}, \sigma_{d\text{lon}}, \sigma_{d\text{lat}}, \rho$. See Appendix A for more details.

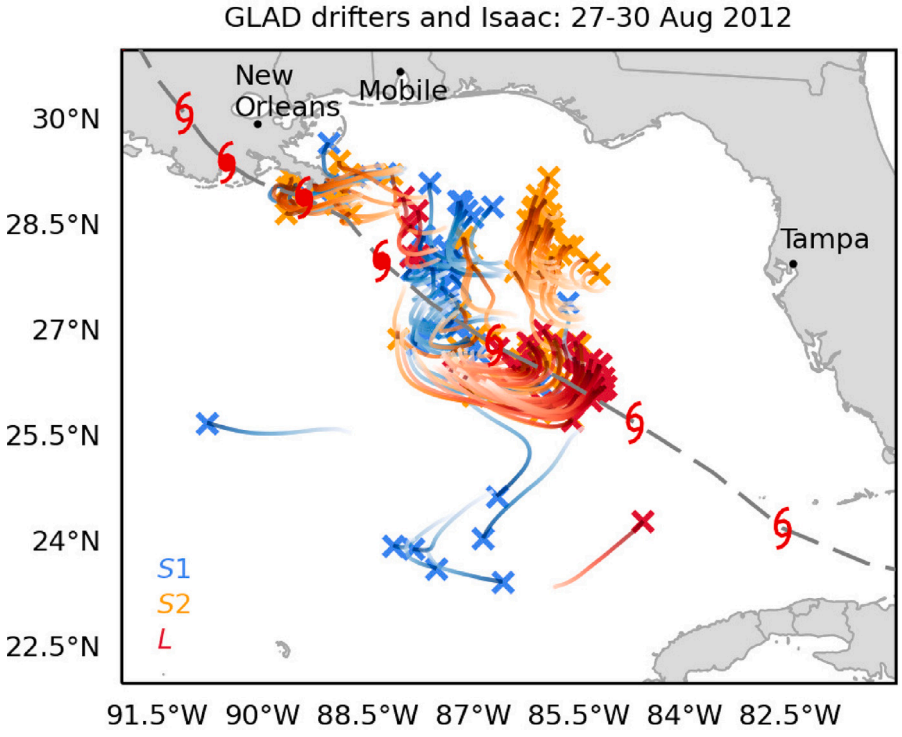


Fig. 11. Trajectory of Tropical Cyclone (TC) Isaac over the drifter array from 27–30 Aug 2012 as a tropical storm and Category 1 hurricane. Active drifters during this time are shown with an "×", tails indicating positions throughout this time window, and color-coded according to original deployment (*S*1: blue, *S*2: orange, *L*: red). TC symbols indicate Isaac's location every 12 h starting 00z 27 Aug 2012 just north of Cuba.

satellite swaths (*e.g.*, Fig. 12c) were filled in using simple linear interpolation. We then trained both 5-day STNs using the full GLAD time series with wind (input: [d(lon), d(lat), d^2(lon), d^2(lat), u_{wind}, v_{wind}]) and 24-h STNs using only data from 25–31 Aug.

Finally, we also trained STNs with two additional architectural modifications. First, the node array was expanded in time to include the last three time steps instead of only one. For example, the first layer in the time stack (the first block in Fig. 10) included [$d(\text{lon})_0$, $d(\text{lon})_{-1}$, $d(\text{lon})_{-2}$, ...], the next time layer [$d(\text{lon})_{-1}$, $d(\text{lon})_{-2}$, $d(\text{lon})_{-3}$, ...], and so forth. This intentional redundancy was introduced to test for any benefit of having time information in the spatial dimension. The second modification extended the similarity metric from Eq. (6) to also quantify geospatial distance and relative trajectory angle θ between the ith and jth drifter:

$$e_t^{ij} = \sum_{k=1}^{N_f} \frac{1}{\sqrt{(\mathbf{x}_t^{k,i} - \mathbf{x}_t^{k,j})^2}} + \frac{1}{\theta^{i,j}} \tag{7}$$

where the attribute vector $\mathbf{x}$ now also included latitude and longitude, and

$$\theta = cos^{-1} \frac{\hat{\mathbf{x}}^i \cdot \hat{\mathbf{x}}^j}{|\hat{\mathbf{x}}^i||\hat{\mathbf{x}}^j|} \tag{8}$$

where $(\cdot)$ indicates the dot product, $|\hat{\mathbf{x}}|$ is the magnitude of vector $\hat{\mathbf{x}}$, and $\hat{\mathbf{x}} = [d(\text{lon}), d(\text{lat})]$. We refer to these as "advanced configuration". The units of e_t^{ij} are non-physical; the idea is to quantify, in a single term, the connectivity between drifters according to multiple physical intuitions or known relationships. As in Eq. (6), if the denominator of either term in Eq. (7) is zero, that term was set to zero.

To summarize, the STN differed from the FNN in a number of fundamental ways. The most important difference is the algorithmic architecture itself, summarized side-by-side in Table 2. Following from this is the number of trainable parameters, which is directly related to the complexity of the model. The FNN had about 500–1000 trainable parameters, while the STN had 3 to 28 times that many, depending on the model configuration (Table 2). In general, the greater the number of parameters, the larger the training data set that is needed to train the model. Finally, the two models differed in their inputs and outputs.

The input to the STN consisted of two arrays, one containing the observations from all drifters over time and the other containing the time-stacked adjacency matrices (Fig. 10). The input to the FNN, on the other hand, was a vector containing the time series of observations from a single drifter. Likewise, the STN produced an array of time-stacked predictions for each drifter at multiple time steps, while the FNN produced a vector for one drifter a time. The FNN directly predicted zonal and meridional velocities, while the SNN returned the bivariate Gaussian probability distributions from which velocities were sampled and trajectories were derived.

4.2. Results: Social spatio-temporal graph convolutional neural networks

Figs. 14 and 15 summarize the predictive performance for each of the STNs along with ARIMA and the FNN from Figs. 8 and 9. Subplots Fig. 14a–c show models from the 3-fold cross validation with the subplot title indicating the deployment that was reserved for testing. The final subplot, which shows the average across all three models, provides an assessment of how well the network performed on the entire data set. Note that, strictly speaking, the STNs with wind are not directly comparable to the FNNs, since no FNNs received wind as input.

All neural networks, including the FNNs, performed similarly on the *S*1 drifters. The best architecture for these drifters was the advanced STN (teal dots) which, at Day 5, had an average error about 75% that of ARIMA (≈72 km versus 102 km) and with considerably smaller standard deviations. Both the no-wind (purple dashes) and advanced STNs performed comparably on *S*2 drifters, with Day 5 RMSE averaging ≈55 km compared to ≈80 km for both ARIMA (orange dot-dashes) and the FNN (blue). On the *L* drifters, all three STN configurations had an average Day 5 RMSE ≈60 km, while ARIMA and the FNN models both averaged ≈80 km on Day 5. Overall, the advanced STN performed the best averaged across all drifters, implying that the added temporal information was beneficial. Although the mean RMSE was nearly identical to that of the no-wind model, the spread (standard deviation) was slightly smaller. Adding wind to the model helped in some cases, such as with *L* drifters, but not enough to make this network stand out among the others, especially considering the increased input data requirements. This result can be partially attributed to the fact that atmospheric scales are often larger than oceanic scales.

STN RMSE hovered around 40 km during the first month of the experiment and around 30 km thereafter (Fig. 15). Prediction variability, as indicated by the standard deviation, also remained fairly constant, with spikes around 15 Aug and 28 Aug as TC Isaac passed through. There is a notable predictability regime shift after the storm passage: prior to the storm, all STN configurations consistently outperformed both ARIMA and the FNN both on average and standard deviations. After the storm, all models performed similarly on average. To better assess any modeling impacts of including wind as a predictor, we trained new STNs that, like the previous FNNs, predicted 24 h out using the last 24 h as input. These were trained only on data from the seven day window 25–31 Aug, which included the passage of Isaac (Fig. 16). Separating these into cross validation test sets as before shows how different each deployment was. Adding wind as input (blue dashes) helped with *S*1 and *L*, at least for the latter half of the prediction window, but slightly hindered performance on *S*2. These STNs had larger mean errors, but smaller standard deviations, than ARIMA (orange dot-dashes), with the exception of the last few hours for *S*1. ARIMA outperformed all networks on the *L* drifters this time, with RMSE at hour 24 around 25 km compared to 45 km for the no-wind network (purple dashes) and 40 km with wind.

Lastly, nondimensional Pattern Correlation Coefficients (PCC) were computed for each reconstruction to compare the predictive capability of the STN, ARIMA, and persistence models to the predictability limit of the data set (Robinson et al., 2002):

$$\mathrm{PCC} = \frac{\left(\hat{V} - V^b\right)^T \left(V^t - V^b\right)}{\|\hat{V} - V^b\|_2 \|V^t - V^b\|_2}, V = (u^2 + v^2)^{1/2} \tag{9}$$

where $\hat{V}$ is the model reconstructed velocity, V^b is a background flow (here, geostrophic surface velocity presented in Section 2), V^t are the actual observed values (targets), T is the transpose operator, and $\|\cdot\|_2$ indicates the L_2 norm. A value of $PCC = 1$ indicates a perfect prediction. Summations are taken over all drifters and prediction times, as in the RMSE calculations. Fig. 17 shows that $\mathrm{PCC}_{\mathrm{GNN}}$ averaged 0.6–0.7 for most of the experiment, with greatest fluctuation early on. By this metric, persistence outperformed the STN towards the beginning – perhaps because the persistent velocity is not bad initially – and again towards the end of the experiment, but the STN outperformed ARIMA consistently after the first two weeks.

5. Discussion

Lagrangian drifter deployments in the northern GoM during the GLAD experiment provided an opportunity to test the ability of FNNs used by G20 to predict observed ocean trajectories. The FNNs performed best with drifters whose trajectories captured inertial oscillations, such as the first couple of weeks of *S*1, but given the regularity of inertial signals, ARIMA performed just as well in these cases. In the absence of inertial oscillations, whether due to stronger surface forcing, such as in *S*2, or when filtered out of the trajectories, the FNNs performed no better than persistence. The FNNs also offered no improvement whatsoever over persistence or ARIMA when configured to make more useful 5-day predictions (Fig. 8). The longer ARIMA predictions tended to become unstable and converge to the time series mean due to the high degree of variability; thus, since the FNNs performed similarly, the best that can be said of the simple FNNs is that they often learned nothing more than the mean flow. They were clearly unable to learn anything meaningful about observed surface dynamics (Figs. 2 and 3).

While G20 intentionally sought the simplest neural network possible to explore using ML for predicting realistic ocean trajectories, the dismal performance on real-ocean scenarios can be attributed to their simplicity, particularly their inability to remember previous time series states or to take into account surrounding information from the drifter's "neighborhood", since they only saw one drifter at a time. Far more sophisticated STNs attempted to address these limitations and explore whether incorporating physical intuition and domain expertise into a ML model might offer more potential (*e.g.*, Faghmous et al., 2014). These models treated all deployed drifters at any given time as a single example and used an adjacency matrix to exchange information between the drifters. This allowed for objectively grouping drifters by similar behavior without needing clustering routines. Overall, these networks performed somewhat better than the previous FNNs and even slightly better than ARIMA in some cases. The original adjacency matrix only calculated Euclidean distances between $d(\mathrm{lon})$ and $d(\mathrm{lat})$ for each drifter. We tried adding two additional terms: the actual spatial distance between two drifters (since drifters might behave similarly but be far apart) and relative angle between the drifters (drifters moving in parallel are more similar than orthogonal trajectories.) This modification, along with introducing temporal information into the spatial feature array, performed well overall but not appreciably better than the original configuration (see Fig. 14). Nevertheless, the successes of the models compared to the FNNs are rooted in using these distances to weigh how much information should be shared between two or more drifters.

Introducing wind as an additional attribute had less of an impact overall than one might expect, but drogues are specifically designed to decouple the drifter's motion from overlying wind forcing to allow the drifter to move with the prevailing 1 m ocean current. Both test tank (Davis, 1985) and field (Poulain and Gerin, 2019; Poulain et al., 2022) experiments have found the CODE drifter to be minimally affected by winds less than 15 m s^{-1}. Thus, the characteristically calm GoM summer winds that averaged <5 m s^{-1} throughout most of the GLAD experiment were arguably too weak to impact the motion

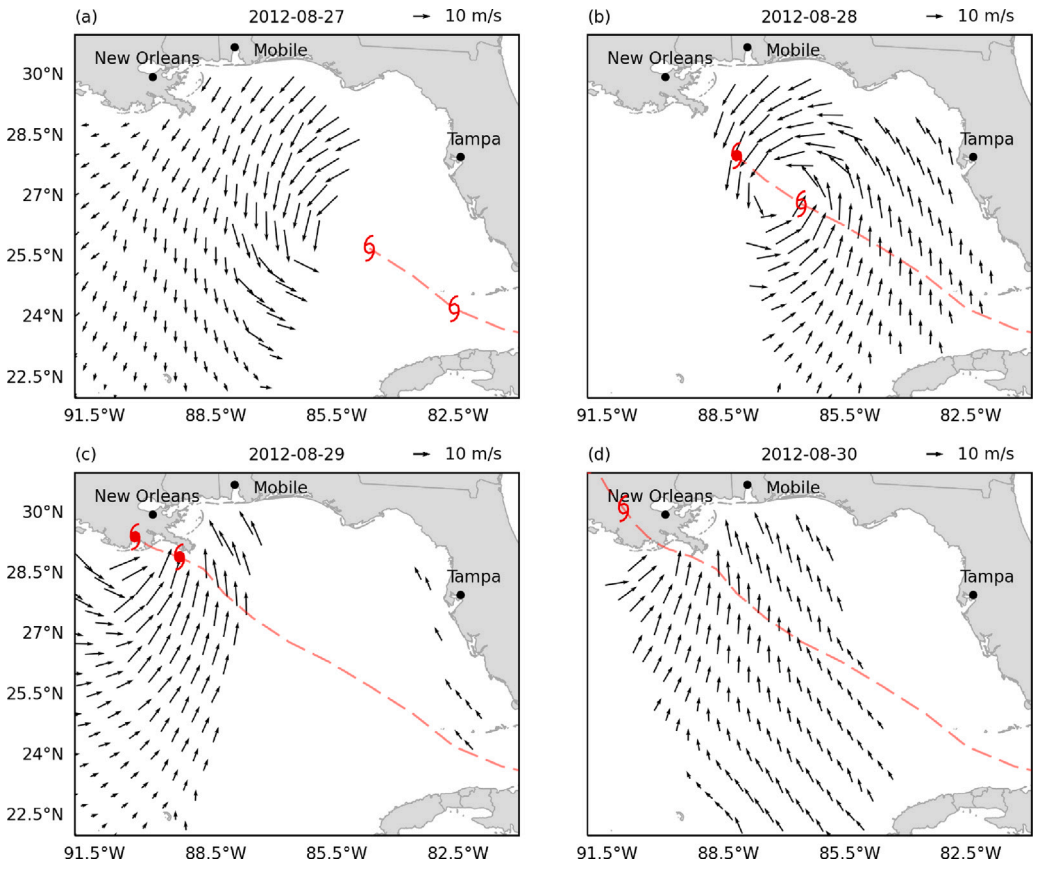


Fig. 12. Daily MetOp-A surface winds (10 m) before spatial interpolation during the passage of TC Isaac from 27–30 Aug 2012. TC symbols indicate the storm's location twice a day (hours 00z and 12z). Compare these subplots to Fig. 11 for the location of the drifter array relative to the observation swaths.

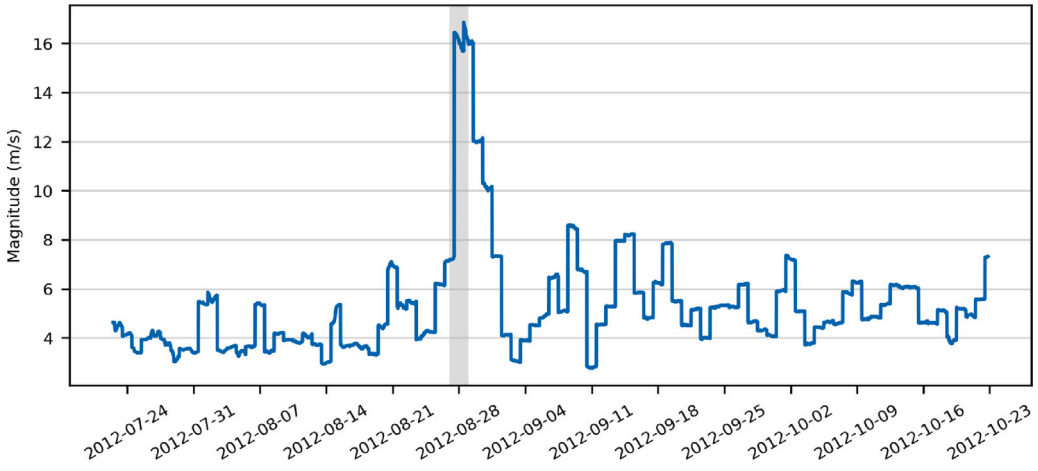


Fig. 13. Daily average of wind magnitude (m/s) over the entire drifter array throughout the GLAD experiment with the passage of TC Isaac highlighted in gray.

of these 1-m drogued drifters. This would imply that the zonal and meridional wind velocities, having little to no direct relationship to the direction of drifter motion, were irrelevant predictive attributes for most of the experiment. Yet, this decoupling has also been shown to break down during high-wind events (Lodise et al., 2019), which may explain the slightly improved 24 h STN reconstructions during TC Isaac for the *S*1 and *L* drifters, but more analysis is needed to conclude this definitively. It is also important to note that changing the input time series length from 7 days to 24 h increased the number of trainable parameters by an order of magnitude (Table 3), making it substantially more challenging to train these networks. In general, the more trainable parameters in a neural network, the greater the number of training examples needed to train it. The inability for us to obtain more data likely impacted the trainability of these 24-h models.

While the neural networks did not benefit from receiving wind as input, winds may have indirectly impacted the overall predictability of the trajectories by advecting most drifters offshore where submesoscale dynamics are less dominant, causing the trajectories to collectively exhibit less variability than on the continental shelf. Fig. 9 shows that all three reconstructions leading up to and during the storm exhibited larger mean error than after the storm. Noting that ARIMA errors were similar in magnitude at the very beginning of the experiment (*e.g.*, 7 Aug in Fig. 9) to post-storm (1 Sep onward), we hypothesize that the wind modified trajectories enough to impact their predictability.

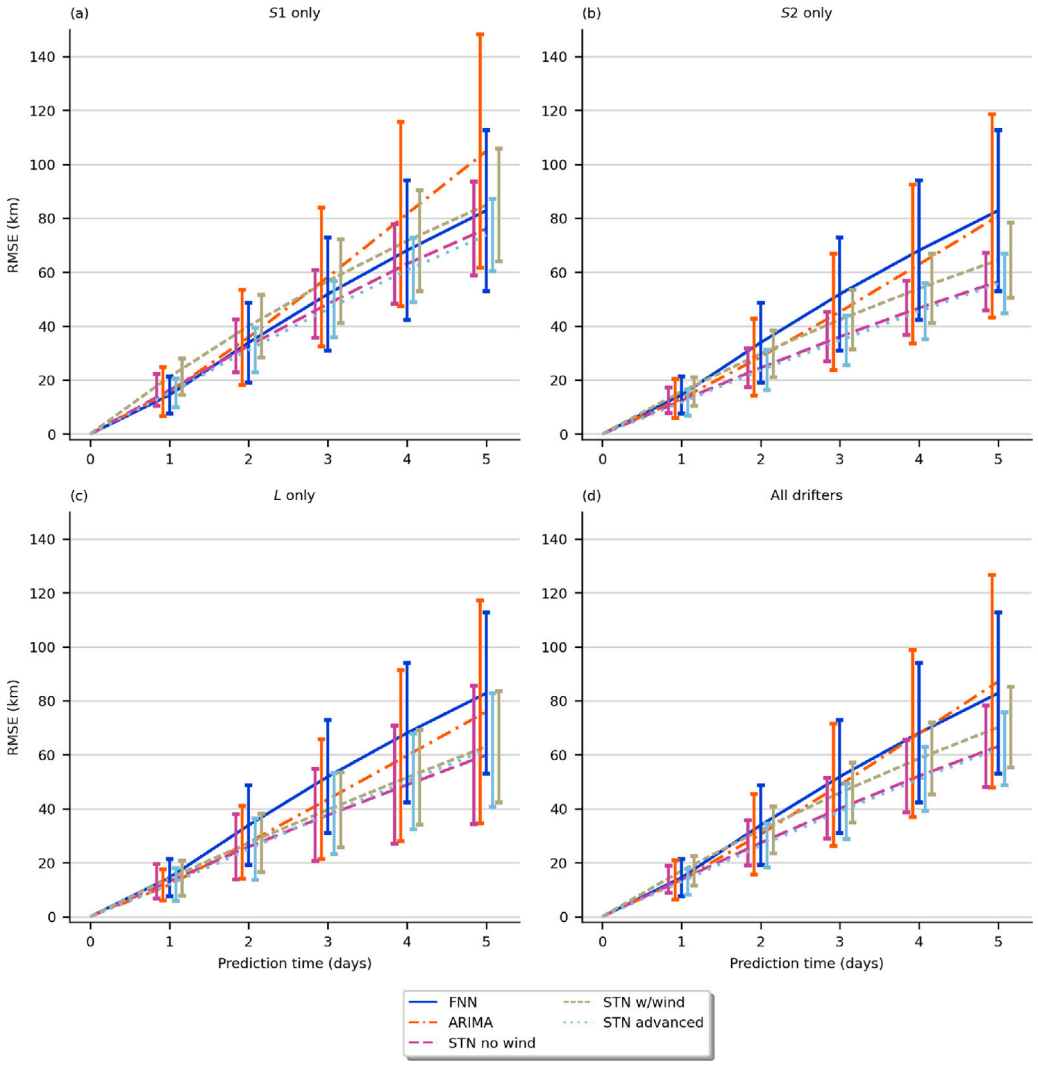


Fig. 14. Average position error (km) versus prediction time for all social spatio-temporal graph convolutional neural networks, simple fully connected neural network, and ARIMA models. RMSE is calculated using Eq. (2) with error bars showing one standard deviation from the mean. During 3-fold cross validation, one deployment was set aside for testing. Panels (a)–(c) show the results of these three models, while (d) shows the average over all test sets.

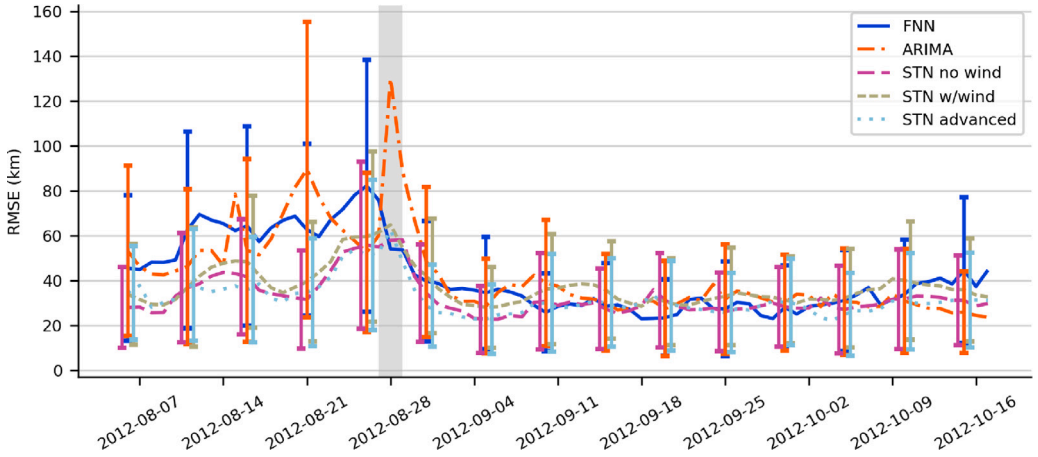


Fig. 15. Average position error (km) versus prediction initiation time for all social spatio-temporal graph convolutional neural networks, simple fully connected neural network, and ARIMA models. RMSE is calculated using Eq. (3) with error bars showing one standard deviation from the mean. The passage of TC Isaac is shaded in gray.

The FNNs' comparatively larger error during this early period can be attributed to the minimal training they had undergone and to the fact that, by their very nature, the FNNs served as single best-fit regressions for all trajectories, while ARIMA regressed each trajectory independently. We also note that average wind speeds over the experimental domain increased from ≈3 m/s during July and early August before

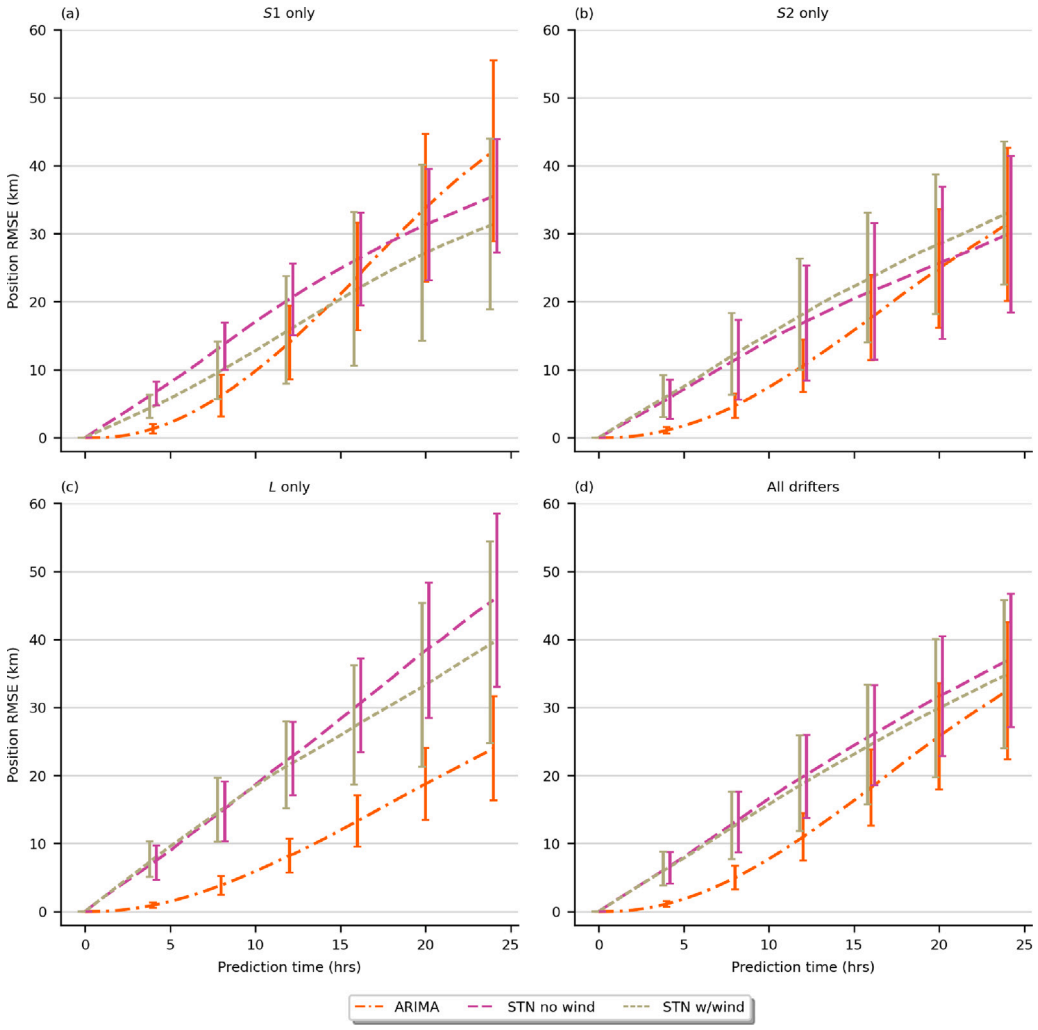


Fig. 16. Average position error (km) versus prediction time for social spatio-temporal graph convolutional neural networks trained only on data from 25–31 Aug 2012, with and without wind included as input. ARIMA model is included in orange. RMSE is calculated using Eq. (2) with error bars showing one standard deviation from the mean. During 3-fold cross validation, one deployment was set aside for testing. Panels (a)–(c) show the results of these three models, while (d) shows the average over all test sets.

the storm to 5–6 m/s during the months of September and October (Fig. 13). Further investigation is needed to determine the extent to which the drifters were impacted by winds during the experiment, but the lack of high resolution observational data over the GoM makes this challenging.

Little spatial variation of RMSE was observed throughout the experiment, but predictions from the beginning of the experiment showed larger spread of prediction error overall. A possible explanation for these larger error spreads initially is the greater variability of submesoscale surface dynamics that were sampled while the drifters were closer together, but this is difficult to determine from our setup, since scales of motion cannot be discerned from velocity PDFs alone. It can be said, however, that submesoscale dynamics are harder to learn than mesoscale and the data set contained far fewer examples of submesoscale because the drifters dispersed within days to weeks. Error distributions for all prediction days show that the variability increased throughout the prediction window as the predicted and actual trajectories diverged (Fig. 18). The largest change in distributions was observed between prediction days 1 and 2, while days 4 and 5 were most similar, indicating that the later prediction times contributed most to the error spread seen in Figs. 14 and 15.

Though the simultaneous deployment of hundreds of drifters in an experiment like GLAD was unprecedented in oceanography, the number of examples this data set produced remains very small by ML standards. Both *S*1 and *S*2 contained 90 drifters, but the bifurcation events of *S*2 caused the same number of drifters to sample larger dynamic variability, a reality that is not uncommon in oceanography. Assuming all drifters remained online throughout the entire 90 day experiment, utilizing the full 15 min sampling frequency, and taking into account that each example required 12 days of observations, less than 7500 examples were available for the STN, for which every moment in time constituted a single example. In reality, GPS battery failure caused drifters to drop out permanently at various times during the experiment while the drifter array simultaneously dispersed throughout the GoM basin capturing increasingly diverse and ever-evolving surface dynamics (Fig. 19a). Drifter density – calculated by dividing the number of active drifters by the area of the convex hull enclosing the cluster of drifters at a given time – started at just over 300 drifters per $100\,\mathrm{km}^2$ and decreased to 2 drifters per $100\,\mathrm{km}^2$ by the end of the experiment. A linear regression analysis indicated no statistical relationship between RMSE and drifter density due to dispersion ($r^2 \approx 0$, Fig. 19b.)

The method by which we divided the data into training, testing, and validation subsets for the STNs (as in Fig. 5) created unique challenges for this task. Splitting time series into training and testing sets is customary in ML, but the drifter time series continuously evolved over this time period. While we could have included observations from the entire time series in our training set to better capture the full time variability, this approach is difficult to justify with operational applications in mind. These trajectories by the third month were vastly different than the first month, primarily because the scales of motion

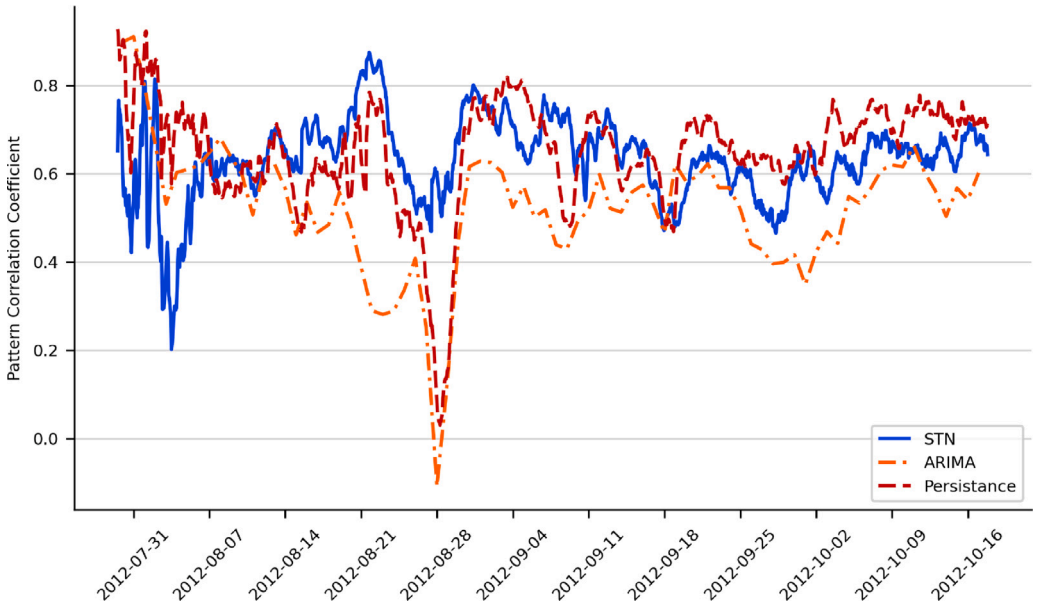


Fig. 17. Pattern Correlation Coefficient (PCC, dimensionless) comparing the predictive capability of the STN (green), ARIMA (orange), and persistence (maroon) prediction models to the predictability limit of the problem based on the model predicted velocity, background geostrophic flow velocity, and ground truth information. Summation in Eq. (9) is over all drifters and prediction times. PCC = 1 indicates an ideal scenario.

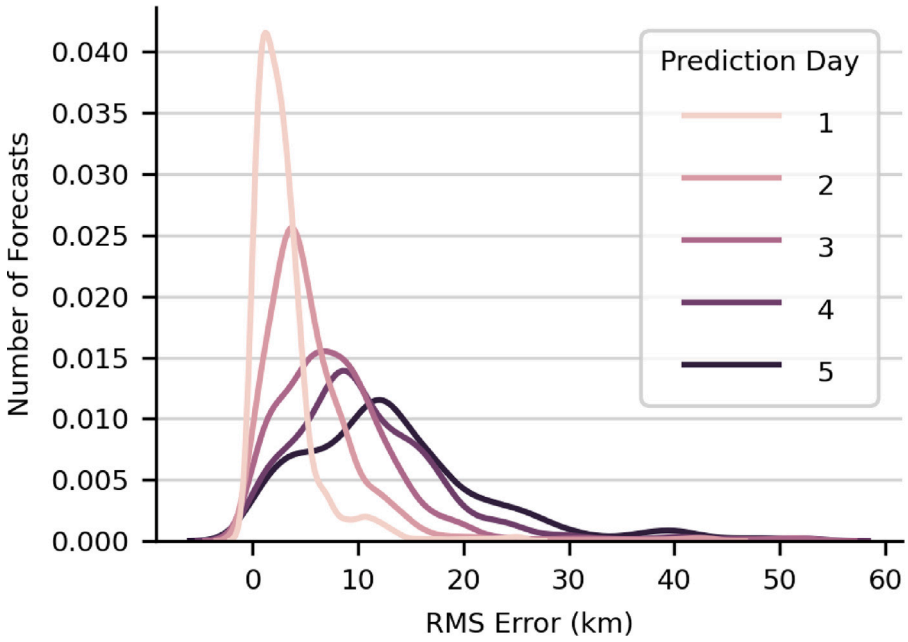


Fig. 18. RMSE distributions from the STN for prediction days 1–5. Distributions include all drifters and reconstructions.

being sampled changed in time as the drifters dispersed from being within a few kilometers of each other to covering the entire eastern GoM. Considerations like this are crucial for engineering ML models for chaotic ocean applications. The reconstructions overall might be described qualitatively as "hit-or-miss", with examples of both good and bad predictions existing throughout the experiment. STN performance as a function of training time reached an asymptote given the available data. We argue that, like all ML architectures, it would benefit from additional training data.

We structured our experimental setup with operational applications in mind. The ongoing rolling window training process, for example, provides a form of domain adaptation that can be invaluable in constantly changing domains like ocean surface dynamics. Possible applications include the need to monitor and predict the spread of spilled oil in a marine environment. Deploying thousands of drifters throughout a geographic area remains nearly impossible in normal conditions, let alone alongside first responders in an emergency response. Our method allows for the localized deployment of as few as 200 drifters (though more is always better) among which a STN can reconstruct additional tracer trajectories. While one would need to acquire 7 days of data before 5 days could be reconstructed, this could be overcome by using shorter term models to predict hours in the interim. While we evaluated our models by comparing single predicted trajectories for each drifter to the real trajectories, any number of possible trajectories can be generated from the PDF output for a given drifter in an operational setting if one is interested in ensemble predictions. Since the model outputs standard deviation, one also benefits from having an estimate of uncertainty for each drifter and for any given trajectory prediction.

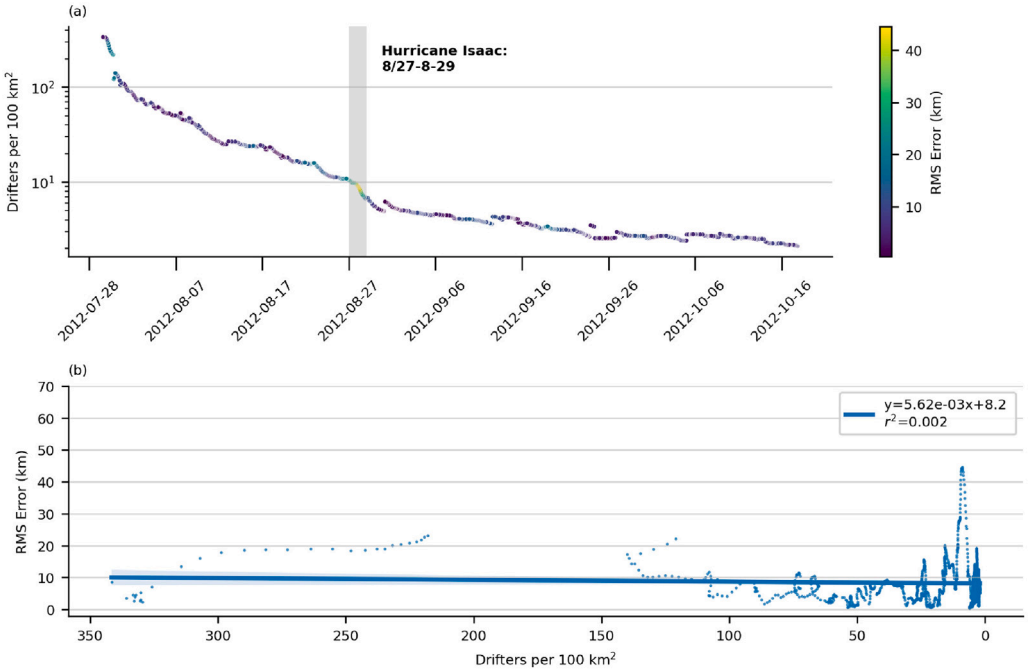


Fig. 19. Number of active drifters per geospatial area (drifters per 100 km^2) throughout the GLAD experiment colored by mean RMSE (km; panel a) and RMSE versus drifters per area (b) with r-squared regression showing this relationship is not statistically significant ($r^2 \approx 0$). The passage of Tropical Cyclone Isaac is marked in gray. The x-axis in (b) is reversed to correspond more closely to the trend over time.

This information could be incorporated into predicted trajectories by including degrees of certainty based on the model-produced standard deviations. Alternatively, instead of producing line trajectories, one could create probability clouds that show the spread of possible scenarios, which may be more useful in forecasting the dispersion of spilled oil or harmful algal blooms. For more analytical uncertainty analyses, one could integrate clustering capabilities (*e.g.* Dutt et al., 2018; Haley et al., 2023; Lermusiaux et al., 2024) or conformal prediction techniques (Huang et al., 2023a; Zargarbashi et al., 2023) with our GNNs so that predictions are accompanied by uncertainty estimates.

Using STNs for true temporal forecasting (as opposed to our reconstructions over the same time window as the training trajectories) would require more data and, most likely, training techniques such as transfer learning, where models are pre-trained on a big dataset and then fine-tuned with data from the specific problem of interest. One might, for example, pre-train our STN on historical trajectories from NOAA's Global Drifter Array program and then fine-tune it on local drifters such as those from the GLAD experiment. Whether this approach would create a model capable of forecasting new trajectories is an area of future work. Related to this is the ability to use a trained STN in another oceanic region. This, too, is an ideal use case for transfer learning.

Finally, we comment on the possibility of interpreting the decision-making process of the STN. While CNNs are among the most interpretable of neural networks – one can visualize the internal convolutions and discover regions of high or low activation throughout the network – GNNs, like most neural networks, including the TXP component of our model, remain much more difficult to interpret due to the many nonlinear transformations that the data undergo within the networks. And while our GCNN treats graphs like images, mathematical graphs do not have features like shapes, object edges (in the artistic sense of the term), or color patterns that humans gravitate to when interpreting images. Nevertheless, one can still visualize the internal convolutions of the STG component of our model using techniques such as graph kernel analysis (Feng et al., 2022) and neuron analysis (Xuanyuan et al., 2023) or tools like GNNExplainer (Ying et al., 2019) or GraphLIME (Huang et al., 2023b). All of these approaches seek to identify subgraphs of importance, or local areas of activation within the graph. Neural network interpretation remains an area of active research and is beyond the scope of this paper. Model interpretation of this nature is left to future work.

6. Conclusions

We have built upon previous work by G20 who developed simple fully connected neural networks to learn oceanic particle trajectories in a variety of representative flows. The authors found that their time series FNNs produced prediction errors that were nearly half those of conventional ARIMA models for particles in realistic flows, but the flow field was generated by an ocean circulation model and the study did not include any observed trajectories. Here we tested the ability of these same FNNs to predict observed oceanic trajectories using an array of Lagrangian drifters released in the Gulf of Mexico in 2012. The GLAD experiment was separated into three separate deployments, each capturing distinct surface dynamics.

We found that FNNs did not outperform ARIMA in this application. In the best case scenario (*S*1), the average error was the same as that of ARIMA. FNNs trained to issue 5 d reconstructions, which are often more useful in operational oceanography, performed even worse than the 24 h networks, with RMSE being the same as both ARIMA and persistence models, despite introducing into the pipeline an unsupervised clustering algorithm to group drifters spatially and inducing separate FNNs for each cluster. The idea was to facilitate learning by restricting the range of dynamics each FNN needed to learn. The results of this first part implied that single-layer FNNs were too simplistic to learn real ocean trajectories that often exhibit chaotic behavior.

We then considered a more sophisticated social spatio-temporal graph convolutional neural network, originally developed by Mohamed et al. (2020) for predicting pedestrian trajectories in social scenes. The STN had two main advantages over the simple FNNs that helped it outperform both ARIMA and the FNN in many (but not all) cases. First, it learned patterns in both time and space, combined these patterns into a single feature embedding, and then extrapolated into the future. Second, it shared information between drifters according to how similarly

pairs of drifters behaved. We also tested sensitivity to including wind as input and found that wind helped most with $S1$ drifters and to a lesser extent with the L deployment drifters when predicting on hourly timescales but did not help at all with predicting on daily timescales. This study showed that incorporating known physical relationships and connectivity into ML architectures can provide improvement over simple FNNs for predicting ocean trajectories, but the problem remains an open challenge.

Future directions include expanding the data set to incorporate drifter data from other localized experiments, which may allow the use of attention to learn the adjacency matrix relationships rather that predefining them. This could also investigate the transferability of this model to different geographic regions and times of year. Transfer learning could be applied in a couple ways: our STN trained in the GoM on GLAD drifters could be tailored to a new region by fine-tuning it with drifter data local to that region; alternatively, a STN could be pre-trained on a much larger global data set, such as the NOAA Global Drifter Program, and then transferred to and fine-tuned in one or more local regions of interest. A spatio-temporal statistical analysis of trajectories predicted by the trained neural networks would also be valuable, as this would highlight how trajectories vary in time and space, and if their physical and statistical properties are oceanographically plausible. Extending the networks to provide uncertainty estimates and clustering capabilities would also be very useful (*e.g.*, Dutt et al., 2018; Haley et al., 2023; Huang et al., 2023a; Zargarbashi et al., 2023; Lermusiaux et al., 2024), especially for practical applications and decision making. Related to this is the topic of model interpretability, which is of great interest to applied machine learning. Techniques and tools for machine learning model interpretability could be applied for the STN to attempt to discern and facilitate its decision-making process (*e.g.*, Gupta and Lermusiaux, 2023). Other types of models such as recurrent neural networks or long short-term memory networks may help with remembering time series states without introducing as many trainable parameters as the STN. These may prove more suitable given the amount of training data available. Finally, the impact of wind warrants further exploration by comparing predictive performance on drogued versus undrogued drifters.

CRediT authorship contribution statement

Matthew D. Grossi: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Stefanie Jegelka:** Writing – review & editing, Validation, Methodology, Conceptualization. **Pierre F.J. Lermusiaux:** Writing – review & editing, Validation, Project administration, Methodology, Funding acquisition, Conceptualization. **Tamay M. Özgökmen:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Methodology, Funding acquisition, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We are grateful to the Office of Naval Research for partial support under grant N00014-20-1-2023 (MURI ML-SCOPE) to the Massachusetts Institute of Technology and the University of Miami. Data are publicly available through the Gulf of Mexico Research Initiative Information and Data Cooperative at https://data.gulfresearchinitiative.org (doi:R1.x134.073:0004) unless otherwise noted.

Appendix A. The social spatio-temporal graph convolutional neural network (STN): a description

Much of the following can be found in any introductory textbook on machine learning methods that includes a discussion on convolutional neural networks. Specific details about the STN are from Mohamed et al. (2020), to which the reader is referred for more information. Other relevant sources are cited where appropriate.

The STN model has three tasks: (1) learn spatial patterns from a cluster of drifters, (2) learn patterns in the observed trajectory time series of these drifters, and (3) make predictions for where these drifters will move next. These are accomplished using two convolutional neural networks (CNNs), the first of which is designed to learn the spatial and temporal behaviors, and the second to issue forecasts. Mohamed et al. (2020) aptly call these the spatiotemporal and time-extrapolator CNNs, respectively, and illustrate them in their Fig. 2. The spatiotemporal CNN maps input observations to desired output features and then combines stacked time snapshots of observed drifter locations with a calculated drifter "connectivity" based on behavioral similarities between all combinations of drifters. The second CNN then operates on the time dimension of the feature embedding produced by the spatiotemporal CNN to generate forecasts for each drifter. We now break apart this summary to explain each step in more detail by working backwards starting with a description of a convolutional neural network, then a graph neural network, and finally the social spatiotemporal component. Details about our specific implementation are included throughout.

A.1. Convolutional neural network (CNN)

CNNs originate in the field of computer vision and are well-suited for learning multi-dimensional data. Consider a two-dimensional image I of size $(h \times w)$, represented by a matrix of pixel values corresponding to grayscale intensities, and let this be the input to a CNN. Like with any neural network, CNNs can have one or many layers; let each layer be denoted by bracketed superscript l. A convolutional layer contains one or more square matrices W, called kernels, of dimensions $(f^{[l]} \times f^{[l]})$, where customarily $f^{[l]} < h^{[l-1]}, w^{[l-1]}$. The elements of W are trainable parameters initialized to small random numbers. "Trainable" here means the parameters are adjusted throughout the training process as the algorithm converges on the optimal values, as in any regression process. The model convolves the input array with the kernel by sliding the kernel over the image and applying a nonlinear transformation g:

$$z^{[l]} = \sum_{i=1}^{h_f} \sum_{j=1}^{h_w} W_{i,j}^{[l]} * x_{i,j}^{[l-1]} + b^{[l]} \tag{A.1a}$$

$$a^{[l]} = g^{[l]}(z^{[l]}) \tag{A.1b}$$

where the " * " operator indicates element-wise multiplication, indices (i, j) correspond to the kernel matrix W, x is the layer input (this is I for the first layer), and $b^{[l]}$ is a trainable bias. For the drifter application we use the so-called parametric rectified linear unit (PReLU) function, first introduced by He et al. (2016), as the nonlinear transfer function throughout the model:

$$g(x) = \begin{cases} c^{[l]}x & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases} \tag{A.2}$$

where $c^{[l]}$ is a trainable parameter.

The output of Eq. (A.1) is an embedding or feature image whose values are composed of the convolution at every position of the sliding kernel. For image processing applications, this feature image is sometimes interpretable; the kernel may, for example, learn edge detection for specific geometric patterns or orientations. Multiple features can be learned simultaneously by simply introducing additional kernels and applying Eq. (A.1) separately for each kernel. We train the STN to learn

the bivariate Gaussian probability density function (PDF) parameters $\mu_{dx}, \mu_{dy}, \sigma_{dx}, \sigma_{dy}, \rho$, corresponding to the means, standard deviations, and correlation between dx and dy (Eq. (4)).

Some comments on dimensionality are necessary here. First, CNN input need not be only two dimensional, as in the example above. A color image having $N_c = 3$ color channels can be represented by an array of size $(N_c \times h \times w)$. In this case, the convolution filter W will be a volume with dimensions $(N_c^{[l]} \times f^{[l]} \times f^{[l]})$ and Eq. (A.1) is summed over the third dimension as well. The key is that the resulting feature image summarizes in a single matrix patterns detected across all N_c channels of the original image. While a given kernel will always return a 2D array, if N_k kernels are used, the N_k outputs are stacked into a 3D feature array. Bias $b^{[l]}$, by convention, has dimensions $(N_k^{[l]} \times 1 \times 1 \times 1)$. Second, the feature array in the example above will have dimensions $[N_k^{[l]} \times (h^{[l-1]} - f^{[l]} + 1) \times (w^{[l-1]} - f^{[l]} + 1)]$ as a result of the convolution operation. This "shrinking output effect" can be eliminated by surrounding input x with p rows and columns of zeros to preserve the size of the image during convolution, where $p^{[l]} = (f^{[l]} - 1)/2$ is called *padding*. Finally, kernels need neither be square nor of size >1. Their shape and size is up to the engineer and ultimately depends on the task at hand, as will be seen below.

The CNNs within the STN employ two techniques to facilitate learning. Batch normalization is a method of normalizing the output of each network layer before passing it through the next layer (Ioffe and Szegedy, 2015). Similar to normalizing data before presenting it to a neural network, batch normalization helps deep networks optimize weights connecting interior layers by re-scaling intermediate feature arrays within the network. Second, residual blocks within the network help minimize exploding or vanishing gradient problems that are common in deep neural networks. Residual blocks allow layer activations (*e.g.*, x in Eq. (A.1a)) to pass deeper into the network by skipping intermediate layers (He et al., 2015). While theory dictates that training error should decrease monotonically as the number of layers of a network increases, this is not always true in practice. In the event that layers deep within a network end up being superfluous, the training algorithm can have a hard time finding parameters for these layers and, as a result, error will begin to increase as training continues. In this scenario, residual blocks learn the identity function for these extra layers, allowing the engineer to add layers to the network in hopes of improving performance without the risk of hampering the training process.

A.2. Graph CNN (GCNN)

Graph neural networks were first introduced by Gori et al. (2005) and have since evolved into their own family of neural networks (Merkwirth and Lengauer, 2005; Scarselli et al., 2009; Zhou et al., 2018; Bianchi et al., 2019; Wu et al., 2020) that include graph convolutional neural networks (*e.g.*, Gilmer et al., 2017; Hamilton et al., 2017; Kipf and Welling, 2017). Let a graph G be defined as a set of vertices V linked by edges E:

$$G \equiv (V, E) \tag{A.3}$$

We graphically represent a set of N_d drifters by compiling T temporal snapshots of feature observations $\mathbf{x} = [d(\text{lon}), d(\text{lat}), d^2(\text{lon}), d^2(\text{lat})]$ for each drifter into an array V with dimensions $(N_f \times T \times N_d)$, where N_f is the number of observed features.

A second edge array E encapsulates some relationship between linked vertices. Mohamed et al. (2020) use the similarity between nodes i and j based on the L^2 distance of their feature vectors $\mathbf{x}$ for this adjacency relationship:

$$e_t^{ij} = \begin{cases} \sum_{k=1}^{N_f} \dfrac{1}{\sqrt{(\mathbf{x}_t^{[k],i} - \mathbf{x}_t^{[k],j})^2}}, & (\mathbf{x}_t^{[k],i} - \mathbf{x}_t^{[k],j})^2 \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{A.4}$$

Eq. (A.4) produces an $(N_d \times N_d)$ adjacency matrix quantifying the connectivity between every combination of drifters in the time snapshot. Stacking T adjacency matrices as before generates an edge array E with dimensions $(T \times N_d \times N_d)$. Graph G_t from Eq. (A.3) can therefore be thought of as representing temporal snapshots of the oceanic flow captured by the drifters and the compilation of stacked graphs G_T as collectively summarizing the evolution of these flow dynamics over some time T.

CNNs are easily adaptable to multidimensional data represented in mathematical graph form. The GCNN takes as input both a vertex array V and an edge array E. The edge array is systematically normalized at each time step t by

$$E_t = D_t^{-\frac{1}{2}} \hat{E}_t D_t^{-\frac{1}{2}} \tag{A.5}$$

where $\hat{E}_t = E_t + I$, I is the identity matrix, and D_t is the diagonal node degree matrix of $\hat{E}_t$. Each E_t is then combined with its corresponding V_t by computing the dot product to generate a modified array $\hat{V}$ having the same dimensions as V but whose values have been adjusted by information coming from other nodes. Defining the vertex and edge arrays as node and adjacency arrays and combining them in this manner incorporates the "social" component into the STN. Its implementation in the pipeline is noted in the next section.

A.3. Social spatio-temporal GCNN

The first notable contribution of the STN model is the custom spatio-temporal unit (STU) that contains two components:

1. A *spatial convolution subunit* operates on the features dimension of the input graph. Convolving with N_k kernels of size $(N_f \times 1 \times 1)$ with no padding produces an embedding graph a (from Eq. (A.1)) with dimensions $(N_k \times T \times N_d)$, where $N_k = 5$ now represents the PDF parameters from Eq. (4) for each drifter. The embedding graph is then combined with the adjacency array E as described in Section A.2 resulting in a modified feature graph $\hat{V}$ having dimensions $(N_k \times T \times N_d)$ where the elements at each time step have been modified to incorporate information from nearby drifters.
2. A *temporal convolution subunit* operates across the time dimension of the output graph from the spatial convolution subunit using N_k kernels of size $(N_k \times 3 \times 1)$ to convolve three temporal snapshots for a single drifter at a time. This layer applies the necessary padding to preserve the size of the time dimension and produces a new output graph with the same dimensions as before $(N_k \times T \times N_d)$ where each element has been adjusted again to incorporate information about the adjacent time steps. For simplicity, the output here can also be denoted $\hat{V}$ since we need only to differentiate between it and the original graph V.

The rationale behind these choices of convolution methods bears explanation. Recalling that the convolution operation extracts patterns across all channels and projects them onto a new single channel feature matrix, it follows that the choice of a $(N_f \times 1 \times 1)$ kernel in the spatial convolution subunit is intended to look across the observations dimension (*i.e.*, the features) for patterns that occur in all of the observed variables. This unit-size kernel prevents information from being pooled between drifters or time steps and is an example of a so-called "network-in-network" (Lin et al., 2013). Note that padding is not necessary to preserve the shape of the original array here. These N_f kernels ultimately learn the Gaussian PDF of the spatial observations based on samples from the distribution by producing $\mu_x, \mu_y, \sigma_x, \sigma_y, \rho$ (see Eq. (4).) The temporal convolution subunit, on the other hand, looks for patterns across the new parameters and convolves three consecutive time steps at a time for each drifter. These kernels essentially learn the temporal trends in adjacent time steps. Note that the feature array

$\hat{V}$ is not necessarily physically interpretable at this point beyond the description just presented.

The social component of the STN is implemented in between these two STU subunits by combining $\hat{V}$ with E. E is essentially a weight matrix where, following from Eq. (A.4), larger e^{ij} indicates greater connectivity allowing more information to be "transferred" between drifters i and j. The general equation for the STN can be given by:

$$f(V^{(l)}, E) = g(D^{-\frac{1}{2}}\hat{E}D^{-\frac{1}{2}}V^{(l)}\mathbf{W}^{(l)}) \tag{A.6}$$

where $\mathbf{W}^{(l)}$ contains the trainable parameters at layer l and g is the PReLU activation function described above (Eq. (A.2)). Thus, a message passing operation of the GNN performs neighbor averaging whereby information is shared between similarly behaving drifters, as it was in the original pedestrian problem for people walking together. This "social" sharing of information is the second notable contribution of this architecture as it relates to the drifter prediction problem.

The final task of the STN is to forecast the PDF parameters into the future. This is done with a second time-extrapolating CNN. Instead of convolving over the features dimension to look for patterns present in each channel, this CNN convolves over the time dimension using kernels of size $(3 \times T \times 3)$ and padding $p = 1$ to preserve the array dimensions. The number of kernels used here corresponds to the number of future time steps T_{forecast} one wishes to predict. The final output is of size $(N_k \times T_{\text{forecast}} \times N_d)$.

We close by pointing out that an important aspect of CNNs is that they only require the number of channels to be constant across every example. Thus, a CNN trained to learn single-channel grayscale images can process an image of any dimensions, as long as it has a single color channel. Because the STN operates on both features and time, these dimensions must be the same across all examples. The big advantage here is that the STN is not sensitive to the number of drifters, provided each drifter contains the same number of observed variables and covers the same period of time. Further, the STN is entirely insensitive to the order in which the drifters are included in V, provided E is constructed with the drifters in the same order as they appear in V. These two factors make it easy to train the STN on real data sets where the number of active drifters is seldom constant throughout an entire experiment.

Data availability

Data will be made available on request.

References

Aksamit, N.O., Sapsis, T., Haller, G., 2020. Machine-learning mesoscale and submesoscale surface dynamics from Lagrangian ocean drifter trajectories. J. Phys. Oceanogr. 50 (5), 1179–1196. http://dx.doi.org/10.1175/JPO-D-19-0238.1.

Aref, H., 1984. Stirring by chaotic advection. J. Fluid Mech. 143, 1–21. http://dx.doi.org/10.1017/S0022112084001233.

Berloff, P.S., McWilliams, J.C., 2003. Material transport in oceanic gyres. Part III: Randomized stochastic models. J. Phys. Oceanogr. 33 (7), 1416–1445. http://dx.doi.org/10.1175/1520-0485(2003)033<1416:MTIOGP>2.0.CO;2.

Beron-Vera, F.J., LaCasce, J.H., 2016. Statistics of simulated and observed pair separations in the Gulf of Mexico. J. Phys. Oceanogr. 46 (7), 2183–2199. http://dx.doi.org/10.1175/JPO-D-15-0127.1.

Berta, M., Griffa, A., Özgökmen, T.M., Poje, A.C., 2016. Submesoscale evolution of surface drifter triads in the Gulf of Mexico. Geophys. Res. Lett. 43 (22), 11,751–11,759. http://dx.doi.org/10.1002/2016GL070357.

Bianchi, F.M., Grattarola, D., Livi, L., Alippi, C., 2019. Graph neural networks with convolutional ARMA filters. arXiv arXiv:1901.01343.

Bolton, T., Zanna, L., 2019. Applications of deep learning to ocean data inference and subgrid parameterization. J. Adv. Model. Earth Syst. 11 (1), 376–399. http://dx.doi.org/10.1029/2018MS001472.

Chassignet, E.P., Hurlburt, H.E., Smedstad, O.M., Halliwell, G.R., Hogan, P.J., Wallcraft, A.J., Baraille, R., Bleck, R., 2007. The HYCOM (HYbrid Coordinate Ocean Model) data assimilative system. J. Mar. Syst. 65 (1–4), 60–83. http://dx.doi.org/10.1016/j.jmarsys.2005.09.016.

Chassignet, E.P., Smith, L.T., Halliwell, G.R., Bleck, R., 2003. North Atlantic Simulations with the Hybrid Coordinate Ocean Model (HYCOM): Impact of the Vertical Coordinate Choice, Reference Pressure, and Thermobaricity. J. Phys. Oceanogr. 33 (12), 2504–2526. http://dx.doi.org/10.1175/1520-0485(2003)033<2504:NASWTH>2.0.CO;2.

Coelho, E.F., Hogan, P., Jacobs, G., Thoppil, P., Huntley, H.S., Haus, B.K., Lipphardt, B.L., Kirwan, A.D., Ryan, E.H., Olascoaga, J., Beron-Vera, F., Poje, A.C., Griffa, A., Özgökmen, T.M., Mariano, A.J., Novelli, G., Haza, A.C., Bogucki, D., Chen, S.S., Curcic, M., Iskandarani, M., Judt, F., Laxague, N., Reniers, A.J., Valle-Levinson, A., Wei, M., 2015. Ocean current estimation using a multi-model ensemble Kalman filter during the Grand Lagrangian Deployment experiment (GLAD). Ocean. Model. 87, 86–106. http://dx.doi.org/10.1016/j.ocemod.2014.11.001.

Coulin, J., Haley, Jr., P., Jana, S., Kulkarni, C., Lermusiaux, P., Peacock, T., 2017. Environmental Ocean and Plume Modeling for Deep Sea Mining in the Bismarck Sea. Ocean. 2017 MTS/ IEEE Anchorage 11, 1–10.

Curcic, M., Chen, S.S., Özgökmen, T.M., 2016. Hurricane-induced ocean waves and stokes drift and their impacts on surface transport and dispersion in the Gulf of Mexico. Geophys. Res. Lett. 43 (6), 2773–2781. http://dx.doi.org/10.1002/2015GL067619.

Davis, R.E., 1985. Drifter observations of coastal surface currents during CODE: The method and descriptive view. J. Geophys. Res. 90 (C3), 4741–4755. http://dx.doi.org/10.1029/JC090iC03p04741.

Dueben, P.D., Bauer, P., 2018. Challenges and design choices for global weather and climate models based on machine learning. Geosci. Model. Dev. 11 (10), 3999–4009. http://dx.doi.org/10.5194/gmd-11-3999-2018.

Dutt, A., Subramani, D., Kulkarni, C., Lermusiaux, P., 2018. Clustering of Massive Ensemble of Vehicle Trajectories in Strong, Dynamic and Uncertain Ocean Flows. Ocean. '18 MTS/ IEEE Charlest. http://dx.doi.org/10.1109/oceans.2018.8604634.

Enriquez, C., Mariño-Tapia, I.J., Herrera-Silveira, J.A., 2010. Dispersion in the Yucatan coastal zone: Implications for red tide events. Cont. Shelf Res. 30 (2), 127–137. http://dx.doi.org/10.1016/j.csr.2009.10.005.

Faghmous, J.H., Banerjee, A., Shekhar, S., Steinbach, M., Kumar, V., Ganguly, A.R., Samatova, N., 2014. Theory-guided data science for climate change. Comput. 47 (11), 74–78. http://dx.doi.org/10.1109/MC.2014.335.

Feng, A., You, C., Wang, S., Tassiulas, L., 2022. KerGNNs: Interpretable Graph Neural Networks with Graph Kernels. Proc. the AAAI Conf. Artif. Intell. 36 (6), 6614–6622.

Feppon, F., Lermusiaux, P.F., 2018. Dynamically orthogonal numerical schemes for efficient stochastic advection and lagrangian transport. SIAM Rev. 60 (3), 595–625. http://dx.doi.org/10.1137/16M1109394.

Franz, K., Roscher, R., Milioto, A., Wenzel, S., Kusche, J., 2018. Ocean eddy identification and tracking using neural networks. Int. Geosci. Remote. Sens. Symp..

Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E., 2017. Neural message passing for Quantum chemistry. Proc. the 34th Int. Conf. Mach. Learn. 70, 1263–1272.

Gori, M., Monfardini, G., Scarselli, F., 2005. A new model for learning in graph domains. Proc. 2005 IEEE Int. Jt. Conf. Neural Networks 2, 729–734. http://dx.doi.org/10.1109/IJCNN.2005.1555942.

Griffa, A., 1996. Applications of stochastic particle models to oceanographic problems. In: Stoch. Model. Phys. Oceanogr.. Birkhäuser Boston, Boston, MA, pp. 113–140. http://dx.doi.org/10.1007/978-1-4612-2430-3_5.

Grossi, M.D., Kubat, M., Özgökmen, T.M., 2020. Predicting particle trajectories in oceanic flows using artificial neural networks. Ocean. Model. 156, 101707. http://dx.doi.org/10.1016/J.OCEMOD.2020.101707.

Gupta, A., Lermusiaux, P.F., 2021. Neural closure models for dynamical systems. Proc. R. Soc. A Math. Phys. Eng. Sci. 477 (2252), 20201004. http://dx.doi.org/10.1098/rspa.2020.1004, arXiv:2012.13869.

Gupta, A., Lermusiaux, P., 2023. Generalized Neural Closure Models with Interpretability. Sci. Rep. 13 (10364), http://dx.doi.org/10.1038/s41598-023-35319-w.

Haley, Jr., P., Mirabito, C., Doshi, M., Lermusiaux, P., 2023. Ensemble Forecasting for the Gulf of Mexico Loop Current Region. OCEANS '23 IEEE/ MTS Gulf Coast http://dx.doi.org/10.23919/OCEANS52994.2023.10337035.

Hamilton, W.L., Ying, R., Leskovec, J., 2017. Inductive Representation Learning on Large Graphs. Adv. Neural Inf. Process. Syst. URL http://arxiv.org/abs/1706.02216. arXiv:1706.02216.

Haza, A.C., Özgökmen, T.M., Griffa, A., C., P.A., Lelong, M.-P., 2014. How does drifter position uncertainty affect ocean dispersion estimates? J. Atmospheric Ocean. Technol. 31 (12), 2809–2828. http://dx.doi.org/10.1175/JTECH-D-14-00107.1, URL https://journals.ametsoc.org/view/journals/atot/31/12/jtech-d-14-00107_1.xml.

Haza, A.C., Özgökmen, T.M., Hogan, P., 2016. Impact of submesoscales on surface material distribution in a Gulf of Mexico mesoscale eddy. Ocean. Model. 107, 28–47. http://dx.doi.org/10.1016/j.ocemod.2016.10.002.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Deep residual learning for image recognition. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 770–778. http://dx.doi.org/10.1109/CVPR.2016.90, arXiv:1512.03385.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. 2016 IEEE Conf. Comput. Vis. Pattern Recognition (CVPR) 770–778. http://dx.doi.org/10.1109/CVPR.2016.90.

Huang, K., Jin, Y., Candès, E., Leskovec, J., 2023a. Uncertainty Quantification over Graph with Conformalized Graph Neural Networks. 37th Conf. Neural Inf. Process. Syst. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/54a1495b06c4ee2f07184afb9a37abda-Paper-Conference.pdf.

Huang, Q., Yamada, M., Tian, Y., Singh, D., Chang, Y., 2023b. GraphLIME: Local interpretable model explanations for graph neural networks. IEEE Trans. Knowl. Data Eng. 35 (7), 6968–6972. http://dx.doi.org/10.1109/TKDE.2022.3187455.

Huntley, H.S., Lipphardt, Jr., B.L., Kirwan, Jr., A.D., 2019. Anisotropy and inhomogeneity in drifter dispersion. J. Geophys. Res.: Ocean. 124 (12), 8667–8682. http://dx.doi.org/10.1029/2019JC015179, arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2019JC015179.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. 32nd Int. Conf. Mach. Learn. 1, 448–456, arXiv:1502.03167.

Isaji, T., Spaulding, M.L., Allen, A.A., 2005. Stochastic particle trajectory modeling techniques for spill and search and rescue models. In: Estuarine and Coastal Modeling. pp. 537–547. http://dx.doi.org/10.1061/40876(209)31.

Jacobs, G.A., Bartels, B.P., Bogucki, D.J., Beron-Vera, F.J., Chen, S.S., Coelho, E.F., Curcic, M., Griffa, A., Gough, M., Haus, B.K., Haza, A.C., Helber, R.W., Hogan, P.J., Huntley, H.S., Iskandarani, M., Judt, F., Kirwan, A., Laxague, N., Valle-Levinson, A., Lipphardt, B.L., J. Mariano, A., Ngodock, H.E., Novelli, G., Olascoaga, M.J., Özgökmen, T.M., Poje, A.C., Reniers, A.J., Rowley, C.D., Ryan, E.H., Smith, S.R., Spence, P.L., Thoppil, P.G., Wei, M., 2014. Data assimilation considerations for improved ocean predictability during the Gulf of Mexico Grand Lagrangian Deployment (GLAD). Ocean. Model. 83, 98–117. http://dx.doi.org/10.1016/j.ocemod.2014.09.003.

Kingma, D.P., Ba, J., 2017. Adam: A method for stochastic optimization. arXiv:1412.6980.

Kipf, T.N., Welling, M., 2017. Semi-Supervised Classification with Graph Convolutional Networks. Conf. Learn. Represent. URL arXiv:1609.02907.

Koshel', K.V., Prants, S.V., 2006. Chaotic advection in the ocean. Phys.-Usp. 49 (11), 1151–1178. http://dx.doi.org/10.1070/PU2006v049n11ABEH006066.

Kubat, M., 1989. Floating approximation in time-varying knowledge bases. Pattern Recognit. 10 (4), 223–227. http://dx.doi.org/10.1016/0167-8655(89)90092-5.

Kubat, M., Holte, R., Matwin, S., 1998. Machine learning for the detection of oil spills in satellite radar images. Mach. Learn. 195–215.

Lermusiaux, P., 2001. Evolving the subspace of the three-dimensional multiscale ocean variability: Massachusetts Bay. J. Mar. Syst. 29, 385–422. http://dx.doi.org/10.1016/S0924-7963(01)00025-2.

Lermusiaux, P., Doshi, M., Kulkarni, C., Gupta, A., Haley, Jr., P., Mirabito, C., Trotta, F., Levang, S., Flierl, G., Marshall, J., Peacock, T., Noble, C., 2019. Plastic Pollution in the Coastal Oceans: Characterization and Modeling. MTS/ IEEE SEATTLE 1–10. http://dx.doi.org/10.23919/OCEANS40490.2019.8962786.

Lermusiaux, P., Haley, Jr., P., Mirabito, C., Mule, E., DiMarco, S., Dancer, A., Ge, X., Knap, A., Liu, Y., Mahmud, S., Nwankwo, U., Glenn, S., Miles, T., Aragon, D., Coleman, K., Smith, M., Leber, M., Ramos, R., Storie, J., Stuart, G., Marble, J., Barros, P., Chassignet, E., Bower, A., Furey, H., Jaimes de la Cruz, B., Shay, L., Tenreiro, M., Pallas Sanz, E., Sheinbaum, J., Perez-Brunius, P., Wilson, D., van Smirren, J., Monreal-Jiménez, R., Salas-de León, D., Contreras Tereza, V., Feldman, M., Khadka, M., 2024. Real-time Ocean Probabilistic Forecasts, Reachability Analysis, and Adaptive Sampling in the Gulf of Mexico. OCEANS '24 IEEE/MTS Halifax 1–10. http://dx.doi.org/10.1109/OCEANS55160.2024.10754153.

Lermusiaux, P., Lekien, F., 2005. Dynamics and Lagrangian Coherent Structures in the Ocean and their Uncertainty. Extended Abstract in report of the "Dynamical System Methods in Fluid Dynamic" Oberwolfach Workshop. Jerrold E. Marsden and Jurgen Scheurle (Eds.). Math. Forschungsinstitut Oberwolfach 2.

Lin, M., Chen, Q., Yan, S., 2013. Network In Network. 2nd Int. Conf. Learn. Represent. arXiv:1312.4400.

Lodise, J., Özgökmen, T., Griffa, A., Berta, M., 2019. Vertical structure of ocean surface currents under high winds from massive arrays of drifters. Ocean. Sci. 15 (6), 1627–1651. http://dx.doi.org/10.5194/os-15-1627-2019.

Lu, P., Lermusiaux, P.F., 2021. Bayesian learning of stochastic dynamical models. Phys. D Nonlinear Phenom. 427, 133003. http://dx.doi.org/10.1016/j.physd.2021.133003.

Mariano, A.J., Ryan, E.H., Huntley, H.S., Laurindo, L., Coelho, E., Griffa, A., Özgökmen, T.M., Berta, M., Bogucki, D., Chen, S.S., Curcic, M., Drouin, K., Gough, M., Haus, B.K., Haza, A.C., Hogan, P., Iskandarani, M., Jacobs, G., Kirwan, A.D., Laxague, N., Lipphardt, B., Magaldi, M.G., Novelli, G., Reniers, A., Restrepo, J.M., Smith, C., Valle-Levinson, A., Wei, M., 2016. Statistical properties of the surface velocity field in the northern Gulf of Mexico sampled by GLAD drifters. J. Geophys. Res. Ocean. 121 (7), 5193–5216. http://dx.doi.org/10.1002/2015JC011569.

Merkwirth, C., Lengauer, T., 2005. Automatic Generation of Complementary Descriptors with Molecular Graph Networks. J. Chem. Inf. Model. 45, 1159–1168. http://dx.doi.org/10.1021/ci049613b.

Mohamed, A., Qian, K., Elhoseiny, M., Claudel, C., 2020. Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction. IEEE/ CVF Conf. Comput. Vis. Pattern Recognit. arXiv:2002.11927.

Moradi Kordmahalleh, M., Gorji Sefidmazgi, M., Homaifar, A., 2016. A sparse recurrent neural network for trajectory prediction of atlantic hurricanes. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016. GECCO '16, Association for Computing Machinery, New York, NY, USA, pp. 957—-964. http://dx.doi.org/10.1145/2908812.2908834.

Nam, Y.-W., Cho, H.-y., Kim, D.-Y., Moon, S.-H., Kim, Y.-h., 2020. An improvement on estimated drifter tracking through machine learning and evolutionary search. Appl. Sci. 10 (22), 1–18. http://dx.doi.org/10.3390/app10228123.

Normile, D., 2014. Lost at sea. Science 344 (6187), 963–965. http://dx.doi.org/10.1126/science.344.6187.963.

Olascoaga, M.J., 2010. Isolation on the West Florida shelf with implications for red tides and pollutant dispersal in the Gulf of Mexico. Nonlinear Process. Geophys. 17 (6), 685–696. http://dx.doi.org/10.5194/npg-17-685-2010.

Olascoaga, M.J., Haller, G., 2012. Forecasting sudden changes in environmental pollution patterns. Proc. Natl. Acad. Sci. 109 (13), 4738–4743. http://dx.doi.org/10.1073/pnas.1118574109.

Özgökmen, T., 2013. GLAD experiment CODE-style drifter trajectories (low-pass filtered, 15 minute interval records), northern Gulf of Mexico near DeSoto Canyon, July-October 2012. Technical Report, Gulf of Mexico Research Initiative Information and Data Cooperative (GRIIDC), Harte Research Institute, Texas A&M University-Corpus Christi, http://dx.doi.org/10.7266/N7VD6WC8.

Özgökmen, T., Chassignet, E., Dawson, C., Dukhovskoy, D., Jacobs, G., Ledwell, J., Garcia-Pineda, O., MadDonald, I., Morey, S., Olascoaga, M., Poje, A., Reed, M., Skancke, J., 2016. Over what area did the oil and gas spread during the 2010 Deepwater Horizon oil spill? Oceanography 29 (3), 96–107. http://dx.doi.org/10.5670/oceanog.2016.74.

Özgökmen, T.M., Iliescu, T., Fischer, P.F., 2009. Large eddy simulation of stratified mixing in a three-dimensional lock-exchange system. Ocean. Model. 26 (3–4), 134–155. http://dx.doi.org/10.1016/j.ocemod.2008.09.006.

Özgökmen, T.M., Piterbarg, L.I., Mariano, A.J., Ryan, E.H., 2001. Predictability of drifter trajectories in the Tropical Pacific Ocean. J. Phys. Oceanogr. 31 (9), 2691–2720. http://dx.doi.org/10.1175/1520-0485(2001)031<2691:podtit>2.0.co;2.

Poje, A.C., Özgökmen, T.M., Lipphardt, B.L., Haus, B.K., Ryan, E.H., Haza, A.C., Jacobs, G.A., Reniers, A.J.H.M., Olascoaga, M.J., Novelli, G., Griffa, A., Beron-Vera, F.J., Chen, S.S., Coelho, E., Hogan, P.J., Kirwan, A.D., Huntley, H.S., Mariano, A.J., 2014. Submesoscale dispersion in the vicinity of the Deepwater Horizon spill. Proc. Natl. Acad. Sci. 111 (35), 12693–12698. http://dx.doi.org/10.1073/pnas.1402452111, arXiv:arXiv:1407.3308v2.

Poulain, P.-M., Centurioni, L., Özgökmen, T., 2022. Comparing the Currents Measured by CARTHE, CODE and SVP Drifters as a Function of Wind and Wave Conditions in the Southwestern Mediterranean Sea. Sensors (Basel) 22 (1), 353. http://dx.doi.org/10.3390/s22010353.

Poulain, P.-M., Gerin, R., 2019. Assessment of the Water-Following Capabilities of CODE Drifters Based on Direct Relative Flow Measurements. J. Atm. Ocean. Tech. 36 (4), 621–633. http://dx.doi.org/10.1175/JTECH-D-18-0097.1.

Pumir, A., Shraiman, B.I., Chertkov, M., 2000. Geometry of Lagrangian dispersion in turbulence. Phys. Rev. Lett. 85 (25), 5324–5327. http://dx.doi.org/10.1103/PhysRevLett.85.5324.

Qamar, R., Zardari, B., 2023. Artificial neural networks: An overview. Mesopotamian J. Comput. Sci. 2023, 130–139. http://dx.doi.org/10.58496/MJCSC/2023/015.

Rajagopal, E., Babu, A., Ryu, T., Haley, Jr., P., Mirabito, C., Lermusiaux, P., 2023. Evaluation of Deep Neural Operator Models toward Ocean Forecasting. OCEANS '23 IEEE/MTS Gulf Coast http://dx.doi.org/10.23919/OCEANS52994.2023.10337380.

Robinson, A., Haley, P., Lermusiaux, P., Leslie, W., 2002. Predictive Skill, Predictive Capability and Predictability in Ocean Forecasting. Proc. "OCEANS 2002 MTS/IEEE" 2, 787–794. http://dx.doi.org/10.1109/OCEANS.2002.1192070.

Santos Gutiérrez, M., Lucarini, V., Chekroun, M.D., Ghil, M., 2021. Reduced-order models for coupled dynamical systems: Data-driven methods and the Koopman operator. Chaos 31 (5), 053116. http://dx.doi.org/10.1063/5.0039496, arXiv:2012.01068.

Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G., 2009. The graph neural network model. IEEE Trans. Neural Netw. 20 (1), 61–80. http://dx.doi.org/10.1109/TNN.2008.2005605.

Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M., 2017. Modeling relational data with graph convolutional networks. Lect. Notes Comput. Sci. (Incl. Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 10843 LNCS, 593–607. http://dx.doi.org/10.1007/978-3-319-93417-4_38, arXiv:1703.06103.

van Sebille, E., Griffies, S.M., Abernathey, R., Adams, T.P., Berloff, P., Biastoch, A., Blanke, B., Chassignet, E.P., Cheng, Y., Cotter, C.J., Deleersnijder, E., Döös, K., Drake, H.F., Drijfhout, S., Gary, S.F., Heemink, A.W., Kjellsson, J., Koszalka, I.M., Lange, M., Lique, C., MacGilchrist, G.A., Marsh, R., Mayorga Adame, C.G., McAdam, R., Nencioli, F., Paris, C.B., Piggott, M.D., Polton, J.A., Rühs, S., Shah, S.H., Thomas, M.D., Wang, J., Wolfram, P.J., Zanna, L., Zika, J.D., 2018. Lagrangian ocean analysis: Fundamentals and practices. Ocean. Model. 121, 49–75. http://dx.doi.org/10.1016/j.ocemod.2017.11.008.

Serra, M., Sathe, P., Rypina, I., Kirincich, A., Ross, S., Lermusiaux, P., Allen, A., Peacock, T., Haller, G., 2020. Search and Rescue at Sea Aided by Hidden Flow Structures. Nat. Commun. 11, 2525. http://dx.doi.org/10.1038/s41467-020-16281-x.

Wei, M., Jacobs, G., Rowley, C., Barron, C.N., Hogan, P., Spence, P., Smedstad, O.M., Martin, P., Muscarella, P., Coelho, E., 2016. The performance of the US Navy's RELO ensemble, NCOM, HYCOM during the period of GLAD at-sea experiment in the Gulf of Mexico. Deep. Res. Part II Top. Stud. Ocean. 129, 374–393. http://dx.doi.org/10.1016/j.dsr2.2013.09.002.

Wilks, D.S., 2006. Statistical Methods in the Atmospheric Sciences. In: Helé, J. (Ed.), Int. Geophys. Ser., second ed. Academic Press, Boston, MA, pp. 88–95.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C., 2020. Connecting the dots: Multivariate time series forecasting with graph neural networks. KDD 753–763.

Xuanyuan, H., Barbiero, P., Georgiev, D., Magister, L.C., Liò, P., 2023. Global Concept-Based Interpretability for Graph Neural Networks via Neuron Analysis. Proc. the AAAI Conf. Artif. Intell. 37 (9), 10675–10683.

Yang, H., Liu, Z., 1997. The three-dimensional chaotic transport and the great ocean barrier. J. Phys. Oceanogr. 27 (7), 1258–1273. http://dx.doi.org/10.1175/1520-0485(1997)027<1258:TTDCTA>2.0.CO;2.

Ying, R., Bourgeois, D., You, J., Zitnik, M., Leskovec, J., 2019. GNNExplainer: Generating Explanations for Graph Neural Networks. 33rd Conf. Neural Inf. Process. Syst. 1–12.

Zargarbashi, S.H., Antonelli, S., Bojchevski, A., 2023. Conformal Prediction Sets for Graph Neural Networks. In: Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., Scarlett, J. (Eds.), Proceedings of the 40th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 202, PMLR, pp. 12292–12318, URL https://proceedings.mlr.press/v202/h-zargarbashi23a.html.

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2018. Graph neural networks: A review of methods and applications. arXiv arXiv:1812.08434.