МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра прикладной океанографии и комплексного управления прибрежными зонами

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

(бакалаврская работа)

На тему «Прогнозирование волнового поля в Балтийском море с использованием машинного обучения на основе спутниковых данных и реанализа»

Исполнитель: Воронкова Мария Сергеевна

Научный руководитель: канд. физ.-мат. наук, доцент Ерёмина Татьяна Рэмовна

«К защите допускаю»

Заведующий кафедрой _____

(подпись)

кандидат географических наук

(ученая степень, ученое звание)

Хаймина Ольга Владимировна

(фамилия, имя, отчество)

«09» 06 2025 г.

г. Санкт-Петербург

2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3
Глава 1. Волновые процессы в Балтийском море 6
1.1 Физико-географическое описание Балтийского моря 6
1.2 Описание волнового режима Балтийского моря 9
1.3 Прогноз волнового поля в Балтийском море 10
Глава 2. Материалы и методы исследований 13
2.1 Данные о ветре и волнах ERA5 13
2.2 Альтиметрические данные о волнах 14
2.3 Объединение данных 16
2.4 Машинное обучение. Теоретическая часть 18
2.5 Методы машинного обучения применительно к прогнозу волнения 22
2.6 Модель машинного обучения ConvLSTM применительно к прогнозу волнения 24
2.7 Модель машинного обучения XGBoost применительно к прогнозу волнения 27
Глава 3. Прогноз ветрового волнения в Балтийском море на основе машинного обучения 30
3.1 Выбор оптимальных параметров для XGBoost 30
3.2 Результаты 45
3.3 Анализ результатов 58
ЗАКЛЮЧЕНИЕ 60
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 62
ПРИЛОЖЕНИЕ А 66
ПРИЛОЖЕНИЕ Б 80

ВВЕДЕНИЕ

Балтийское море представляет собой внутренний шельфовый бассейн Атлантического океана, обладающий исключительной экономической, экологической и транспортной значимостью для Северо-Западного региона России. Его относительно небольшая площадь, сложная береговая линия и высокая изменчивость ветровых полей определяют разнообразие волновых процессов, а отсутствие длиннопериодной зыби обусловлено изолированностью акватории. Волновой режим Балтийского моря отличается выраженной сезонностью и значительной пространственно-временной изменчивостью, что напрямую влияет на безопасность судоходства, эксплуатацию портовой инфраструктуры и устойчивое развитие прибрежных территорий [3,14].

Традиционно прогнозирование волнения в регионе осуществлялось с помощью атласов волнения и синоптических методов, что позволяло лишь в общих чертах оценивать состояние моря [12]. Современные задачи обеспечения безопасности мореплавания, развития портовой инфраструктуры и минимизации ущерба от экстремальных погодных явлений требуют более точных и оперативных инструментов прогноза. Численные гидродинамические модели, такие как SWAN и WaveWatch III, позволяют учитывать широкий спектр физических процессов, однако их точность ограничивается качеством входных данных полей ветра, сложностью учёта локальных особенностей и высокой вычислительной затратностью, особенно в прибрежных и мелководных районах [11].

В последние годы наблюдается интенсивное развитие методов машинного обучения и интеграции спутниковых данных, что открывает новые перспективы для повышения точности и детализации прогноза морской среды. Современные алгоритмы анализа больших данных позволяют выявлять скрытые закономерности в изменении волновых характеристик, а также

оперативно обрабатывать большие массивы информации, поступающие из различных источников наблюдений и реанализа. В условиях роста интенсивности судоходства, увеличения числа экстремальных погодных явлений и необходимости адаптации к изменяющемуся климату, точное и своевременное прогнозирование волнового поля приобретает особую значимость.

Объектом исследования в данной работе является Балтийское море. Предметом исследования выступает прогноз волнового поля и его пространственно-временной изменчивости с использованием машинного обучения, а также интеграция спутниковых данных и данных реанализа для повышения точности и детализации прогноза.

Целью работы является разработка и оценка эффективности модели машинного обучения для прогнозирования волнового поля в Балтийском море на основе комплексного анализа спутниковых данных и результатов реанализа. Предполагается, что такой подход позволит повысить точность и заблаговременность прогнозов по сравнению с существующими численными моделями, а также обеспечит более детальное описание пространственно-временной изменчивости волнового поля в сложных прибрежных акваториях.

Для достижения поставленной цели в работе решаются следующие задачи: анализируются современные методы прогноза волнения; производится сбор и подготовка данных; интегрируются спутниковые альтиметрические данные и данные реанализа ERA5 для формирования обучающих выборок для одной из моделей; реализуются и оптимизируются модели машинного обучения, такие как ConvLSTM и XGBoost; проводится сравнительный анализ точности прогнозов с использованием различных метрик.

В работе используются современные массивы данных: реанализ ERA5, предоставляющий почасовые поля ветра и значительной высоты волн, а также спутниковые альтиметрические измерения. Для моделирования применяются

нейросетевые архитектуры, способные учитывать как временные, так и пространственные закономерности, а также алгоритмы градиентного бустинга для точечных прогнозов.

Глава 1. Волновые процессы в Балтийском море

1.1 Физико-географическое описание Балтийского моря

Балтийское море — внутриконтинентальный шельфовый бассейн Атлантического океана, глубоко врезанный в северо-западную часть Евразии. Оно расположено между 65°56′ и 54°46′ северной широты и 9°57′ и 30°00′ восточной долготы, почти полностью окружено сушей и лишь на юго-западе соединяется с Северным морем через датские проливы (Зунд, Большой и Малый Бельты), а далее — через Каттегат и Скагеррак [1,3].

Береговая линия Балтийского моря сильно изрезана, особенно в северной части, что обусловлено наличием многочисленных заливов и островов. Всего в море несколько тысяч островов, большинство из которых мелкие [3].

Площадь моря с проливами составляет 425,4 тыс. км², объём воды — 20,1 тыс. км³ [1]. При многообразии берегов рельеф дна Балтийского моря очень неровен. Небольшие глубины моря свидетельствуют о том, что оно целиком лежит в пределах материковой отмели. Средняя глубина моря 48 м, максимальная 459 м (Ландсортская котловина). Преобладают глубины до 50 м, на долю которых приходится 60 % площади моря, на долю глубин более 200 м— около 0,3 % площади моря [2,14].

Расположенное в умеренном поясе вблизи Атлантического океана и глубоко вклиненное в сушу Балтийское море характеризуется в основном морским климатом умеренных широт, вместе с тем морю свойственны черты континентального климата. Своеобразная конфигурация моря обусловливает его значительную протяженность с севера на юг и с запада на восток, что создает различия климатических условий в разных районах моря. Эти различия проявляются в неодинаковых от места к месту величинах основных

метеорологических элементов, характерных для каждого сезона. На формирование погодных условий в Балтийском регионе значительное влияние оказывают такие крупномасштабные атмосферные образования, как Исландский минимум, а также Сибирский и Азорский антициклоны. Взаимодействие этих барических систем определяет основные сезонные черты климата региона. В осенний и особенно зимний периоды наиболее активно проявляется взаимодействие между Исландским минимумом и Сибирским максимумом, что приводит к усилению циклонической активности над Балтийским морем. В результате осенью часто наблюдается прохождение глубоких циклонов с запада на восток, а зимой — на северо-восток. Эти атмосферные вихри приносят с собой облачную, пасмурную погоду, сопровождаемую сильными ветрами преимущественно юго-западного и западного направлений [3].

Среднемесячная температура воды в феврале составляет от 0 до 2 °C, летом достигает 20 °C. В отдельных районах море покрывается льдом. Балтийское море — солоноватый бассейн: солёность поверхностных вод изменяется от 8 ‰ у о. Борнхольм до 2–3 ‰ в Финском заливе, а придонные воды более солёные — 15–20 ‰. Благодаря речному стоку и эпизодическому проникновению более солёных вод из Северного моря формируются два слоя: верхний опреснённый и глубинный осолонённый, разделённые галоклином, который располагается на глубинах от 20–25 м (Арконская впадина) до 70–80 м (центральная часть моря). Солёность поверхностного слоя подвержена сезонным изменениям, связанным с колебаниями речного стока, а солёность нижнего слоя определяется неравномерным притоком североморских вод и может существенно меняться в многолетнем плане [4,5].

Южная и центральная части моря не замерзают. Сплошные морские льды сковывают лишь Ботнический, Финский и Рижский заливы [4]. Самым крупным северным незамерзающим портом на Балтике является Клайпеда. Водосборный бассейн характеризуется разветвленной речной сетью. В

Балтийское море впадает около 250 крупных и малых рек. По средним многолетним (1961—1970 гг.) данным они ежегодно вливают в море примерно 433 км^3 воды, что соответствует 2,1% от общего объема моря. Наибольшее количество воды приносят за год Нева (83,5 км^3). Висла (30,4 км^3), Неман (20,8 км^3), Даугава (19,7 км^3) и некоторые другие реки [2,3,5].

В северо-восточной части моря встречаются плавучие льды, их распространение зависит от суровости зимы. Ограниченный водообмен с Северным морем и значительный речной сток обусловливают низкую солёность. Поверхностная циркуляция вод имеет в целом циклонический характер (против часовой стрелки), осложнённый вихревыми образованиями. Обновление и вентиляция придонных вод зависят от поступления североморской воды; при его сокращении на глубинах и во впадинах формируются зоны с дефицитом кислорода [3,5].

Наиболее сильное ветровое волнение наблюдается осенью и зимой в глубоководных районах моря. Штормовые ветры вызывают волны высотой до 5–6 м и длиной 50–70 м. Из-за изолированности от Мирового океана приливы в Балтийском море незначительны — в отдельных пунктах их амплитуда не превышает 10–20 см. Более существенны сгонно-нагонные явления: у берегов уровень может меняться на 50 см, а в вершинах заливов — до 2 м [3].

Природные условия Балтийского моря способствуют его многостороннему хозяйственному использованию. Здесь ведётся промысел различных видов животных и растений, развивается марикультура. На побережье распространены россыпи полезных ископаемых, ведётся добыча янтаря, нефти и железной руды. По дну моря между Россией и Германией проложены две нити газопровода «Северный поток» («Nord Stream») протяжённостью 1224 км, соединяющие энергосистему Евросоюза с российскими месторождениями газа [13].

Балтийское море — район интенсивного судоходства, играющий важную роль в экономических связях европейских стран и мировых морских перевозках. Многочисленные порты связаны напряжёнными морскими и речными путями, по которым осуществляются крупные перевозки грузов и пассажиров. Через Балтийское море проходит значительная часть внешней торговли прибалтийских стран. В грузообороте преобладают нефтепродукты, уголь, лесоматериалы, целлюлоза, бумага и железная руда [3].

1.2 Описание волнового режима Балтийского моря

Волновой режим Балтийского моря отличается умеренной интенсивностью по сравнению с открытыми океаническими акваториями. В центральной части моря обычно наблюдаются волны высотой до 3,5 метров, иногда — выше 4 метров. В мелководных заливах волны ниже (до 3 метров), но они более крутые. Однако в условиях сильных штормов возможно формирование волн высотой более 10 метров; зафиксированы случаи, когда высота достигала 11 метров (например, в районе банки Эландс-Седра-Грунт).

Годовые колебания уровня моря играют важную роль в волновом и гидрологическом режиме Балтики, оказывая влияние на берега и инфраструктуру. На основе спутниковых и многолетних мареографных измерений показано, что волнообразные годовые возмущения уровня распространяются с юго-запада на северо-восток со скоростями 0,06–0,36 м/с. Эти колебания идентифицируются как внутренние волны Кельвина, а на юго-западе — в ряде случаев как бароклинные волны Россби [6].

Амплитуда годовых колебаний уровня увеличивается от 4–6 см в Датских проливах до 12–13 см в вершинах Финского и Ботнического заливов. Фаза годовых колебаний также изменяется при движении от юго-запада к северо-востоку, что подтверждает волновую природу этих процессов.

Модуляция амплитуды годовых волн связана с влиянием колебаний с периодами около года и межгодовой изменчивостью стратификации моря [6].

В последние десятилетия отмечается рост штормовой активности, связанный со смещением траекторий циклонов, проходящих над Балтикой. Анализ многолетних данных выявил цикличность периодов усиления и ослабления ветрового волнения с длительностью 10–12 лет [10]. Эти процессы оказывают влияние на экстремальные характеристики волн и их распределение по акватории.

Из-за изолированности моря и слабого поступления океанских волн через Датские проливы, волновой режим Балтики в основном формируется под воздействием местных ветров. Для численного моделирования ветрового волнения применяются спектральные модели (например, SWAN), которые показывают высокую точность при использовании современных реанализов атмосферных данных [10,14]. В расчетах море часто рассматривается как замкнутая система, что отражает его слабую связь с соседними акваториями [14].

Таким образом, волновой режим Балтийского моря определяется сочетанием уникальных гидрологических условий, ограниченного водообмена, выраженной сезонности и цикличности, а также значительного влияния локальных ветровых и атмосферных процессов.

1.3 Прогноз волнового поля в Балтийском море

Прогнозирование волнового поля в Балтийском море основывается на использовании современных численных моделей, интеграции спутниковых данных, а также методов машинного обучения. Каждый из подходов обладает своими особенностями и преимуществами, что позволяет повысить точность

и оперативность прогнозов для различных временных и пространственных масштабов.

К численным методам прогноза относятся спектральные модели третьего поколения, такие как SWAN (Simulating WAves Nearshore) и WaveWatch III [11,12]. Эти модели учитывают физические процессы генерации, распространения и затухания волн под действием ветра, рельефа дна и береговой линии. В качестве входных данных используются поля ветра, полученные из реанализа атмосферных данных (например, NCEP/CFSR, ERA5). Преимущества спектральных моделей заключаются в возможности моделирования волнения как в открытых, так и в прибрежных и мелководных районах, а также в учёте влияния ледового покрова в зимний период. Для повышения точности расчётов используются адаптивные расчетные сетки, что особенно важно для сложной береговой линии Балтийского моря.

Современные методы прогноза активно интегрируют спутниковые данные, такие как альтиметрические измерения, радиолокационные снимки и термические данные. Альтиметрические данные позволяют корректировать расчёты значительной высоты волн, а радиолокационные снимки используются для анализа пространственной структуры волновых полей. Интеграция спутниковых данных с результатами численного моделирования позволяет существенно повысить точность и достоверность прогнозов, особенно в условиях штормовой активности.

В последние годы активно развиваются методы машинного обучения, которые применяются для коррекции ошибок физических моделей, а также для самостоятельного прогноза волновых характеристик на основе больших массивов данных. Наиболее распространённые подходы включают регрессионные алгоритмы (градиентный бустинг, случайный лес), нейронные сети (в том числе LSTM для обработки временных рядов) и ансамблевое прогнозирование. Гибридные системы, сочетающие физические модели и машинное обучение, позволяют повысить точность прогнозов. Особое

значение приобретает дальнейшее развитие гибридных систем, сочетающих физическое моделирование, спутниковые данные и машинное обучение.

Для оценки эффективности методов прогноза используются стандартные статистические показатели, такие как среднеквадратическая ошибка (RMSE), средняя абсолютная ошибка (MAE) и коэффициент корреляции между расчетными и наблюдаемыми значениями. Валидация моделей проводится на основе данных буёв, платформ и независимых спутниковых наблюдений.

Глава 2. Материалы и методы исследований

2.1 Данные о ветре и волнах ERA5

ERA5 — это пятое поколение реанализа глобального климата и погоды, разработанное Европейским центром среднесрочных прогнозов погоды (ECMWF). Данный проект предоставляет глобальные почасовые оценки атмосферных, волновых и поверхностных переменных с горизонтальным разрешением 31км и 137 уровнями по вертикали от поверхности до 0,01hPa (около 80км).Там есть обширный набор данных глобального климатического реанализа с 1940 года по настоящее время включая в себя почасовые данные с пространственным разрешением при повторном анализе: 0,25° x 0,25° (атмосфера), 0,5° x 0,5° (волны в океане) по набору атмосферных и волновых параметров [15,18]. Данные основаны на современных моделях численного прогнозирования погоды и включают в себя интеграцию различных источников наблюдений, что позволяет получать более точные результаты.

В настоящей работе с сайта ECMWF были взяты данные о ветре и волнах. Данные о ветре включали в себя 10 м u- и v- составляющие скорости ветра, а данные о волнах включали в себя значительную высоту ветровых волн. Данные о ветре включали в себя горизонтальные компоненты вектора ветра — U и V на высоте 10 метров. Эти параметры можно объединить, чтобы получить скорость и направление горизонтального 10-метрового ветра. Данные о волнах включали значительную высоту ветровых волн. Этот параметр представляет собой среднюю высоту самой высокой трети поверхностных волн в океане/море, вызванных местным ветром. Он представляет собой расстояние по вертикали между гребнем и впадиной волны [15].

2.2 Альтиметрические данные о волнах

Альтиметрические данные были взяты из набора данных за 33 года (с 1985 по 2018), который содержит информацию о глобальной высоте волн и скорости ветра. Данные получены с помощью 13 спутников: GEOSAT, ERS-1, TOPEX, ERS-2, GFO, JASON-1, ENVISAT, JASON-2, CRYOSAT-2, HY-2A, SARAL, JASON-3 и SENTINEL-3A. Для наглядной иллюстрации длительности и охвата спутниковых наблюдений в работе приведён рисунок 1, где показаны интервалы функционирования каждой спутниковой миссии, что позволяет оценить полноту временного покрытия исходных данных. Результаты были откалиброваны и проверены по данным буёв Национального океанографического центра данных (NODC). Данные были объединены в ячейки размером 1° на 1° по всему миру, чтобы обеспечить удобный доступ для загрузки только интересующих регионов. Качество всех данных контролируется, что позволяет использовать их для различных научных исследований и прогнозов [16].

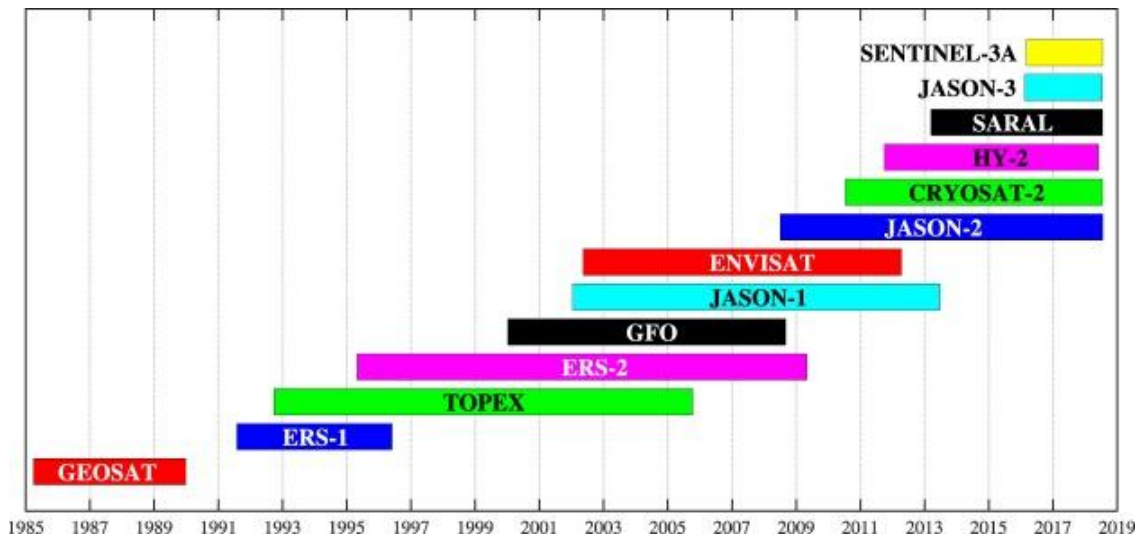


Рисунок 1 – Длительность альтиметрических данных в базе данных со всех спутниковых миссий [16]

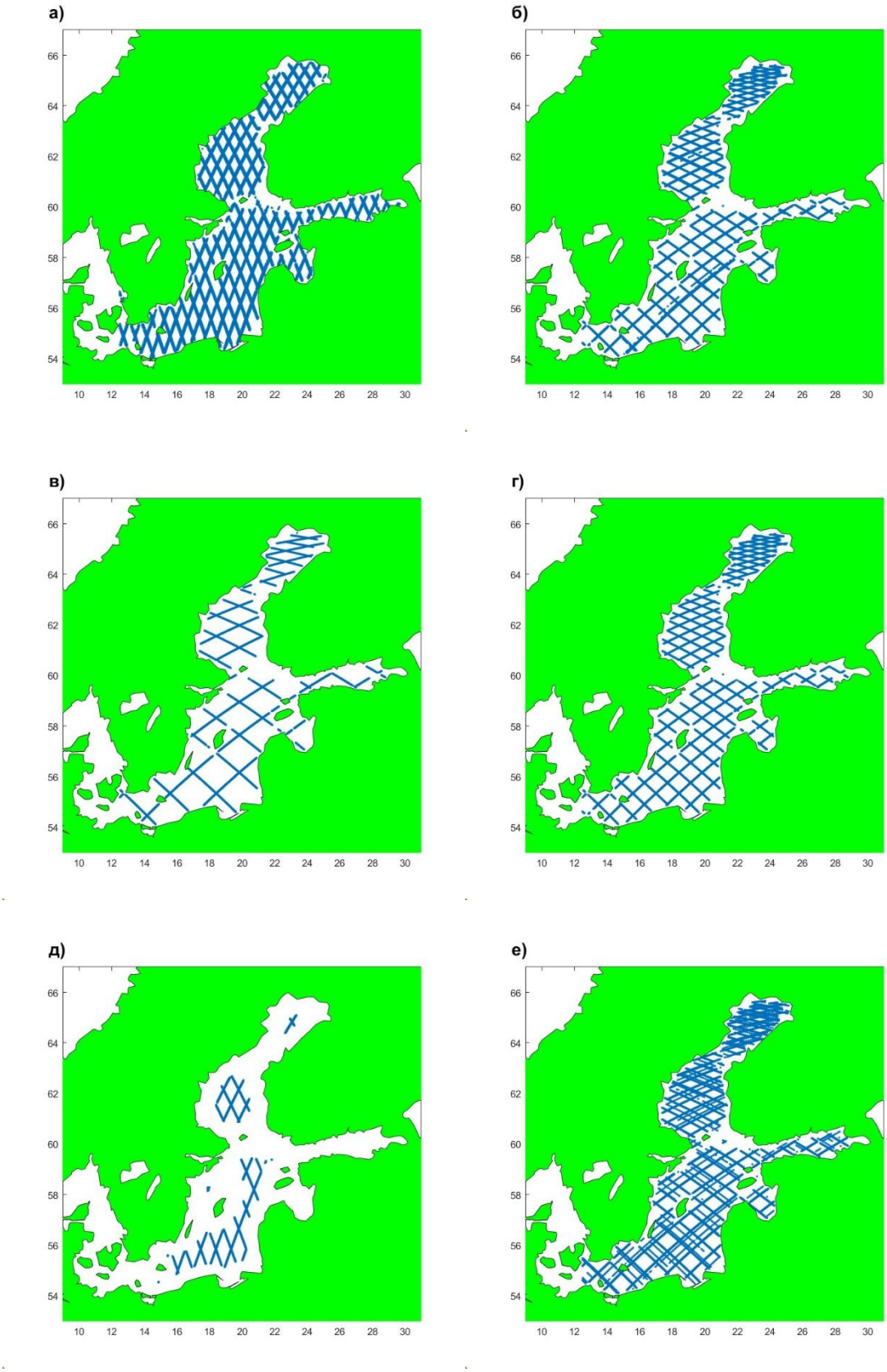


Рисунок 2 − Траектории спутников-альтиметров: а) ERS; б) Jason-1; в) Jason-2; г) Jason-3; д) Sentinel-3A; е) TOPEX

Траектории движения спутников-альтиметров, обеспечивающих пространственное распределение наблюдений по акватории Балтийского моря, представлены на рисунке 2. На данном рисунке детально отображены орбиты ключевых миссий (ERS, Jason-1, Jason-2, Jason-3, Sentinel-3A, TOPEX), что позволяет визуально оценить плотность и регулярность покрытия региона спутниковыми измерениями.

Как видно на рисунке 1, длительность спутниковых миссий обеспечивает непрерывное покрытие исследуемого периода. А траектории спутников, представленные на рисунке 2, демонстрируют высокую плотность наблюдений в акватории Балтийского моря, что критически важно для повышения пространственной детализации прогноз.

2.3 Объединение данных

Эффективное прогнозирование морских волн требует использования максимально репрезентативных и высокоточных данных, охватывающих как пространственную, так и временную изменчивость волнового режима. Для достижения этой цели в настоящей работе реализована процедура интеграции спутниковых альтиметрических измерений и данных реанализа с целью формирования единой обучающей выборки для моделей машинного обучения.

Реанализы, такие как ERA5, обеспечивают непрерывное пространственно-временное покрытие и согласованность данных, однако, как показано в ряде исследований, они могут систематически занижать экстремальные значения высоты волн и недостаточно точно воспроизводить локальные особенности, особенно в прибрежных и мелководных [18]. В то же время спутниковые альтиметрические данные обладают высокой точностью и независимостью от модельных предположений, но характеризуются ограниченной частотой и пространственным охватом, зависящими от орбит

спутников и погодных условий наблюдений. Объединение этих источников позволяет компенсировать недостатки каждого из них и получить более достоверную и детализированную картину волнового поля [26].

В рамках данного исследования интеграция данных осуществлялась по следующему алгоритму. На первом этапе производится загрузка временных рядов значительной высоты волн из реанализа ERA5 на регулярной сетке с почасовым разрешением. Затем извлекаются альтиметрические измерения высоты волн с различных спутников (ERS, Jason, Sentinel и др.) для акватории Балтийского моря. Для обеспечения сопоставимости данных проводится калибровка и контроль качества спутниковых наблюдений по данным буёв и платформ. На этапе пространственно-временного сопоставления для каждой точки спутникового трека определяется ближайшая по времени и координатам ячейка ERA5. Если разница по времени между спутниковым измерением и ближайшим временным шагом ERA5 не превышает заданного порога, в соответствующей ячейке ERA5 производится замена модельного значения на спутниковое. В случае отсутствия спутниковых данных в конкретной ячейке или временном интервале используется исходное значение ERA5, что обеспечивает полноту покрытия и непрерывность обучающей выборки. Итоговый массив содержит как оригинальные значения ERA5, так и скорректированные (замещённые) значения в местах и моментах наличия спутниковых наблюдений.

Такой подход позволяет повысить точность и пространственную детализацию данных о волновом поле, особенно в районах с недостаточной плотностью наземных наблюдений и в условиях экстремальных штормовых событий. Кроме того, формирование объединённой обучающей выборки, сочетающей преимущества спутниковых измерений (точность, независимость) и реанализа (полнота покрытия), позволяет повысить обобщающую способность моделей машинного обучения, снизить риск

переобучения на модельных ошибках и обеспечить более надёжный прогноз волнового режима в различных гидрометеорологических условиях.

2.4 Машинное обучение. Теоретическая часть

Машинное обучение представляет собой раздел искусственного интеллекта, направленный на разработку методов и алгоритмов, способных автоматически извлекать закономерности из данных и использовать их для решения прогностических и аналитических задач без явного программирования логики для каждого конкретного случая [7,16]. В основе машинного обучения лежат математические модели, позволяющие анализировать большие массивы информации, выявлять скрытые взаимосвязи и строить прогнозы на основе обнаруженных закономерностей. Данный подход особенно эффективен в задачах, где традиционные аналитические методы оказываются недостаточными для обработки сложных, многомерных и нелинейных зависимостей [9,17].

Процесс решения задач машинного обучения представляет собой структурированную последовательность взаимосвязанных этапов, каждый из которых играет критическую роль в обеспечении качества итогового результата. Первый этап включает формулировку бизнес-проблемы и формализацию задачи, на котором определяются цели исследования, доступные входные данные и критерии оценки успешности модели. Этап сбора данных предполагает агрегацию информации из различных источников и формирование единой базы данных для последующего анализа. Предобработка данных является одним из наиболее важных и трудозатратных этапов, включающим очистку от аномалий, заполнение пропущенных значений, нормализацию и стандартизацию признаков. На этапе генерации признаков осуществляется создание новых информативных характеристик на

основе исходных данных, что позволяет модели более эффективно выявлять сложные зависимости. Отбор информативных признаков направлен на исключение избыточной информации, которая может снижать качество обучения и приводить к переобучению модели. Настройка модели включает выбор подходящего алгоритма машинного обучения и оптимизацию его гиперпараметров для достижения максимальной точности прогнозов. Оценка качества модели производится с использованием независимых тестовых данных и специализированных метрик, таких как среднеквадратичная ошибка (RMSE), средняя абсолютная ошибка (MAE) и коэффициент корреляции. Завершающие этапы включают внедрение модели в производственную среду и организацию системы мониторинга ее работы для своевременного выявления деградации качества прогнозов.

Современные алгоритмы машинного обучения классифицируются по различным критериям, основным из которых является тип решаемой задачи [7]. Методы обучения с учителем применяются для задач классификации и регрессии, где модель обучается на размеченных данных с известными правильными ответами. К данной категории относятся линейная и логистическая регрессия, деревья решений, метод опорных векторов, случайный лес и градиентный бустинг [7,18].

Методы обучения без учителя используются для выявления скрытых структур в данных, включая кластеризацию, снижение размерности и анализ главных компонент. Обучение с подкреплением применяется в задачах, где агент должен принимать последовательность решений в динамической среде для максимизации долгосрочного вознаграждения. Глубокое обучение, основанное на многослойных нейронных сетях, показывает высокую эффективность в обработке сложноструктурированных данных, таких как изображения, временные ряды и пространственно-временные последовательности [7].

Применение методов машинного обучения в океанографии и гидрометеорологии обладает рядом существенных преимуществ по сравнению с традиционными подходами. Высокая скорость вычислений после завершения обучения модели позволяет получать прогнозы практически в реальном времени, что критически важно для оперативных задач морского мониторинга и предупреждения опасных явлений [8,9]. Способность алгоритмов машинного обучения выявлять сложные нелинейные зависимости между параметрами особенно ценна при анализе многофакторных океанографических процессов [8,9].

Гибкость в интеграции разнородных источников данных, включая спутниковые наблюдения, данные буев, реанализ и численные модели, позволяет создавать комплексные системы мониторинга морской среды. Автоматическая адаптация к изменяющимся условиям и способность работать с большими объемами данных делают методы машинного обучения особенно эффективными для задач климатического моделирования и долгосрочного прогнозирования.

Несмотря на значительные преимущества, методы машинного обучения имеют ряд ограничений, которые необходимо учитывать при их применении в гидрометеорологических задачах. Критическая зависимость от качества и объема обучающих данных может приводить к снижению точности модели в районах с редкой сетью наблюдений или при недостаточной представленности экстремальных событий в обучающей выборке. Риск переобучения возникает при избыточной сложности модели относительно объема доступных данных, что приводит к хорошему качеству на обучающей выборке, но низкой обобщающей способности.

Стохастическая природа алгоритмов машинного обучения может приводить к нарушению физических законов и ограничений, что особенно критично для задач, где важна физическая интерпретируемость результатов. Проблема "черного ящика" затрудняет понимание принципов принятия

решений моделью и может снижать доверие к полученным результатам среди экспертов предметной области. Необходимость регулярной переподготовки и валидации моделей для поддержания их актуальности требует значительных вычислительных ресурсов и экспертного времени.

Современные тенденции в области машинного обучения для океанографических приложений связаны с развитием гибридных методов, сочетающих преимущества физических моделей и алгоритмов искусственного интеллекта. Такие подходы позволяют использовать физически обоснованные ограничения для повышения надежности и интерпретируемости результатов машинного обучения. Интеграция методов машинного обучения с численными моделями океанской циркуляции и волновых процессов открывает новые возможности для повышения точности прогнозов и снижения вычислительных затрат. Для акватории Балтийского моря, характеризующейся сложной береговой линией, высокой пространственно-временной изменчивостью волнового режима и выраженной сезонностью, применение методов машинного обучения открывает особые возможности. Способность алгоритмов адаптироваться к локальным особенностям позволяет повысить точность прогнозов в прибрежных и мелководных районах, где традиционные численные модели показывают ограниченную эффективность. Интеграция спутниковых альтиметрических данных и результатов реанализа с использованием методов машинного обучения обеспечивает более детальное описание пространственной структуры волнового поля. Возможность оперативной обработки больших массивов гидрометеорологической информации делает машинное обучение перспективным инструментом для развития систем раннего предупреждения об опасных явлениях в Балтийском море. Автоматическая коррекция систематических ошибок численных моделей на основе исторических данных наблюдений позволяет повысить надежность долгосрочных климатических прогнозов для региона.

2.5 Методы машинного обучения применительно к прогнозу волнения

Одним из эффективных подходов является применение рекуррентных нейронных сетей с долгой краткосрочной памятью (LSTM) для краткосрочного прогноза значительной высоты волны (SWH). В исследовании, посвященном краткосрочному прогнозированию значительной высоты волны (SWH) с применением рекуррентных нейронных сетей с долгой краткосрочной памятью (Minuzzi и Farina, 2023) LSTM-модель обучалась на временных рядах ERA5 и данных с буёв для семи точек наблюдений. Прогноз выполнялся на промежутках 6, 12, 18 и 24 ч. Было показано, что средняя абсолютная ошибка (MAPE) при прогнозе SWH на 6 часов составляет 13%, а на 24 часа — около 26%, что сопоставимо с точностью реанализа ERA5. Для обучения использовалась кросс-валидация (80/20 split), а вычисления реализованы на базе TensorFlow и Keras. Модель показала высокую устойчивость и скорость работы: обучение занимало около 14 минут, а прогноз одного значения — доли миллисекунды. Таким образом, LSTM может выступать суррогатом физических моделей для оперативного прогноза волнения, особенно в задачах, где важна скорость расчёта и доступность данных [20].

В работе, где многослойный перцептрон (MLP) и метод опорных векторов (SVM) применялись для регрессии и классификации волновых параметров (James и др., 2018), модель обучалась на данных физической модели SWAN для залива Монтерей, где был сформирован обучающий набор на основе тысяч запусков SWAN, где входными параметрами выступали граничные условия (Hs, T, D), поля ветра, океанские течения и батиметрия. MLP использовался для регрессии значительной высоты волны, а SVM — для классификации характеристического периода. MLP достигал RMSE 9 см по SWH (менее 5% от среднего значения), а SVM правильно определял период волны в 90% случаев. Преимущество этого подхода — многократное

ускорение расчётов: ML-модели работают более чем в 1000 раз быстрее SWAN, что критично для оперативных задач и сценариев с ограниченными вычислительными ресурсами. Качество моделей оценивалось с помощью кросс-валидации и стандартных метрик ошибок. Такой подход позволяет эффективно заменять физические модели в ряде практических задач, не теряя в точности [17].

Для глобального моделирования волнения перспективным является использование глубоких сверточных нейронных сетей. В исследовании, где глубокая сверточная сеть U-Net применялась для глобального прогнозирования SWH с ассимиляцией спутниковых данных (Wang и др., 2024), модель обучалась на данных ERA5 и альтиметрических измерениях CCI-Sea State. модель обучалась на глобальных данных ERA5 (ветер и SWH, 0,5°×0,5°, 1 час) за 2000–2017 годы, а тестировалась на данных 2020 года. Входом для модели служили поле SWH на текущий момент и поле ветра на следующий час, что позволяет имитировать работу физических моделей. Для повышения устойчивости использовалось ансамблирование по эпохам. Модель показала, что после 240 часов автономного прогноза ошибки стабилизируются на уровне RMSE 0,23 м и коэффициента корреляции 0,986 по сравнению с ERA5, что сопоставимо с современными численными моделями. Ассимиляция альтиметрических данных CCI-Sea State дополнительно снижала ошибки (RMSE до 0,17 м, CC до 0,992) и ускоряла достижение стационарного уровня. Скорость расчёта: годовой глобальный прогноз — менее 10 минут на GPU. Модель устойчива к ошибкам и способна работать автономно без постоянной инициализации физическими моделями, что важно для оперативных и ресурсно-ограниченных задач [24].

Таким образом, современные методы машинного обучения демонстрируют высокую эффективность и точность в задачах прогноза волнения. Их применение позволяет существенно ускорить расчёты, повысить оперативность и снизить зависимость от вычислительно дорогих физических

моделей. Ассимиляция спутниковых данных дополнительно повышает устойчивость и точность моделей, что особенно важно для длительных автономных прогнозов и интеграции в оперативные гидрометеорологические сервисы.

2.6 Модель машинного обучения ConvLSTM применительно к прогнозу волнения

В данной работе для прогнозирования высоты волн используется модель машинного обучения ConvLSTM (Convolutional Long Short-Term Memory). Выбор архитектуры ConvLSTM для прогнозирования волнового поля в Балтийском море обусловлен рядом фундаментальных преимуществ данной модели перед альтернативными подходами машинного обучения. ConvLSTM представляет собой гибридную архитектуру, объединяющую преимущества сверточных нейронных сетей (CNN) для обработки пространственных данных и рекуррентных сетей с долгой краткосрочной памятью (LSTM) для анализа временных зависимостей. Эта комбинация делает ConvLSTM особенно подходящим для задач пространственно-временного прогнозирования, где критически важно одновременное моделирование как пространственных паттернов, так и временной динамики [19,23,24,25].

Ключевое преимущество ConvLSTM заключается в замене традиционных полносвязных операций внутри LSTM-ячеек на сверточные операции. Это позволяет модели сохранять пространственную структуру входных данных и эффективно выявлять локальные пространственные зависимости при обработке последовательных данных. В отличие от простого последовательного соединения CNN и LSTM слоев, ConvLSTM интегрирует сверточные операции непосредственно в внутренние преобразования LSTM, что обеспечивает более глубокое понимание пространственно-временных

закономерностей [19,23,24,25]. Особенно важным преимуществом ConvLSTM является способность обрабатывать граничные условия и внезапные изменения в пространственных паттернах. В реальных условиях прогнозирования волнения часто возникают ситуации, когда новые штормовые системы появляются на границах расчетной области, что представляет серьезную проблему для традиционных методов, основанных на оптическом потоке и полу-лагранжевой адвекции. ConvLSTM, обученная на разнообразных паттернах, способна распознавать подобные изменения и давать обоснованные прогнозы даже при появлении новых явлений [2]. Предложенная модель ConvLSTM имеет архитектуру, представленную на рисунке 3.

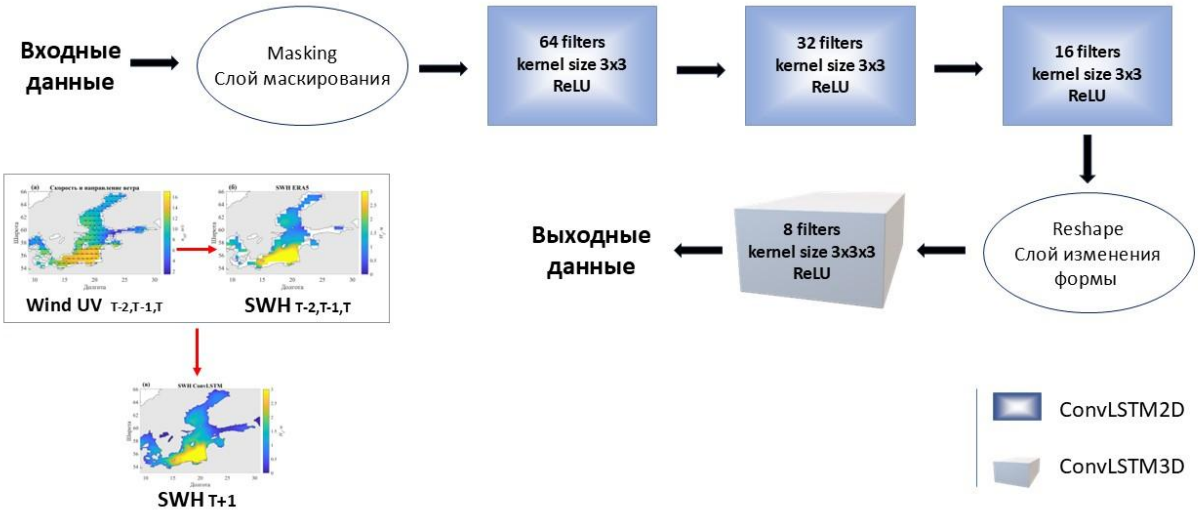


Рисунок 3 – Архитектура модели ConvLSTM

Модель, реализованная с помощью высокоуровневого набора функций и процедур нейронный сетей (TensorFlow Keras), предназначена для прогнозирования значительной высоты волны (SWH) с использованием данных о ветре. Она использует слои ConvLSTM2D для обработки

последовательных пространственных данных и слои Conv3D для окончательного прогноза. Входной слой определяет количество обучающих примеров и форму входных данных (длина последовательности временных шагов, количество ячеек широты и долготы в пространственной сетке и количество характеристик в каждой ячейке сетки). Далее идет слой маскирования, на котором предотвращается влияние значений, расположенных на суше, на процесс обучения. В модели используется три слоя ConvLSTM2D и заключительный слой ConvLSTM3D**.** Основные параметры ConvLSTM включают количество фильтров, размер окна свёртки и функцию активации. Например, в данной работе используется 64, 32 и 16 фильтров с размером ядра 3x3 или 3x3x3 и функцией активации ReLU, которая вносит нелинейность в модель. В ConvLSTM операция свёртки применяется как к входным, так и к рекуррентным преобразованиям, что позволяет эффективно обрабатывать пространственно-временные данные, такие как временные ряды с пространственными зависимостями. Это делает ConvLSTM особенно полезным для задач, требующих одновременного учёта временных и пространственных закономерностей. В таблице ниже представлены параметры модели машинного обучения ConvLSTM для данного исследования (табл. 1).

Таблица 1 – Параметры машинного обучения ConvLSTM

Параметр модели ConvSTM	Значение параметра
Фильтры (filters)	Число фильтров в слое
Размер окна (kernel size)	Размерность ядра фильтра
Возвращаемые последовательности (return sequences)	True - Возвращает полную последовательность состояний; False - Возвращает только результат последнего временного шага
Отсутп (padding)	Same - сохраняет исходные пространственные размеры; Valid - Выходные данные сжимаются
Функция активации (activation)	ReLU - Простая и эффективная нелинейная функция, позволяющая решить проблему изчезающего градиента

Модель, реализованная с помощью слоев ConvLSTM2D и Conv3D, демонстрирует высокую эффективность в прогнозировании значительной высоты волн на основе данных о ветре. Использование данных слоёв позволяет учитывать как временные, так и пространственные закономерности в данных, что делает модель особенно полезной для решения сложных задач прогнозирования.

2.7 Модель машинного обучения XGBoost применительно к прогнозу волнения

Помимо модели ConvLSTM, в данной работе для прогнозирования высоты волн используется модель машинного обучения, использующая градиентный бустинг. В основе используемой в данной работе модели XGBoost лежит алгоритм градиентного бустинга деревьев решений — техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей. Ключевое отличие градиентного бустинга от других ансамблевых методов заключается в последовательном характере обучения: на каждой итерации вычисляются отклонения предсказаний уже обученного ансамбля, и следующая модель добавляется для предсказания этих отклонений [19,22].

XGBoost значительно превосходит традиционные реализации градиентного бустинга благодаря ряду ключевых инноваций. Алгоритм использует передовые методы регуляризации L1 и L2 для предотвращения переобучения, что особенно важно при работе со сложными наборами данных [21].

Система эффективно обрабатывает разреженные данные и пропущенные значения, изучая оптимальное направление по умолчанию в структуре дерева. Это особенно ценно для океанографических наборов данных, где

неполнота информации является распространённой проблемой. XGBoost также реализует параллельное построение деревьев, что обеспечивает высокую производительность при работе с большими массивами данных [21].

Таким образом, XGBoost строит ансамбль слабых моделей (деревьев), последовательно минимизируя ошибку с помощью градиентного спуска и регуляризации для предотвращения переобучения. В контексте данной работы XGBoost применяется для прогнозирования высоты волн в Балтийском море с использованием объединённого массива спутниковых и реанализных данных. Модель строит ансамбль слабых моделей (деревьев), последовательно минимизируя ошибку с помощью градиентного спуска и регуляризации для предотвращения переобучения. Данные проходят предварительную очистку, кодирование категориальных признаков и разделение на обучающую и тестовую выборки. Критическим этапом является настройка гиперпараметров, включающих скорость обучения, глубину деревьев и количество итераций (табл. 2). Обучение проводится с помощью библиотеки xgboost, а оценка модели производится с использованием стандартных метрик производительности на тестовом наборе данных. Окончательно выбирается наилучшая обученная модель для прогнозирования новых данных.

Таблица 2 – Параметры машинного обучения ConvLSTM

Параметр модели XGBoost	Значение параметра
Выбор функции потерь (objective)	reg:squarederror - Это функция потерь, основанная на квадратичной ошибке (Mean Squared Error, MSE); reg:pseudohubererror - Это функция потерь, которая является гладкой аппроксимацией Huber loss
Метрики оценки производительности модели (eval metric)	rmse (Root Mean Squared Error) – корень из среднеквадратичной ошибки, mae (Mean Absolute Error) – средняя абсолютная ошибка, mape (Mean Absolute Percentage Error) – средняя абсолютная процентная ошибка
Скорость обучения (learning rate)	Определяет скорость обучения, которая контролирует вклад каждого дерева в итоговый прогноз. Низкое значение способствует стабильности обучения и предотвращает переобучение
Максимальная глубина дерева (max depth)	Ограничение глубины помогает предотвратить переобучение модели
Доля выборки для обучения каждого дерева (subsample)	Использование подвыборки помогает уменьшить переобучение
Доля признаков для построения каждого дерева (colsample bytree)	Доля признаков для построения каждого дерева. Использование части признаков для каждого дерева также способствует уменьшению переобучения
L1-регуляризация (alpha)	L1-регуляризация добавляет к функции потерь сумму абсолютных значений весов модели, умноженную на коэффициент alpha
Количество раундов бустинга (num boost round)	Общее количество раундов бустинга (или итераций), которое XGBoost выполнит при обучении модели. Каждый раунд бустинга включает в себя добавление нового дерева к ансамблю, которое корректирует ошибки предыдущих деревьев
Остановка обучения (early stopping rounds)	Количество раундов, в течение которых модель продолжает обучение, даже если метрика (например, RMSE) на валидационном наборе данных не улучшается

XGBoost представляет собой мощный и универсальный инструмент для прогнозирования морских параметров, сочетающий высокую точность предсказаний с вычислительной эффективностью. Его способность работать с табличными данными, обрабатывать пропущенные значения и предотвращать переобучение делает его особенно подходящим для океанографических приложений. Методы регуляризации и оптимизации XGBoost, наряду с его доказанной эффективностью в морском прогнозировании, подтверждают его статус как одного из наиболее популярных инструментов в области машинного обучения для структурированных данных.

Глава 3. Прогноз ветрового волнения в Балтийском море на основе машинного обучения

3.1 Выбор оптимальных параметров для XGBoost

В данном разделе представлен подробный анализ процесса подбора оптимальных гиперпараметров для модели XGBoost, используемой для прогноза значительной высоты волн в Балтийском море. Анализ основан на результатах серии тестов, отражённых в таблице 3 и соответствующих графических материалах (рисунки тестов №1–10) , где систематически изменялись функции потерь, скорость обучения, максимальная глубина деревьев и доля признаков, что позволило выявить наиболее эффективные комбинации параметров, что позволяет комплексно оценить влияние каждого параметра на итоговое качество модели.

Таблица 3 – Результаты настройки гиперпараметров модели XGBoost для прогнозирования волнового поля

№ теста	Функция потерь	Скорость обучения	Макс. глубина	Доля обучающих данных	Доля признаков	RSME (тест)	MAE (тест)	SI	Количество раундов обучения	Размер тестового набора данных	Время, мин
1	squarederror	0.01	15	1	1	0.27329	0.20563	0.2463	611	0.2	6.2
2	squarederror	0.05	15	1	0.85	0.25134	0.18927	0.2265	502	0.2	3.9
3	squarederror	0.01	25	1	1	0.24873	0.18482	0.2242	1000	0.2	17.7
4	squarederror	0.01	25	1	0.85	0.24641	0.18320	0.2221	945	0.2	17.0
5	**squarederror**	**0.1**	**25**	**1**	**0.85**	**0.24157**	**0.18044**	**0.2177**	**221**	**0.2**	**4.7**
6	pseudohubererror	0.01	15	1	1	0.26368	0.19814	0.2377	999	0.2	8.2
7	pseudohubererror	0.05	15	1	1	0.24802	0.18595	0.2235	582	0.2	4.4
8	pseudohubererror	0.01	25	1	0.85	0.24167	0.18080	0.2178	999	0.2	21.9
9	pseudohubererror	0.1	25	1	0.85	0.24112	0.17988	0.2173	269	0.2	4.7
10	**pseudohubererror**	**0.05**	**25**	**1**	**0.85**	**0.23921**	**0.17853**	**0.2156**	**460**	**0.2**	**7.5**

В ходе серии экспериментов по подбору параметров модели XGBoost было выявлено, что каждый из параметров оказывает заметное влияние на итоговое качество прогноза значительной высоты волн в Балтийском море. Влияние функции потерь на качество прогноза хорошо прослеживается при сравнении результатов тестов с идентичными параметрами, но разными функциями потерь. Например, переход от стандартной квадратичной ошибки к сглаженной Huber loss (сравнение теста №1 и теста №6) позволил снизить значение RMSE с 0.27329 м до 0.26368 м, что соответствует улучшению на 3,5%. Аналогичное снижение наблюдается по средней абсолютной ошибке (с 0.20563 м до 0.19814 м, то есть на 3,6%) и по индексу рассеяния (с 0.2463 до 0.2377, или на 3,5%). Таким образом, сглаженная Huber loss обеспечивает более устойчивую работу модели в присутствии выбросов и аномальных значений, что особенно важно для задач гидрометеорологического прогнозирования.

Скорость обучения также оказывает значимое влияние на итоговые метрики модели. Например, при использовании сглаженной Huber loss и глубине деревьев 15 увеличение скорости обучения с 0.01 до 0.05 (тест №6 и тест №7) приводит к снижению RMSE с 0.26368 м до 0.24802 м, что составляет 5,9% улучшения. Для моделей с максимальной глубиной 25 и долей признаков 0.85 снижение скорости обучения с 0.1 до 0.05 (тест №9 и тест №10) позволило дополнительно уменьшить RMSE с 0.24112 м до 0.23921 м (0,8%). Для моделей с функцией потерь с квадратичной ошибкой аналогичная тенденция наблюдается при сравнении теста №4 и теста №5: увеличение скорости обучения с 0.01 до 0.1 уменьшило RMSE с 0.24641 м до 0.24157 м (2%).

Максимальная глубина деревьев является одним из наиболее чувствительных параметров. Наиболее наглядно это видно при сравнении теста №1 и теста №3, где все параметры, кроме глубины, совпадают. Увеличение глубины с 15 до 25 позволило снизить RMSE с 0.27329 м до 0.24873 м, что означает улучшение на 9%. При этом средняя абсолютная

ошибка уменьшилась с 0.20563 м до 0.18482 м (10,1%), а индекс рассеяния — с 0.2463 до 0.2242 (9%). Это свидетельствует о том, что увеличение глубины деревьев позволяет модели лучше выявлять сложные и нелинейные зависимости между входными параметрами, что особенно важно для сложных природных процессов.

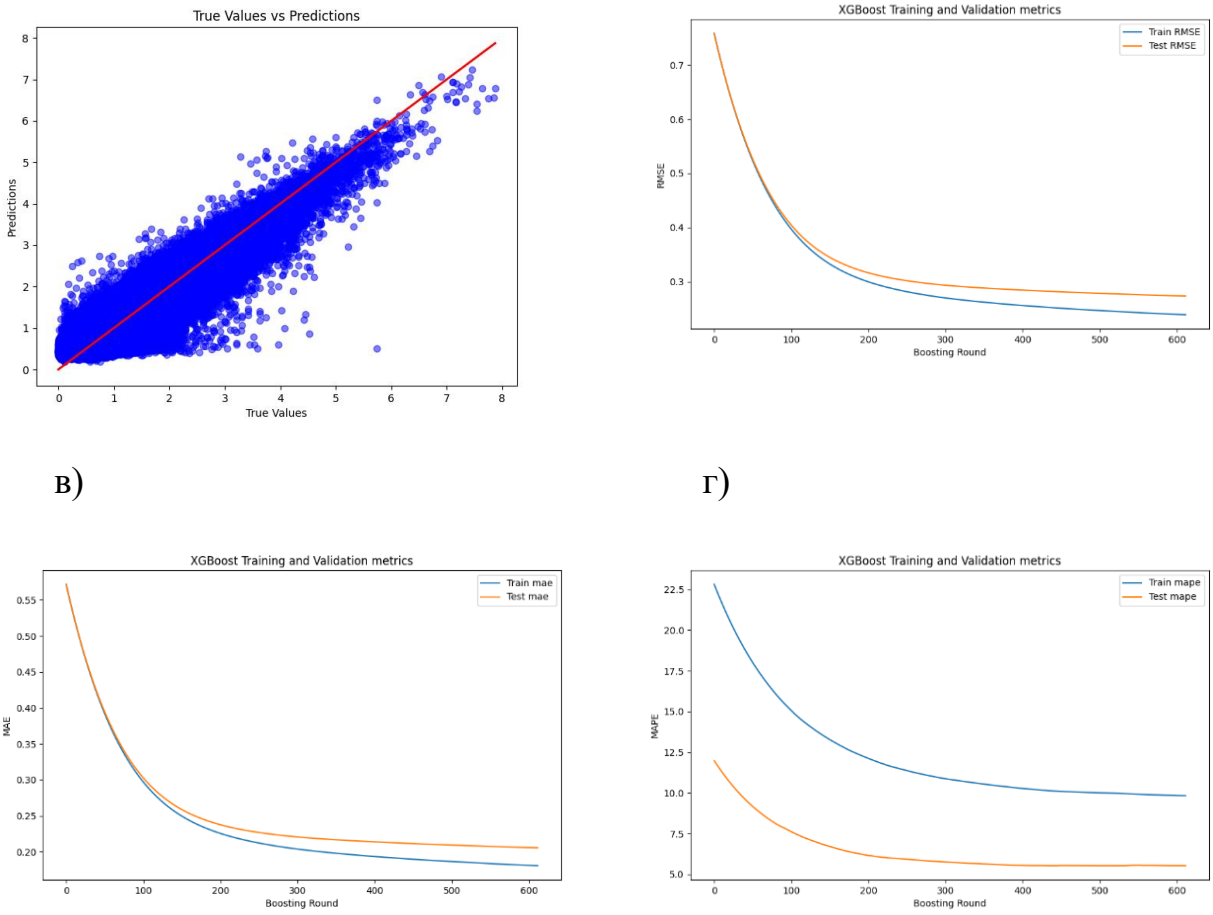
Доля признаков, используемых при построении каждого дерева, также оказывает влияние на итоговые метрики. Например, уменьшение доли признаков с 1 до 0.85 при прочих равных условиях (тест №3 и тест №4) позволило дополнительно снизить RMSE с 0.24873 м до 0.24641 м (0,9%) и MAE с 0.18482 м до 0.18320 м (0,9%). Такой подход способствует снижению риска переобучения и повышает обобщающую способность модели.

В результате комплексной настройки всех ключевых параметров (тест №10: сглаженная Huber loss, скорость обучения 0.05, глубина 25, доля признаков 0.85) удалось достичь минимального значения RMSE — 0.23921 м, что на 12,5% ниже по сравнению с первоначальной моделью (тест №1). Средняя абсолютная ошибка составила 0.17853 м, а индекс рассеяния — 0.2156.

Таким образом, наибольший вклад в повышение точности прогноза вносит именно комплексная оптимизация гиперпараметров, а не отдельные изменения. Проведённый анализ показывает, что для повышения точности прогноза значительной высоты волн в Балтийском море необходимо комплексно подходить к настройке модели XGBoost. Наибольший эффект достигается при использовании устойчивой к выбросам функции потерь, увеличении глубины деревьев, оптимизации скорости обучения и снижении доли используемых признаков. Такой подход позволяет существенно снизить ошибки прогноза и повысить надёжность модели для практического применения.

На рисунках 4-14 каждый рисунок иллюстрирует результаты отдельного теста по подбору гиперпараметров модели XGBoost для прогноза значительной высоты волн в Балтийском море. Каждый набор рисунков для теста включает несколько ключевых визуализаций: график "True Values vs Predictions", а также кривые обучения по метрикам RMSE, MAE и MAPE для обучающей и тестовой выборок

а) б)



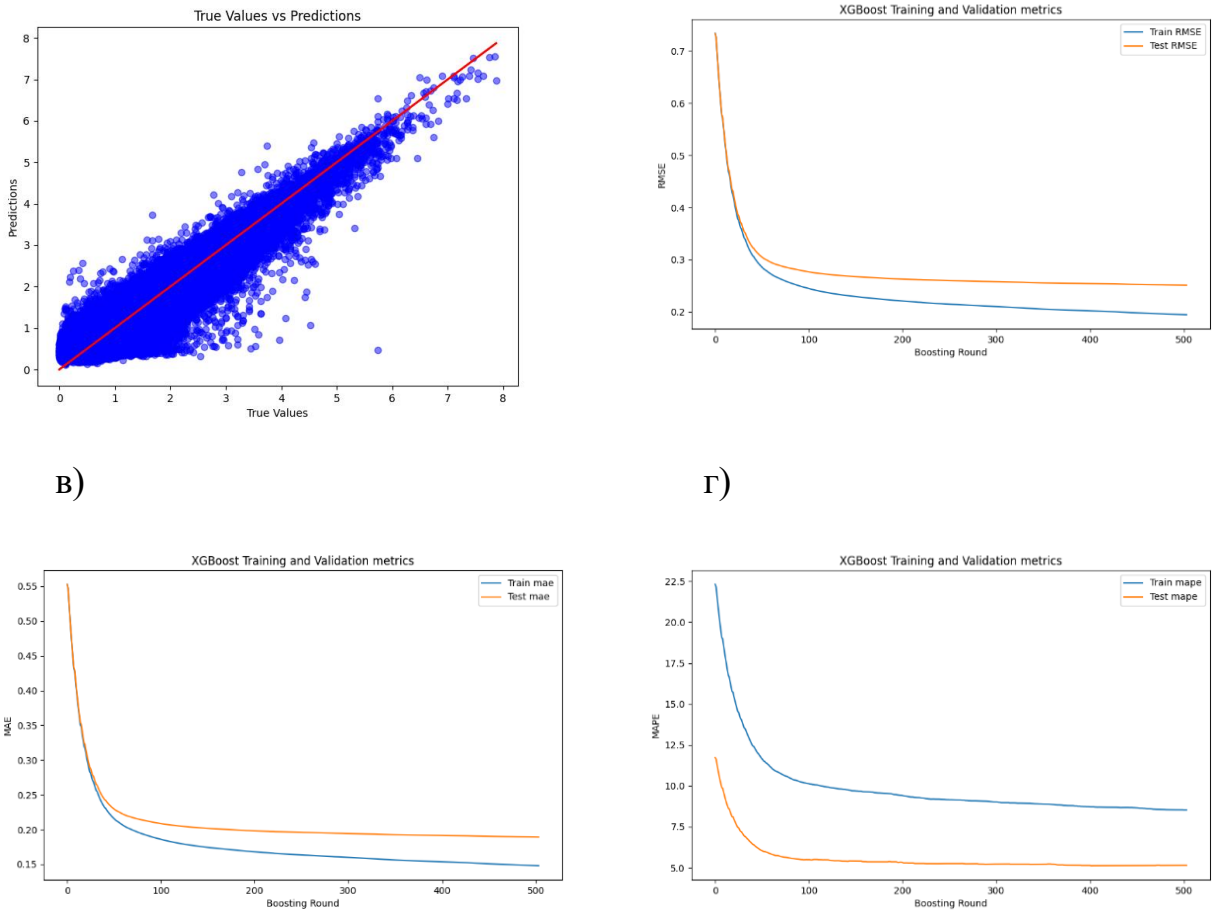
в) г)

Рисунок 4 – Результаты теста №1

Рисунок 4, отображающий результаты первого теста, показывает результат базовой конфигурации XGBoost с минимальной глубиной деревьев и стандартной функцией потерь. На графике "True Values vs Predictions" наблюдается значительное рассеяние точек относительно идеальной линии,

что указывает на недостаточную гибкость модели и наличие систематических ошибок. Кривые ошибок на обучающей и тестовой выборках расходятся, что свидетельствует о недообучении.

а) б)

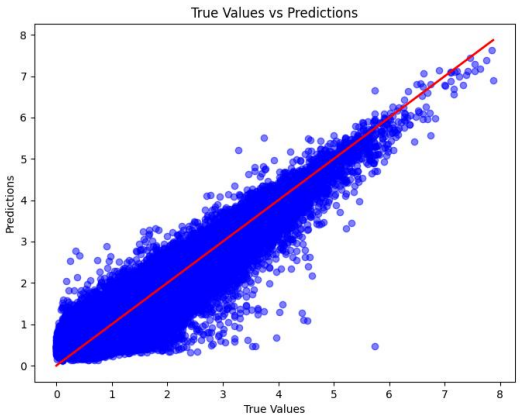


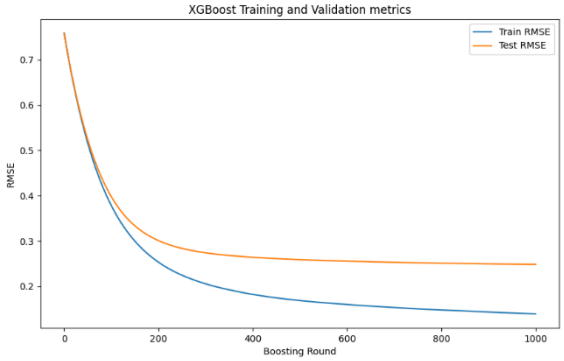
в) г)

Рисунок 5 – Результаты теста №2

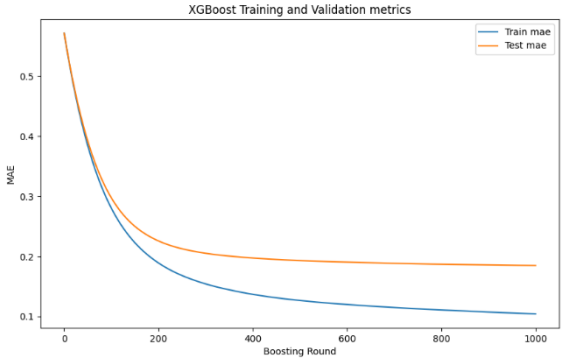
Рисунок 5 демонстрирует влияние изменения скорости обучения (увеличение) и доли признаков (уменьшение). Видно небольшое улучшение: точки начинают группироваться ближе к диагонали, но разброс все еще значителен. Кривые ошибок становятся чуть более пологими, но разница между обучающей и тестовой ошибками сохраняется.

а) б)





в) г)



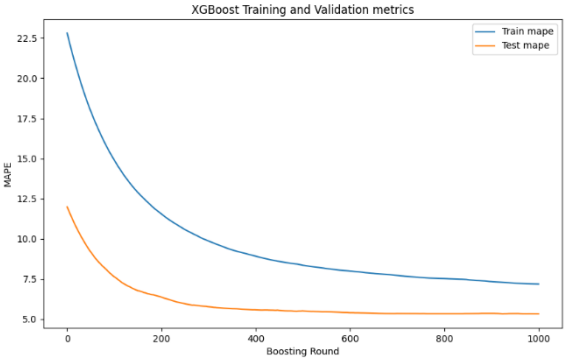
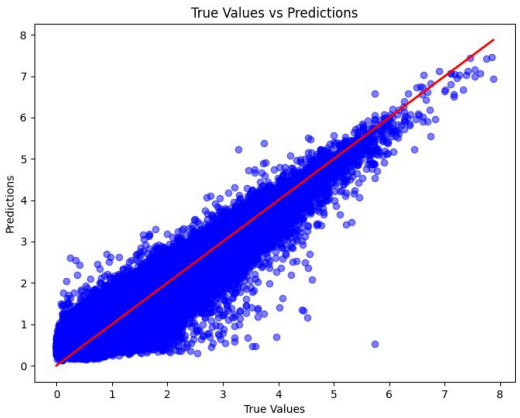
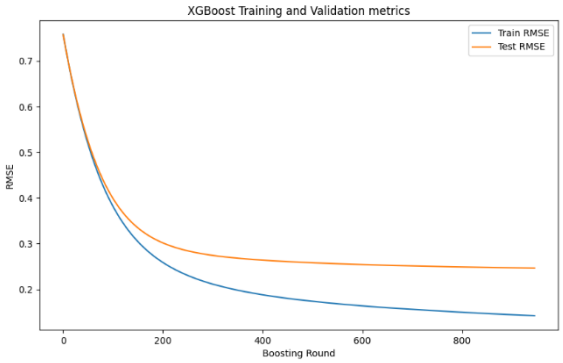


Рисунок 6 – Результаты теста №3

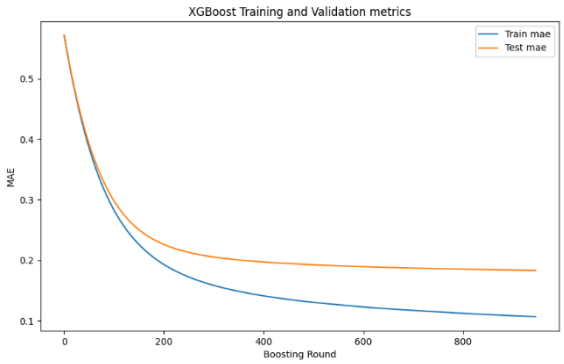
Здесь увеличена глубина деревьев относительно теста№1 (рис. 6). Это приводит к заметному снижению ошибок: точки на диаграмме "True Values vs Predictions" ближе к идеальной линии, а кривые ошибок на обучающей и тестовой выборках сближаются. Это свидетельствует о лучшем выявлении сложных зависимостей моделью.

а) б)





в) г)



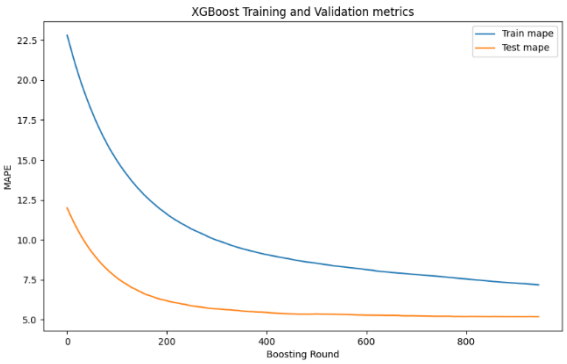
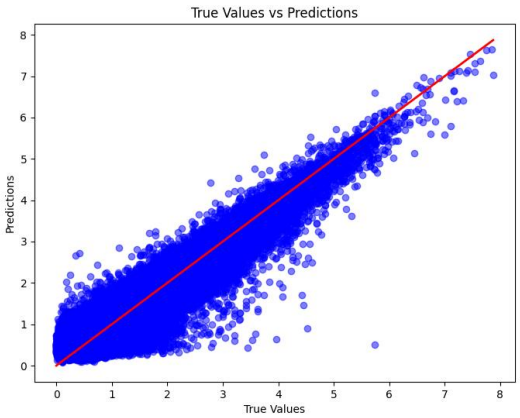
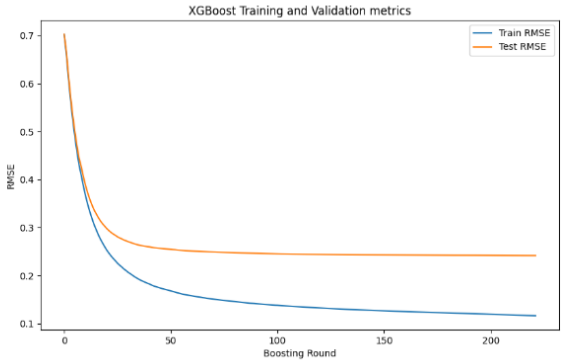


Рисунок 7 – Результаты теста №4

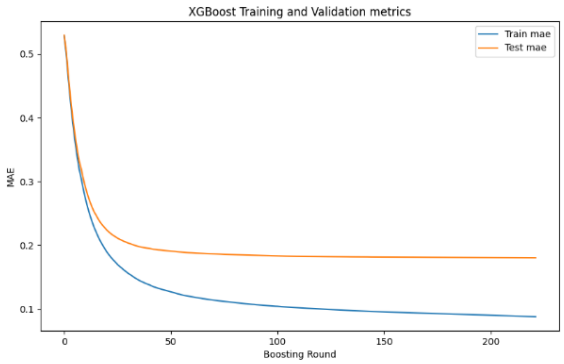
На 4 тесте уменьшена доля признаков при построении каждого дерева относительно теста №3. Это способствует снижению риска переобучения: кривая тестовой ошибки становится более стабильной, а разброс точек на диаграмме уменьшается (рис. 7).

а) б)





в) г)



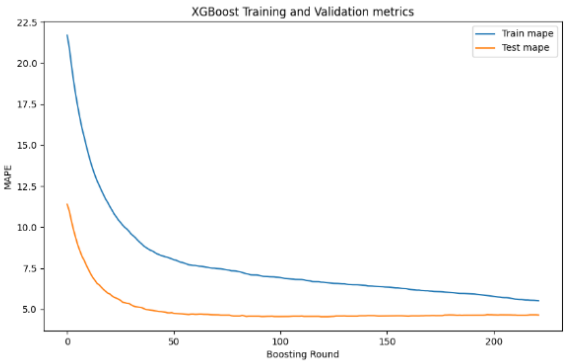
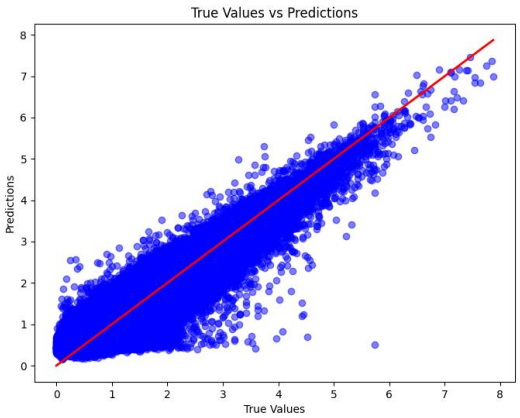
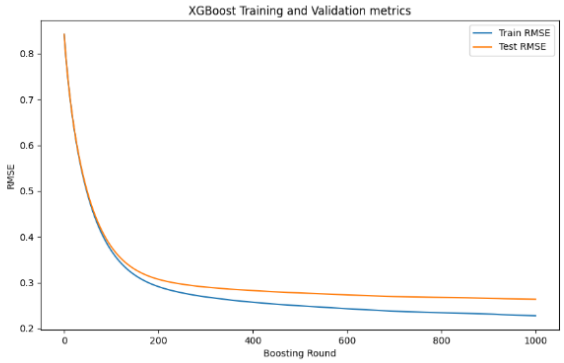


Рисунок 8 – Результаты теста №5

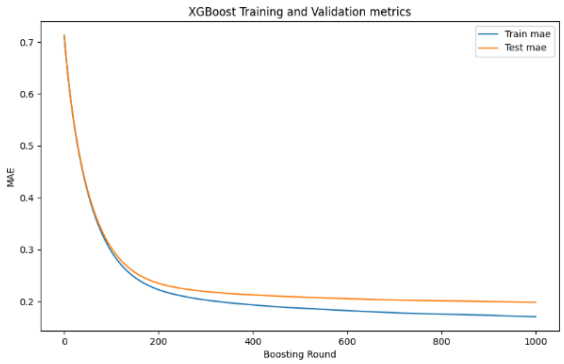
На рисунке 8 комплексно оптимизированы несколько параметров (скорость обучения, максимальная глубина, доля признаков). Кривая тестовой ошибки быстро выходит на плато, а разница между обучающей и тестовой ошибками минимальна. Это указывает на хороший баланс между обучением и обобщающей способностью модели

а) б)





в) г)



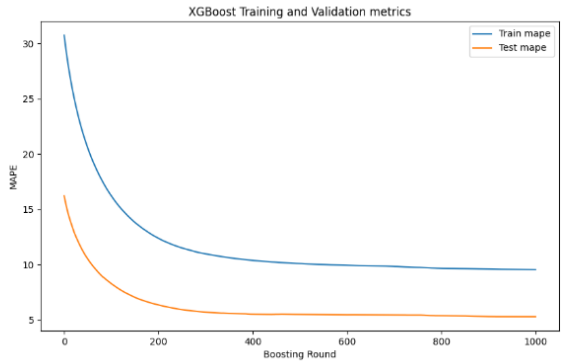
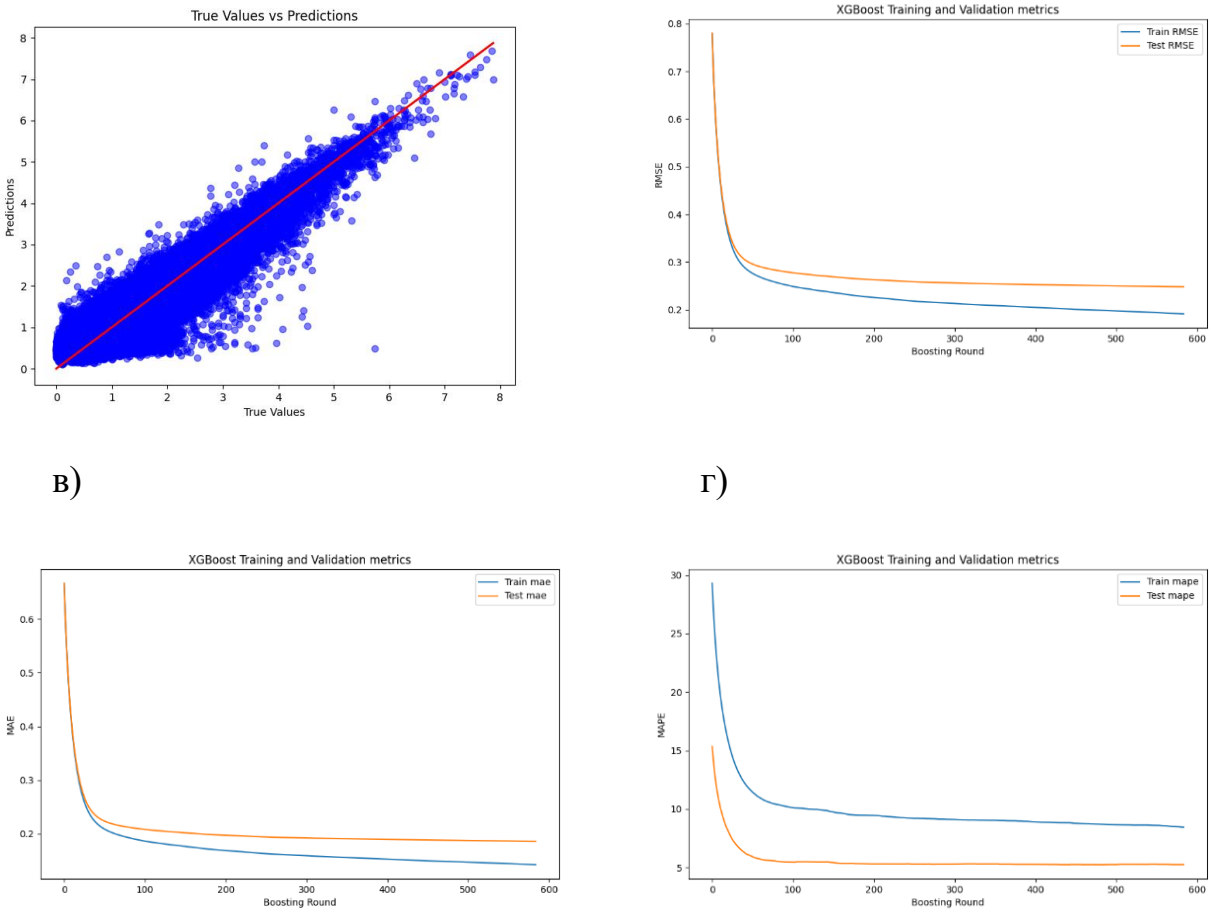


Рисунок 9 – Результаты теста №6

В серии тестов №6-10 был проведен переход к сглаженной функции потерь Huber loss. На риснунке 9 на графике "True Values vs Predictions" точки располагаются ближе к диагонали по сравнению с тестами, использующими стандартную квадратичную ошибку, что указывает на уменьшение влияния выбросов. Кривые ошибок (RMSE, MAE, MAPE) для обучающей и тестовой выборок становятся более стабильными, разница между ними сокращается. Переход к Huber loss позволил снизить RMSE и MAE по сравнению с базовой моделью (тест №1). Модель стала более устойчива к аномальным значениям, что важно для гидрометеорологических данных с выбросами. Однако при сохранении низкой скорости обучения и глубины деревьев полностью

избавиться от недообучения не удалось — кривая ошибок выходит на плато только после большого числа итераций

а) б)



в) г)

Рисунок 10 – Результаты теста №7

На рисунке 10 отображен результат увеличения скорости обучения относительно теста №6. Кривые ошибок на обучающей и тестовой выборках быстро стабилизируются, достигая плато уже после 200–300 итераций. На графике "True Values vs Predictions" точки ещё плотнее группируются вдоль диагонали, разброс уменьшается. Увеличение learning rate до 0.05 позволило дополнительно снизить RMSE по сравнению с тестом №6. Модель быстрее сходится, что сокращает время обучения без потери качества. Сохраняется

баланс между обучением и обобщающей способностью, признаки переобучения отсутствуют.

а) б)

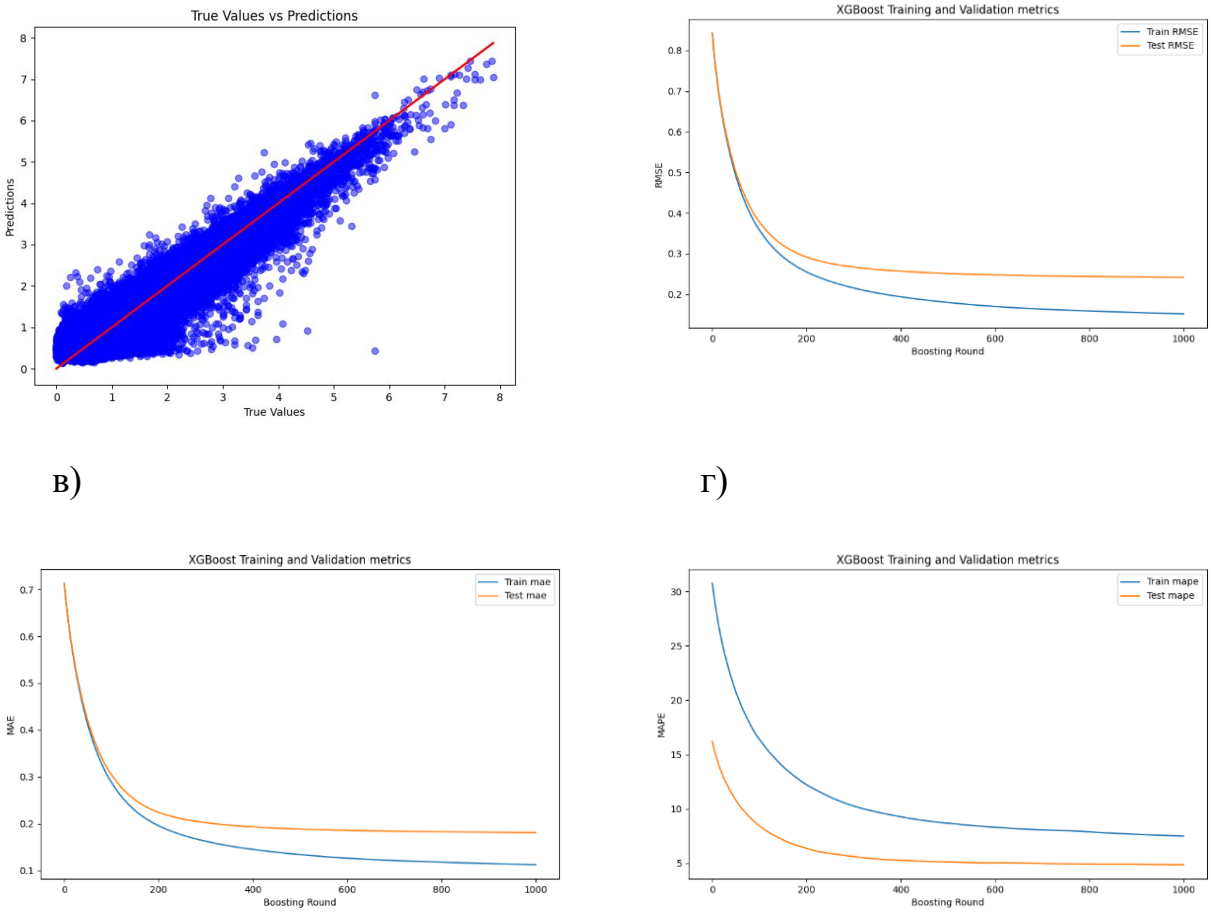
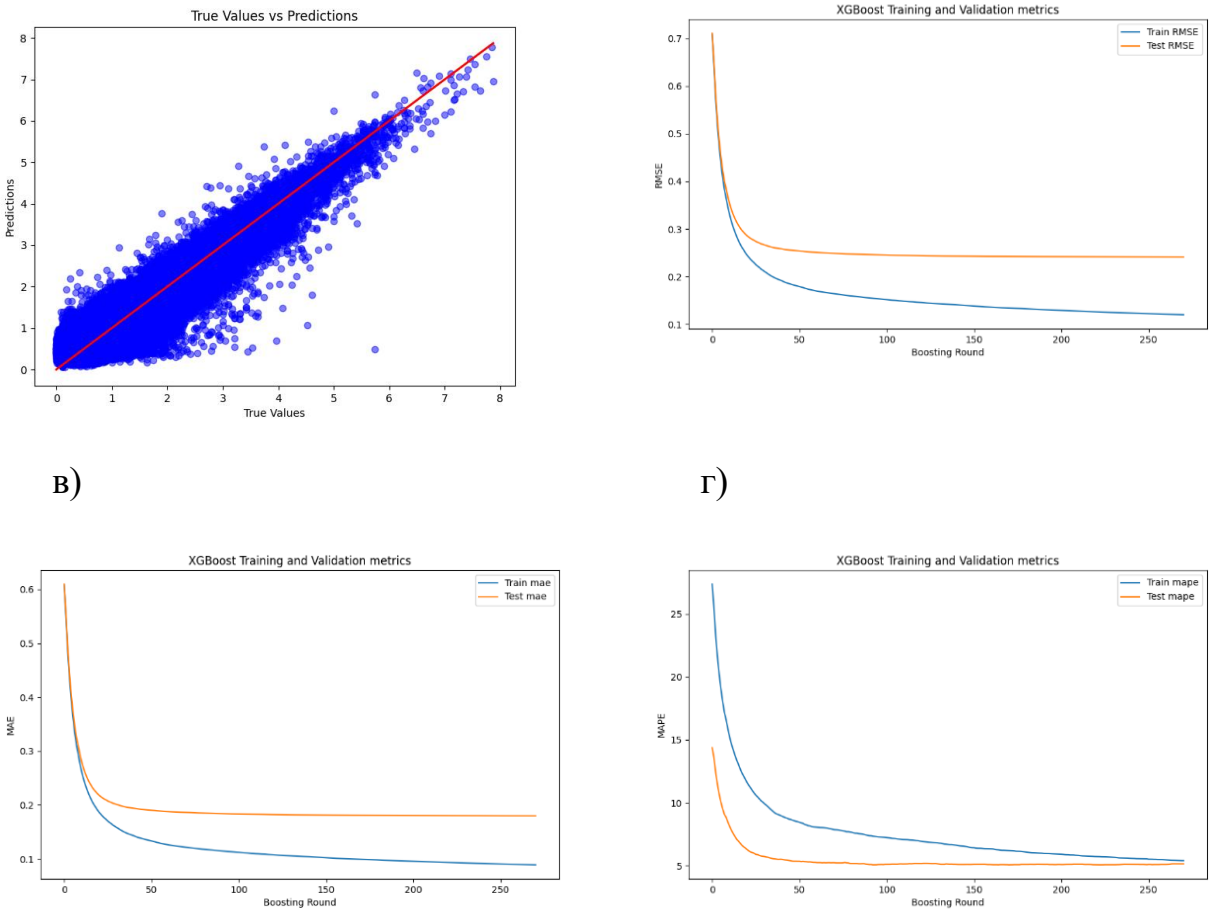


Рисунок 11 – Результаты теста №8

В тесте №8 (рисунок 11) отображаются результаты влияния увеличения глубины деревьев. Кривые ошибок становятся ещё более пологими, разница между обучающей и тестовой ошибками минимальна. На графике "True Values vs Predictions" точки максимально приближены к идеальной линии, особенно в области экстремальных значений. Увеличение глубины деревьев с 15 до 25 позволило ещё сильнее снизить RMSE и MAE, особенно за счёт лучшего выявления сложных и нелинейных зависимостей. Модель стала чувствительнее к сложным структурам данных, что важно для прогноза волн в

сложных акваториях. Однако при слишком большой глубине возможен риск переобучения, поэтому важно контролировать кривые ошибок на тестовой выборке.

а) б)



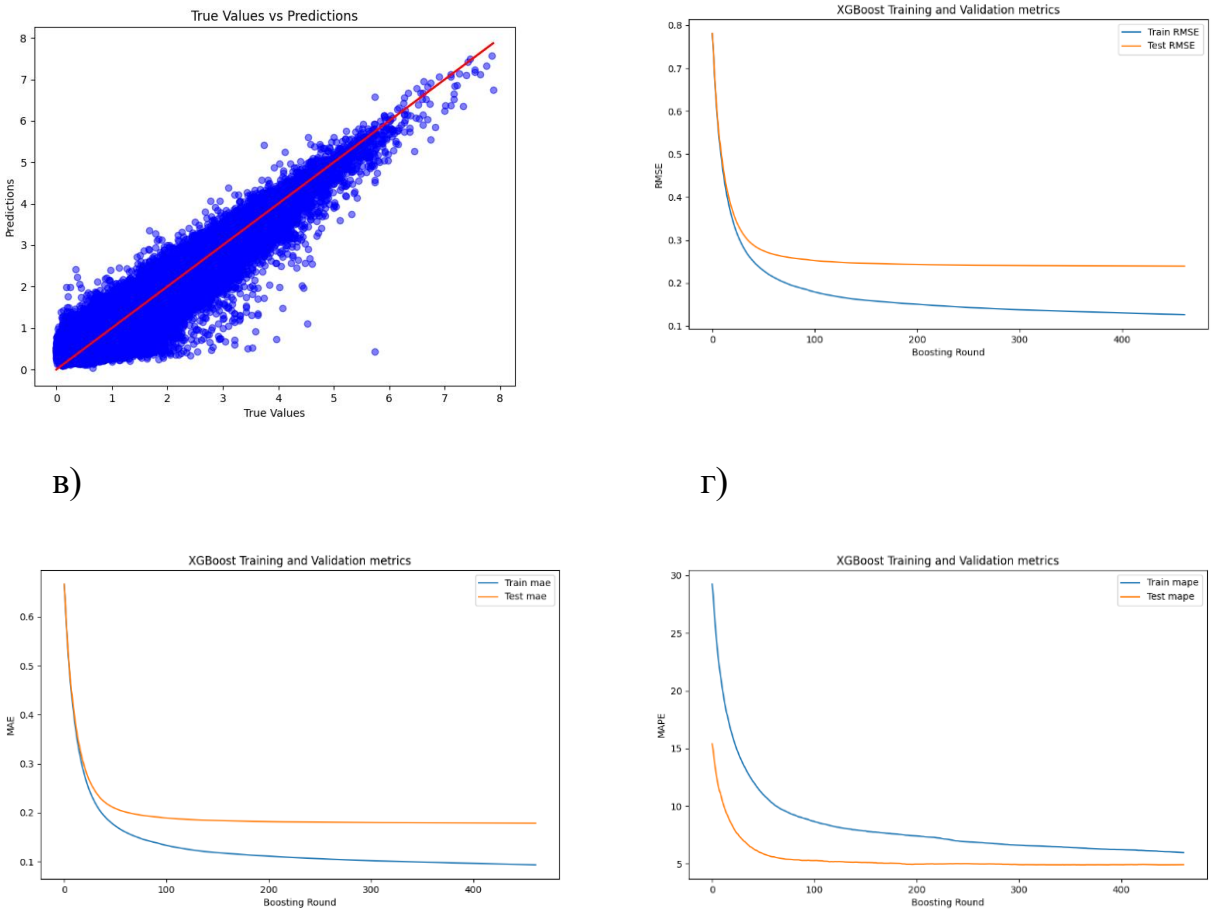
в) г)

Рисунок 12 – Результаты теста №9

На рисунке 12 отображен тест №9, в котором производилось снижение доли признаков. Кривые ошибок на обучающей и тестовой выборках практически совпадают, что говорит о хорошем балансе и отсутствии переобучения. На графике "True Values vs Predictions" сохраняется высокая плотность точек вдоль диагонали, экстремальные значения воспроизводятся адекватно. Уменьшение доли признаков до 0.85 позволило дополнительно снизить RMSE и MAE, а также повысить устойчивость модели к

переобучению. Такой подход способствует лучшей обобщающей способности, особенно при большом количестве входных признаков. Модель становится менее чувствительной к шуму и избыточным данным.

а) б)



в) г)

Рисунок 13 – Результаты теста №10

Тест №10— это комплексная оптимизация всех параметров. Кривые ошибок (RMSE, MAE, MAPE) на обучающей и тестовой выборках полностью совпадают и стабилизируются уже после 200–300 итераций. На графике "True Values vs Predictions" точки максимально сгруппированы вдоль идеальной линии, минимальный разброс даже для экстремальных значений. Это итоговая и наиболее оптимальная конфигурация: минимальные значения ошибок. Модель демонстрирует высокую точность, устойчивость и способность

воспроизводить как средние, так и экстремальные значения высоты волн (рис. 13).

Таким образом, график "True Values vs Predictions" отображает зависимость между истинными значениями значительной высоты волн и предсказанными моделью значениями на тестовой выборке. Для идеальной модели точки располагаются вдоль прямой под углом 45°, что свидетельствует о высокой точности предсказаний. В первых тестах (например, тест №1) наблюдается заметное рассеяние точек относительно идеальной линии, что указывает на наличие систематических ошибок и недостаточную гибкость модели. По мере оптимизации параметров (например, тесты №9, №10) точки группируются ближе к диагонали, уменьшается разброс, что свидетельствует о росте точности и лучшем соответствии модели реальным данным.

На кривых обучения (learning curves) по метрикам RMSE, MAE, MAPE по оси X отложено количество раундов бустинга (итераций), а по оси Y — значения соответствующих ошибок на обучающей и тестовой выборках. В начальных тестах (№1–№4) видно, что снижение ошибок происходит медленно, а для достижения приемлемого уровня требуется большое число итераций (до 1000), что связано с низкой скоростью обучения и недостаточной глубиной деревьев. В тестах с увеличенной глубиной деревьев и оптимизированной скоростью обучения (например, тесты №5, №7, №10) кривая ошибок быстро стабилизируется, а разница между ошибками на обучающей и тестовой выборках становится минимальной. Это говорит о хорошем балансе между обучением и обобщающей способностью модели, а также об отсутствии переобучения. В наиболее удачных конфигурациях (тест №10) кривая тестовой ошибки выходит на плато уже после 200–300 итераций, что позволяет существенно сократить время обучения без потери качества.

Сравнивая результаты разных тестов, В тестах с низкой глубиной деревьев и базовой функцией потерь (№1, №2, №6) наблюдается более широкий разброс предсказаний и заметная разница между обучающей и тестовой ошибками, что указывает на недообучение модели. А при увеличении глубины деревьев (№3, №4, №8) и оптимизации доли признаков наблюдается заметное снижение ошибок, кривая ошибок становится более пологой, а разброс точек на графике "True Values vs Predictions" уменьшается. Наилучшие результаты достигаются в тестах с комплексной оптимизацией всех параметров (№5, №9, №10): здесь кривая тестовой ошибки быстро стабилизируется, а точки на диаграмме "True Values vs Predictions" максимально приближены к идеальной линии. Это свидетельствует о высокой точности и устойчивости модели. В дальнейшем именно модель из теста №10 будет использована для прогнозирования волнового поля.

Анализ всех рисунков показывает, что оптимизация гиперпараметров XGBoost (выбор устойчивой к выбросам функции потерь, увеличение глубины деревьев, корректировка скорости обучения и доли признаков) приводит к существенному улучшению качества прогноза. Кривые обучения позволяют визуально контролировать процесс сходимости модели, выявлять признаки переобучения или недообучения, а также выбирать оптимальное число итераций для остановки обучения. Графики "True Values vs Predictions" подтверждают, что итоговая модель не только хорошо аппроксимирует средние значения, но и адекватно воспроизводит экстремальные значения высоты волн, что особенно важно для задач гидрометеорологического прогнозирования. Визуальный анализ рисунков для всех тестов подтверждает, что грамотная настройка гиперпараметров XGBoost позволяет добиться высокой точности и устойчивости модели.

3.2 Результаты

Для наглядного отображения результатов моделирования было выбрано 10 характерных случаев, соответствующих эпизодам сильного ветра (скорость ветра превышала 10 м/с) в различных районах Балтийского моря. Эти примеры отражают типовые сценарии формирования экстремального волнения и позволяют детально проанализировать пространственное распределение значительной высоты волн, а также особенности работы моделей XGBoost и ConvLSTM в условиях штормовой активности.

Комплексная визуализация результатов прогноза волнового поля на основе машинного обучения в Балтийском море для различных моментов времени представлена на рисунках 14-23. На каждом из представленных рисунков показаны поля скорости и направления ветра по данным ERA5, что позволяет оценить синоптическую ситуацию и локальные особенности ветрового воздействия; поля значительной высоты волн по данным ERA5 — как эталонная модельная информация о состоянии волнения на момент эпизода; поля значительной высоты волн по прогнозу моделей ConvLSTM, обученной на данных реанализа ERA5, что позволяет проанализировать, насколько хорошо нейросетевая модель воспроизводит пространственную структуру волнения; поля значительной высоты волн по прогнозу модели XGBoost, построенной на объединённом массиве спутниковых и реанализных данных, что даёт возможность оценить преимущества и ограничения подхода, использующего интеграцию разнородных источников информации, а также оценить насколько хорошо иная нейросетевая модель воспроизводит пространственную структуру волнения; а также поля ошибок или разности между прогнозом и эталонными данными, что позволяет наглядно выявить зоны переоценки и недооценки высоты волн обеими моделями.

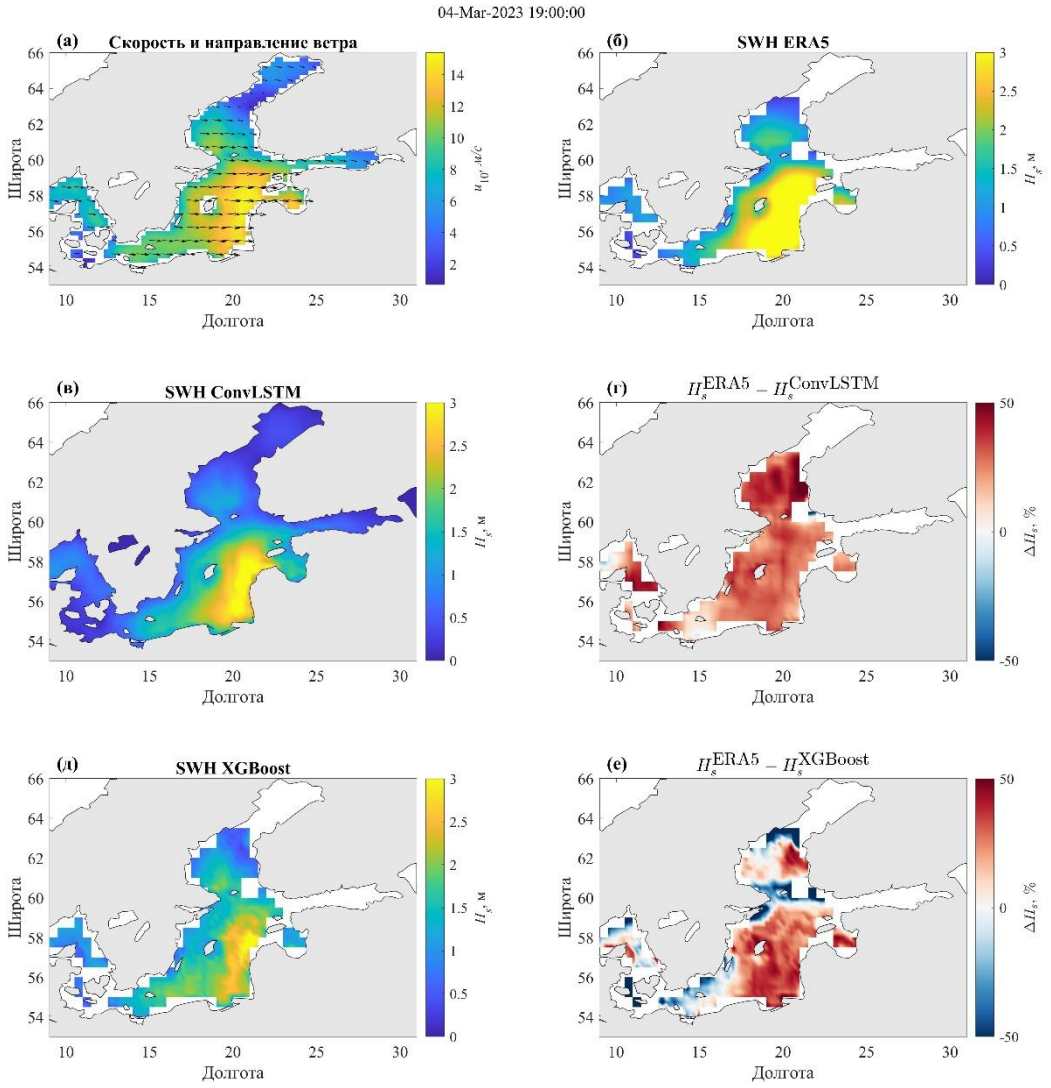


Рисунок 14 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (04.03.2023 19:00)

На рисунке 14 для 04.03.2023 19:00 по данным ERA5 в районе центральной и юго-восточной частей Балтийского моря наблюдаются западные ветры со скоростью, достигающей 14 м/с и более. Эталонные значения значительной высоты волн (ERA5) показывают максимум более 3 м в частях моря, где наблюдаются высокие значения скорости ветра. Модель ConvLSTM достаточно точно воспроизводит пространственную структуру волнения, включая экстремальные значения. XGBoost также фиксирует

основные зоны волнения, однако по сравнению с ConvLSTM хуже воспроизводит карту волнового поля. Анализ ошибок показал, что ConvLSTM недооценивает значения равномерно по всей акватории, а для XGBoost характерна переоценка в прибрежных районах и на мелководьях и недооценка в районах, где наблюдаются высокие значения высоты волн.

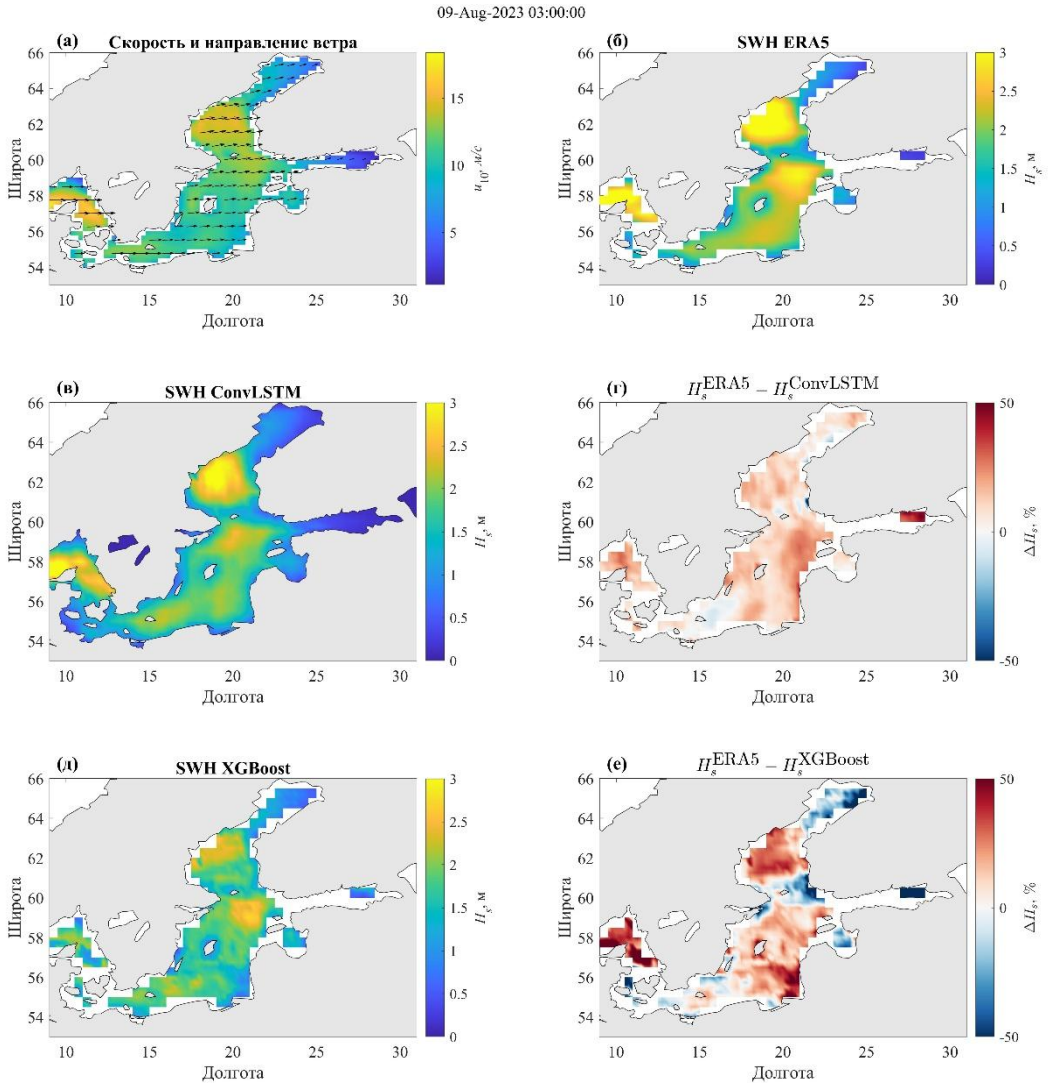
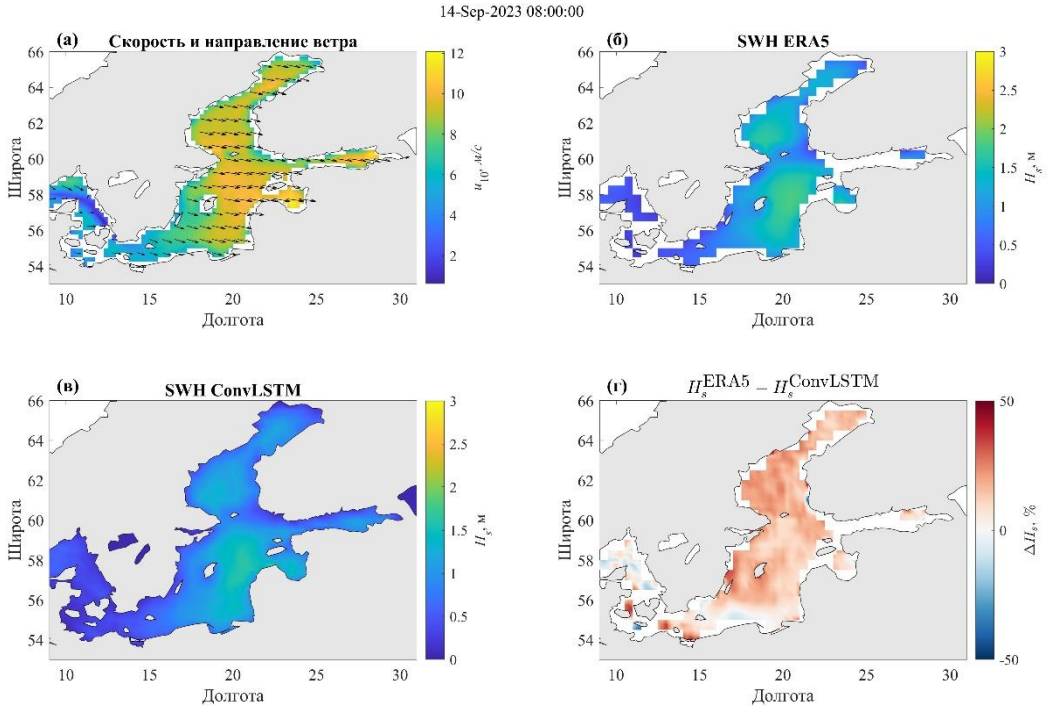


Рисунок 15 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (09.08.2023 03:00)

По данным ERA5 09.08.2023 03:00 преобладают западные и юго-западные ветры достигающие скорости 15 м/с и более. Наиболее высокие значения скорости ветра отмечаются практически во всех частях акватории, кроме районов со сложным рельефом и заливов. В аналогичных областях наблюдаются и высокие значения высоты волн и достигают 2–3 м и более. ConvLSTM хорошо воспроизводит пространственное распределение волнения, особенно в Ботническом заливе, в то время как XGBoost склонен завышать значения в заливах и на отдельных участках побережья и занижать, где эталонные значения были высоки. Ошибки (переоценка) ConvLSTM минимальны (рис. 15)



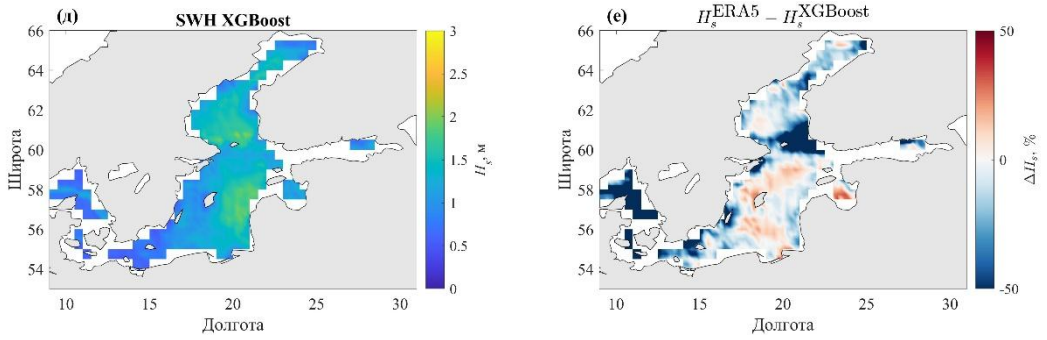
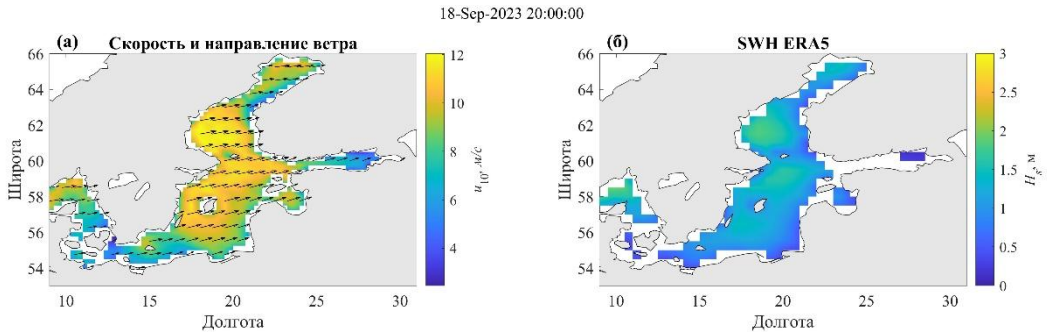


Рисунок 16 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (14.09.2023 08:00)

На рисунке 16 для 14.09.2023 08:00 в центральной части моря и заливах фиксируются сильные ветры 10–12 м/с, направление ветра — с запада (северо-запада) на восток. По данным ERA5 максимум значительной высоты волн превышает 1.5 м в районе глубоководной южной части Ботнического залива и юго-восточной части Балтийского моря. ConvLSTM и XGBoost достаточно точно прогнозируют значения значительной высоты волн. В случае первой модели наблюдается систематическая недооценка равномерно по всей акватории, тогда как XGBoost недооценивает высоту волн в районе, где наблюдаются высокие значения значительной высоты волн, и переоценивает в других частях моря.



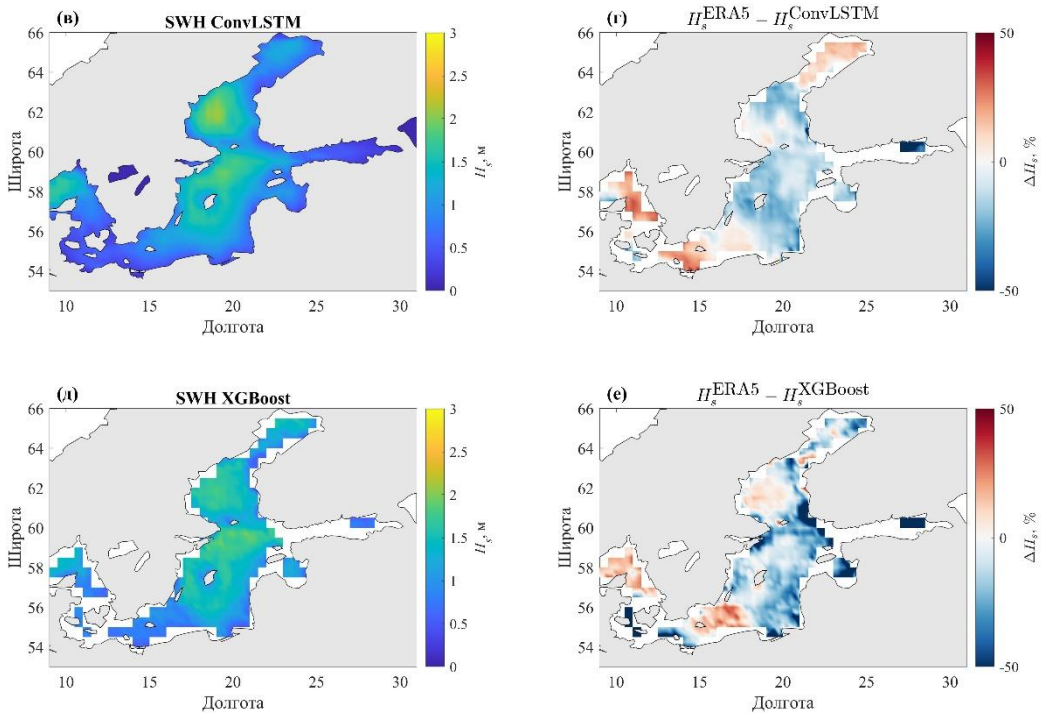


Рисунок 17 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (18.09.2023 20:00)

18.09.2023 в 20:00 в исследуемой акватории преобладают юго-западные ветры 12 и более м/с, максимальные значения отмечаются по всех акватории кроме прибрежных восточных областей. Однако высота волн по эталонным данным невысокая и даже в частях с сильными ветрами достигает 2 м. Обе модели хорошо воспроизводят эталонные данные и склонны в переоценке, причем для модели ConvLSTM ошибки минимальны, а для XGBoost характерна переоценка до в районах с мелководьем или сложным рельефом (рис. 17).

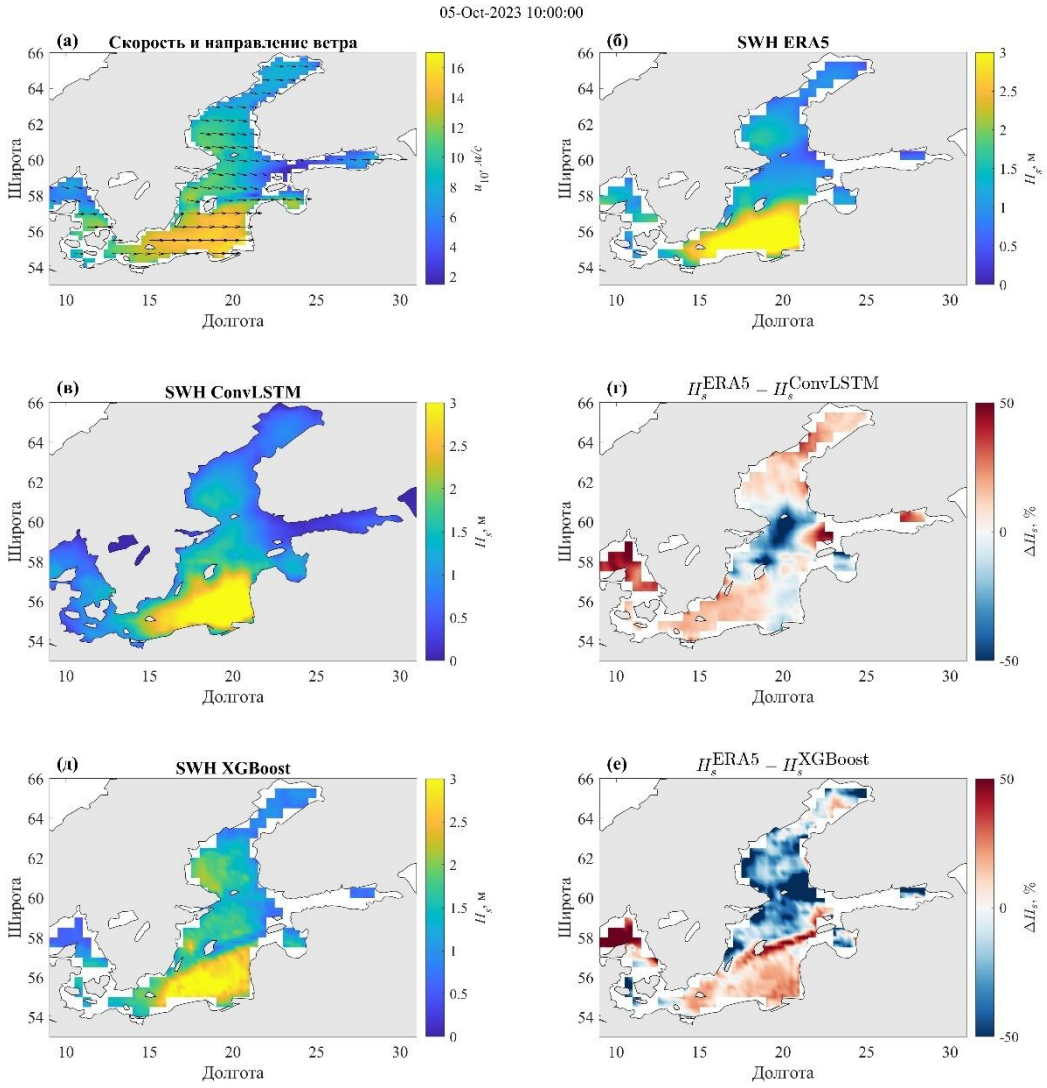


Рисунок 18 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (05.10.2023 10:00)

На рисунке 18 отображены результаты для 05.10.2023 10:00. В этот момент времени также над морем преобладают западные ветры достигающие 16 м/с, особенно сильные потоки наблюдаются в южной части Балтийского моря. По данным ERA5 максимальная высота волн достигает 3 м и более. ConvLSTM хорошо воспроизводит высоту волнения, переоценивая в центральной части, а XGBoost завышает значения в северной части акватории и занижает – в южной.

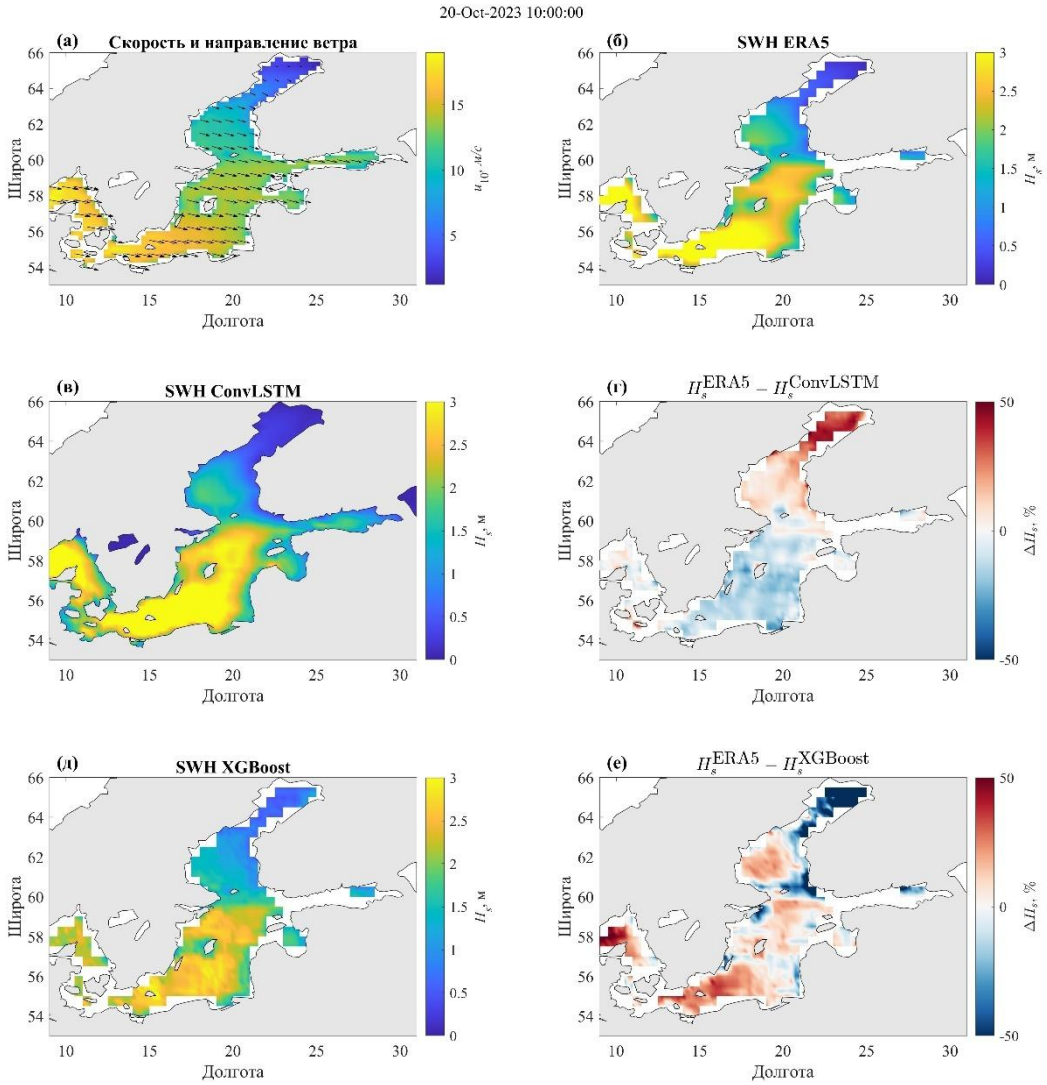


Рисунок 19 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (20.10.2023 10:00)

20 октября 2023 года в 10:00 над Балтийским морем преобладают северо-западные ветры со скоростью, достигающей 15 м/с и более, максимальные значения фиксируются в центральной и южной частях акватории. По данным ERA5, значительная высота волн достигает 2,5–3 м в южной части аквтории. Модель ConvLSTM хорошо воспроизводит пространственное распределение волнения, особенно в глубоководных районах, однако наблюдается легкая

недооценка высоты волн по сравнению с эталонными данными в северной части и переоценка в южной. Модель XGBoost демонстрирует тенденцию к переоценке волнения в заливах и недооценке в центральной и западной частях моря, в то время как в глубоководных районах значения ближе к реальным. На полях ошибок видно, что ConvLSTM обеспечивает более равномерное распределение ошибок по акватории, а для XGBoost характерны локальные зоны завышения высоты волн, особенно у побережья (рис. 19).

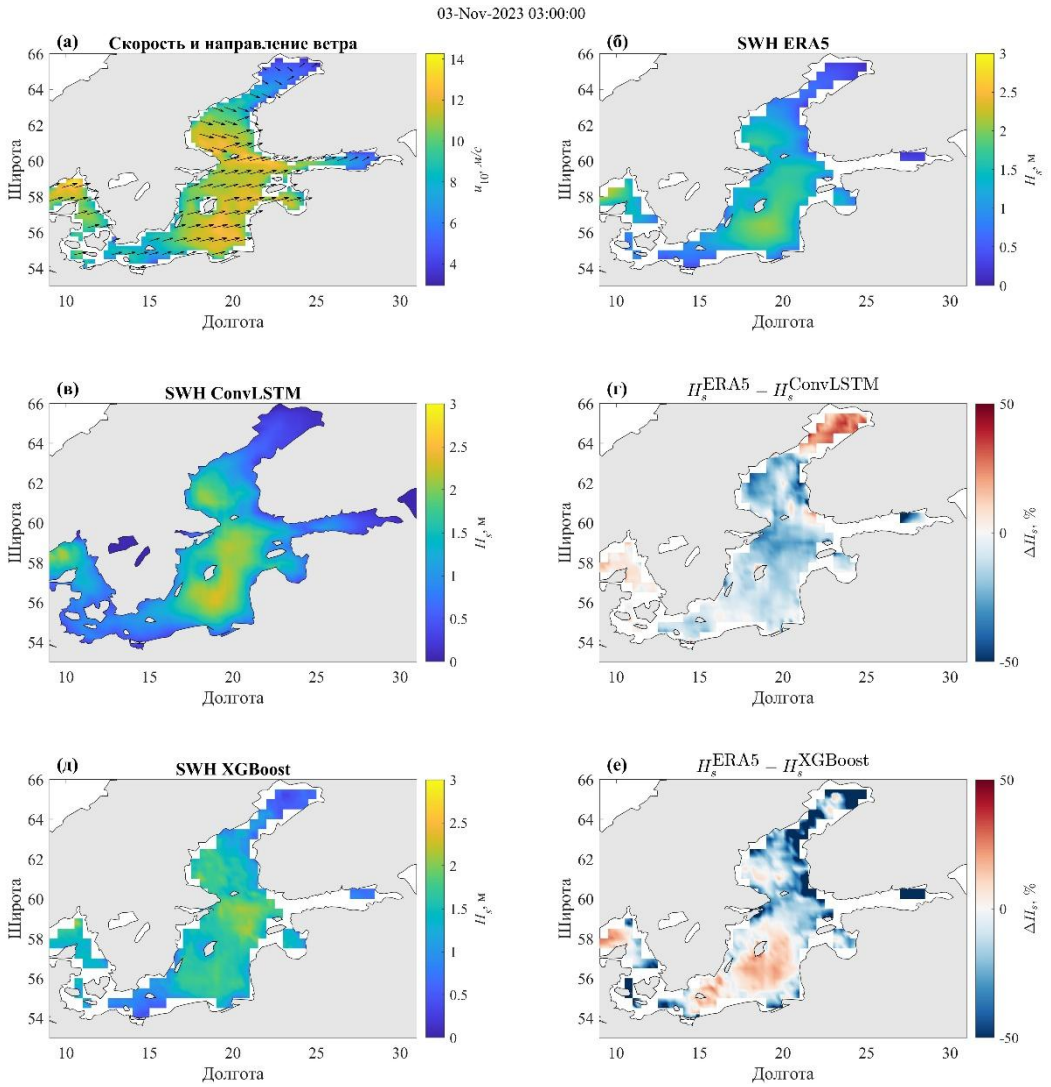
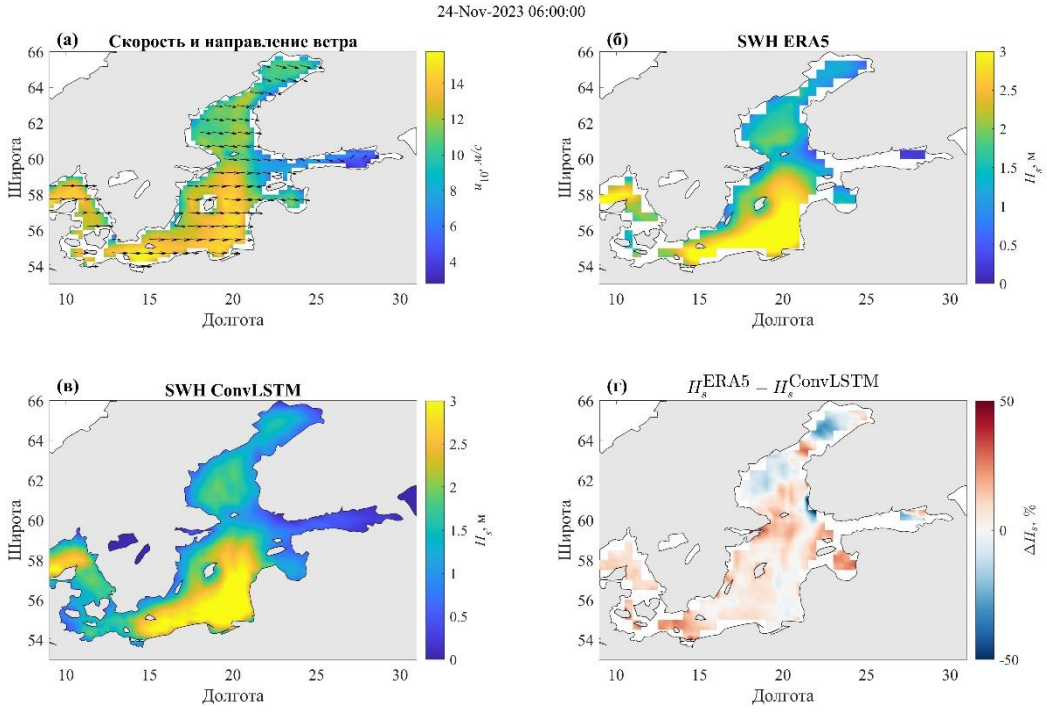


Рисунок 20 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (03.11.2023 03:00)

На рисунке 20 для 3 ноября 2023 года в 03:00 наблюдаются сильные западные ветры до 15 м/с, максимальные значения скорости отмечаются в центральной части Балтийского моря. По данным ERA5, значительная высота волн достигает 2 м в центральной области. ConvLSTM корректно воспроизводит основные зоны волнения с незначительной переоценкой, однако в Ботническом заливе отмечается недооценка. XGBoost склонен к недооценке высоты волн в южных глубоководных районах и переоценке в северо-восточных прибрежных и мелководных районах. В целом, обе модели фиксируют основные закономерности распределения волнения, но ConvLSTM обеспечивает более точное соответствие эталонным данным.



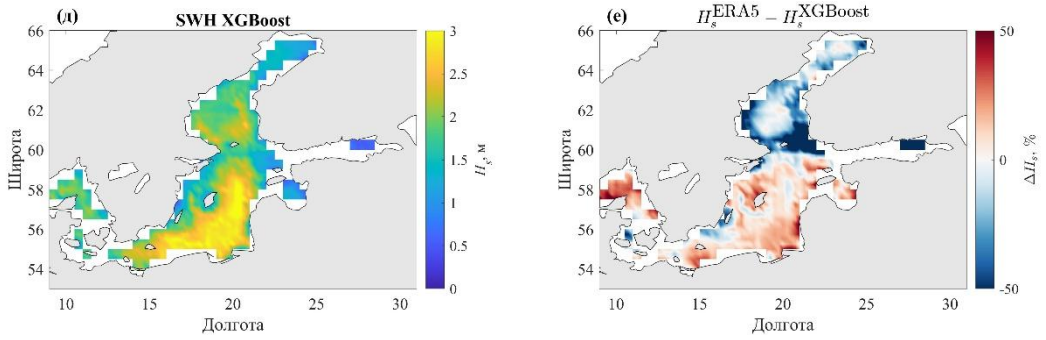
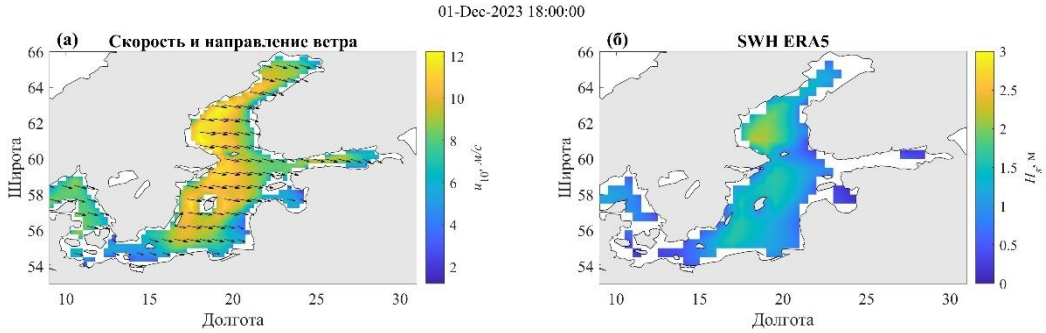


Рисунок 21 − Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (24.11.2023 06:00)

На рисунке 21 для 24 ноября 2023 года в 06:00 над Балтикой преобладают западные и северо-западные ветры 10–14 м/с, максимальные значения скорости отмечаются в южной части моря. По данным ERA5, значительная высота волн достигает 2,5-3 м в южных районах. ConvLSTM и XGBoost стабильно воспроизводят структуру волнения. Причем на графике ошибок для ConvLSTM преимущественно наблюдается незначительная недооценка. XGBoost демонстрирует завышение высоты волн в северных зонах и недооценку в глубоководных южных районах. Анализ полей ошибок вновь показывает, что ConvLSTM обеспечивает более равномерное распределение ошибок, в то время как XGBoost характеризуется локальными зонами переоценки.



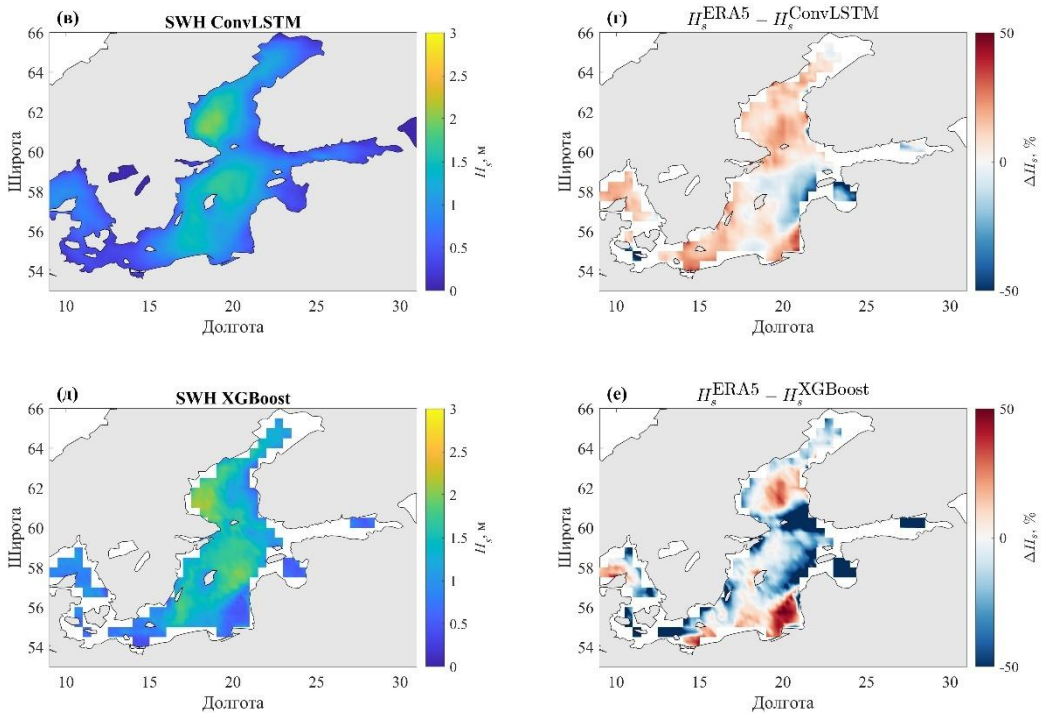


Рисунок 22 − Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (01.12.2023 18:00)

Визуализация результатов для 1 декабря 2023 года отображается на рисунке 22. Здесь в 18:00 фиксируются западные и северо-западные ветры со скоростью 10–12 м/с, максимальные значения отмечаются в центральной и северной частях Балтики. По данным ERA5, значительная высота волн превышает 2 м в южной части Ботнического залива. Обе модели достаточно хорошо воспроизводит экстремальные значения. На полях ошибок видно, что ConvLSTM обеспечивает более точную аппроксимацию эталонных данных с равномерно распределённой по акватории недооценкой и переоценкой в западной части вблизи Рижского залива. XGBoost демонстрирует недооценку в локальных глубоководных районах.

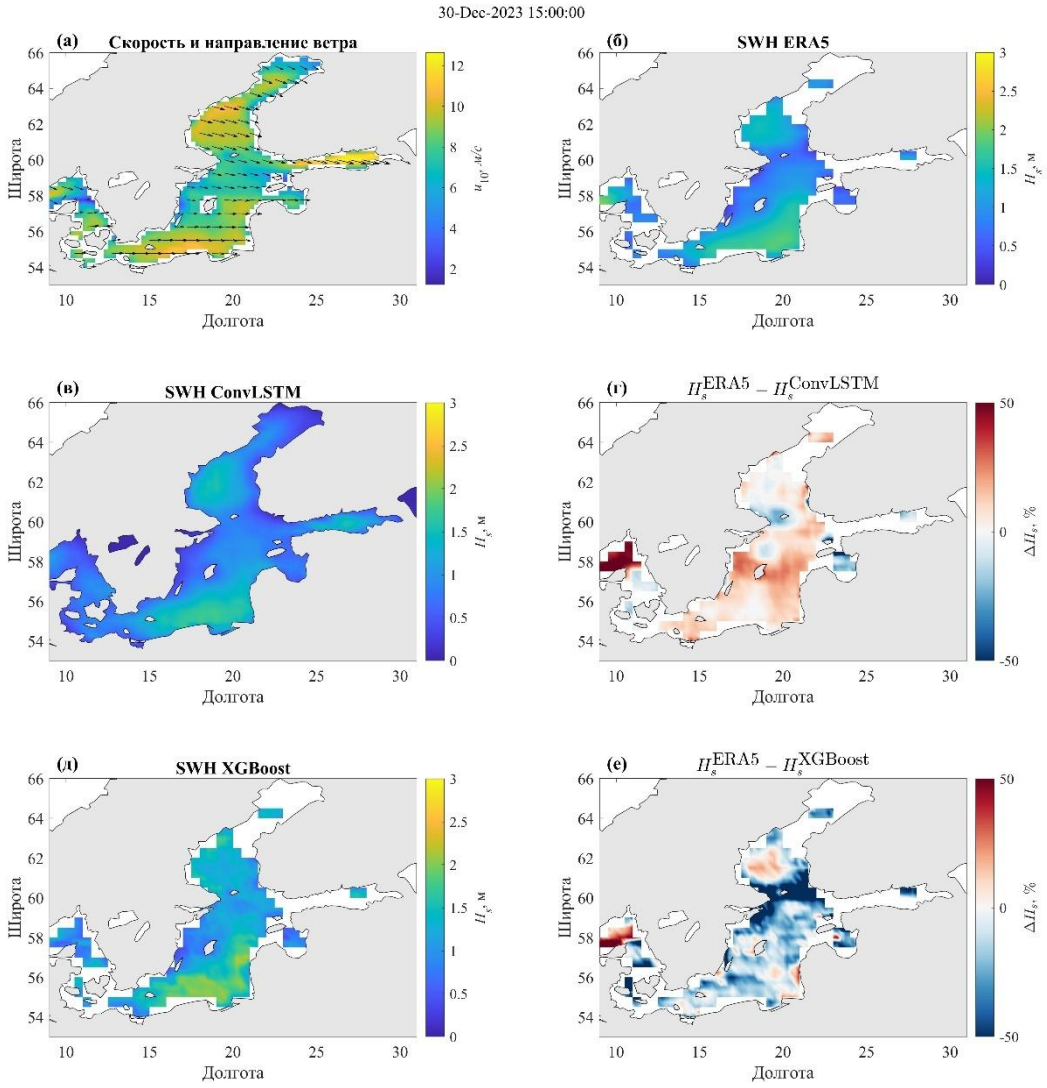


Рисунок 23 – Поля скорости и направления ветра по данным ERA5 (а), значительная высота волн по данным ERA5 (б), прогноз значительной высоты волн по модели ConvLSTM (в), прогноз значительной высоты волн по модели XGBoost (д), а также поля ошибок для обеих моделей (г, е) (30.12.2023 15:00)

30 декабря 2023 года в 15:00 над Балтикой преобладают западные и северо-западные ветры 8–12 м/с, максимальные значения фиксируются северной и южной частях акватории, а также в Финском заливе. По данным ERA5, значительная высота волн достигает 1,5–2 м в южных районах. ConvLSTM точно воспроизводит структуру волнения, особенно в глубоководных областях, с минимальной ошибкой по сравнению с эталонными данными. XGBoost демонстрирует переоценку высоты волн практически во

всех районах. Поля ошибок показывают, что ConvLSTM обеспечивает наименьшие ошибки по акватории (рис. 23).

Визуализация позволяет наглядно оценить, как обе модели воспроизводят пространственную структуру волнения при различных синоптических ситуациях. На большинстве рисунков видно, что ConvLSTM точнее повторяет распределение высоты волн, особенно в центральных и глубоководных районах моря: поля прогнозных значений по ConvLSTM близки к данным ERA5, а ошибки распределены равномерно. В то же время XGBoost имеет тенденцию к переоценке волнения в районах со сложным рельефом, что отражено в положительном смещении ошибок и локальных зонах завышения высоты волн.

3.3 Анализ результатов

В работе реализованы и сравнительно проанализированы две современные модели машинного обучения — ConvLSTM и XGBoost. Прогностическая эффективность моделей оценивалась по четырём ключевым показателям: среднеквадратической ошибке (RMSE), средней абсолютной ошибке (MAE), систематическому смещению (Bias) и коэффициенту корреляции Пирсона.

Модель ConvLSTM, учитывающая как временные, так и пространственные закономерности, показала высокую точность воспроизведения динамики волнового поля (RMSE=0.221 м, MAE=0.153 м, коэффициент корреляции 0.949), что сопоставимо с лучшими результатами современных численных моделей. При этом выявлено незначительное отрицательное смещение (Bias = –0.044 м), обусловленное особенностями реанализа ERA5. Модель XGBoost, построенная на объединённом массиве спутниковых и реанализных данных, продемонстрировала приемлемую

точность (RMSE=0.590 м, MAE=0.392 м, корреляция 0.699, Bias = 0.126 м), однако была склонна к переоценке волнения в районах со сложным рельефом, мелководных и прибрежных областях, что связано с неоднородностью исходных данных и влиянием локальных факторов. Вдобавок, наблюдаемые различия в точности и систематических смещениях моделей ConvLSTM и XGBoost во многом объясняются характеристиками данных, использованных для их обучения. Модель ConvLSTM обучалась исключительно на данных реанализа ERA5, которые, согласно множественным исследованиям, имеют тенденцию к систематическому занижению значительной высоты волн. В отличие от ConvLSTM, модель XGBoost обучалась на объединённом массиве спутниковых альтиметрических данных и данных реанализа ERA5. Спутниковые альтиметрические измерения обладают высокой точностью и часто показывают значения высоты волн выше, чем данные реанализа.

Визуальный и анализ результатов моделирования для различных штормовых эпизодов (рис. 14-23) показал, что модели машинного обучения способны не только воспроизводить средние значения значительной высоты волн, но и адекватно описывать экстремальные события, что критически важно для задач обеспечения безопасности мореплавания, портовой инфраструктуры и мониторинга прибрежных территорий.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была решена актуальная задача прогнозирования значительной высоты волн в Балтийском море с применением современных методов машинного обучения и интеграции разнородных источников данных — спутниковых альтиметрических измерений и реанализа. Проведён комплексный анализ существующих подходов, обоснован выбор моделей ConvLSTM и XGBoost, реализована процедура объединения и калибровки данных, что позволило сформировать репрезентативную обучающую выборку для построения прогностических моделей.

В результате численных экспериментов и сравнительного анализа установлено, что модель ConvLSTM, способная учитывать как временные, так и пространственные закономерности, обеспечивает высокую точность прогноза динамики волнового поля, что сопоставимо с лучшими результатами современных численных моделей. Модель XGBoost, построенная на интегрированной выборке спутниковых и реанализных данных, показала приемлемую точность, а комплексная оптимизация гиперпараметров позволила снизить ошибки прогноза на 12,5% по сравнению с базовой конфигурацией. Визуальный и количественный анализ результатов моделирования для различных штормовых эпизодов подтвердил, что обе модели способны воспроизводить как средние, так и экстремальные значения высоты волн, что особенно важно для задач обеспечения безопасности мореплавания, эксплуатации портовой инфраструктуры и мониторинга прибрежных территорий.

Наибольшие расхождения с эталонными данными отмечаются в прибрежных районах, что связано с высокой изменчивостью локальных ветровых условий, сложным рельефом дна и неоднородностью распределения наблюдений. Это определяет направления для дальнейших исследований:

интеграция локальных гидродинамических моделей, повышение пространственного разрешения данных и расширение набора признаков для машинного обучения.

В целом, выполненное исследование убедительно демонстрирует эффективность современных методов машинного обучения для прогноза волнового поля в сложных акваториях, таких как Балтийское море, и закладывает основу для дальнейшего развития цифровых технологий в гидрометеорологии и океанологии. Полученные результаты могут быть рекомендованы для внедрения в системы оперативного прогноза волнения, а предложенные методики — для совершенствования инструментов мониторинга и управления морской деятельностью в регионе.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Гидрометеорология и гидрохимия морей СССР. Т. 3: Балтийское море. Вып. 1: Гидрометеорологические условия / под ред. Ф. С. Терзиева, В. А. Рожкова, А. И. Смирновой. — СПб.: Гидрометеоиздат, 1992. — 451 с.
2. Гидрометеорология и гидрохимия морей. Том III: Балтийское море. Вып. 2: Гидрохимические условия и океанологические основы формирования биологической продуктивности / под ред. Ф. С. Терзиева, В. А. Рожкова, Е. Я. Римша, И. С. Шпаер. — СПб.: Гидрометеоиздат, 1994. — 432 с.
3. Добровольский А.Д., Залогин Б.С. Моря СССР. — М.: . Моря СССР. М., Изд-во МГУ, 1982 г. С ил., 192 с
4. Дубравин В.Ф. Эволюции термохалинной структуры вод Балтийского моря. — М.: Перо, 2017. — 437 с. ил., карт., табл.; 25. — ISBN 978-5-907016-56-9.
5. Единая государственная система информации об обстановке в Мировом океане (ЕСИМО). URL:http://esimo.oceanography.ru/esp2/index/index/esp_id/1/section_id/7/menu_id/4583 (дата обращения: 01.12.2024).
6. Захарчук Е. А., Тихонова Н. А., Сухачев В. Н. Волновая природа и модуляция годовых колебаний уровня Балтийского моря // Морской гидрофизический журнал. 2024. №2 (236) . URL:https://cyberleninka.ru/article/n/volnovaya-priroda-i-modulyatsiya-godovyh-kolebaniy-urovnya-baltiyskogo-morya
(дата обращения: 01.12.2024).
7. Машинное обучение для решения исследовательских и инженерных задач в науках о Земле // Институт океанологии РАН. – URL: https://ocean.ru/index.php/basic-chairs/item/1441-mashinnoe-obuchenie-

dlya-resheniya-issledovatelskikh-i-inzhenernykh-zadach-v-naukakh-o-zemle (дата обращения: 16.01.2025).

8. Машинное обучение помогает определять солёность моря в Арктике // Научная Россия. – URL: https://scientificrussia.ru/articles/masinnoe-obucenie-pomogaet-opredelat-solenost-mora-v-arktike (дата обращения: 16.01.2025).
9. Машинное обучение открывает более чёткое представление о движении океана // Климатический центр Росгидромета – URL: https://cc.voeikovmgo.ru/ru/novosti/novosti-partnerov/2497-eos-mashinnoe-obuchenie-otkryvaet-bolee-chjotkoe-predstavlenie-o-dvizhenii-okeana (дата обращения: 16.01.2025).
10. Медведева, А. Ю. Волновой климат Балтийского моря / А. Ю. Медведева, С. А. Мысленков, В. С. Архипкин // Комплексные исследования Мирового океана : материалы II Всероссийской научной конференции молодых ученых, г. Москва, 10–14 апреля 2017 года. – г. Москва: Институт океанологии им. П.П. Ширшова Российской академии наук, 2017. – С. 173-176. – EDN ZHWNPH.
11. Медведева А.Ю., Мысленков С.А., Медведев И.П., Архипкин В.С., Кречик В.А., Добролюбов С.А. Моделирование ветрового волнения в Балтийском море на прямоугольной и неструктурной сетках на основе реанализа NCEP/CFSR. — 18 с. URL:https://method.meteorf.ru/publ/tr/tr362/medv.pdf (дата обращения: 01.12.2024).
12. Морские гидродинамические модели Санкт-Петербургского ЦГМС-Р. – URL:http://www.meteo.nw.ru/articles/index.php?id=532 (дата обращения: 08.11.2024).

13. Официальный сайт проекта "Северный поток". URL:https://www.gazprom.ru/projects/nord-stream/ (дата обращения: 10.03.2025).

14. Справочные данные по режиму ветра и волнения Балтийского, Северного, Черного, Азовского и Средиземного морей / Л.И. Лопатухин, А.В. Бухановский, С.В. Иванов, Е.С. Чернышева; Российский морской регистр судоходства. — СПб.: Российский морской регистр судоходства, 2006. — 450 с. ISBN 5-89331-071-3.

15. ERA5 hourly data on single levels from 1940 to present URL:https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels?tab=overview (дата обращения: 08.12.2024).

16. Global Ocean L 4 Significant Wave Height From Nrt Satellite Measurements. URL:https://data.marine.copernicus.eu/product/WAVE_GLO_PHY_SWH_L4_NRT_014_003/description (дата обращения: 18.02.2025)

17. James S.C., Zhang Y., O'Donncha F. A machine learning framework to forecast wave conditions // Coastal Engineering. 2018. Vol. 137. P. 1–10.

18. Hersbach H. и др. The ERA5 global reanalysis // Quarterly Journal of the Royal Meteorological Society. 2020. Vol. 146, No. 730. P. 1999–2049. DOI: 10.1002/qj.3803. URL:https://doi.org/10.1002/qj.3803 (дата обращения: 08.04.2025).

19. Hu H., van der Westhuysen A.J., Chu P., Fujisaki-Manome A. Wave height and period forecasting on Lake Erie using XGBoost and LSTM // Ocean Modelling. 2021. - URL: https://doi.org/10.1016/j.ocemod.2021.101832 (дата обращения: 08.04.2025).

20. Minuzzi F.C., Farina L. A deep learning approach to predict significant wave height using long short-term memory // Ocean Modelling. 2023. Vol. 181. Article 102151.

21. NVIDIA. XGBoost. URL:https://www.nvidia.com/en-us/glossary/xgboost/ (дата обращения: 22.11.2024).

22. Riswanda A.D., Safira A., Fahmiyah I., Ghani M. Ocean wave prediction using Long Short-Term Memory (LSTM) and Extreme Gradient Boosting (XGBoost) in Tuban Regency for fisherman safety. - URL:https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11639740/ (дата обращения: 08.04.2025).

23. Shi X., Chen Z., Wang H., Yeung D.Y., Wong W., Woo W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. P. 1-9. URL:https://proceedings.neurips.cc/paper_files/paper/2015/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf (дата обращения: 08.04.2025).

24. Wang X., Wang J., Lu W., Dong C., Qin H., Jiang H. Data-driven rolling model for global wave height // Earth System Science Data. 2024. Vol. 16, № 1. P. 123–140.

25. Yang J., Zhang T., Zhang J., Lin X., Wang H., Feng T. A ConvLSTM nearshore water level prediction model with integrated attention mechanism // Frontiers in Marine Science. 2024. Vol. 11. Article 1470320. DOI: 10.3389/fmars.2024.1470320. URL: https://doi.org/10.3389/fmars.2024.1470320 (дата обращения: 08.04.2025)

26. Yu H., Li J., Zhai X., et al. A global high-resolution ocean wave model improved by assimilating the satellite altimeter significant wave height // International Journal of Applied Earth Observation and Geoinformation. 2018. Vol. 70. P. 43–50.

ПРИЛОЖЕНИЕ А

Код для расчета модели ConvLSTM

```
import tensorflow as tf
import os
os.environ["OMP_NUM_THREADS"] = "4"  # Limits OpenMP to 4 threads
os.environ["TF_NUM_INTEROP_THREADS"] = "4"
os.environ["TF_NUM_INTRAOP_THREADS"] = "4"
tf.config.threading.set_inter_op_parallelism_threads(4)
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '4'
import numpy as np
import xarray as xr
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import ConvLSTM2D, BatchNormalization, Conv3D, Dropout, Input, Layer, Reshape
from tensorflow.keras.optimizers import Adam
import matplotlib.pyplot as plt
import time
from sklearn.preprocessing import MinMaxScaler
from scipy.io import savemat, loadmat
import joblib
import gc
# Save timestamp
start = time.time()
# Load and interpolate data
def load_data(wind_file):
```

```
    # Load the datasets
    wind_data = xr.open_dataset(wind_file)
    wind_u = wind_data['U10'].values
    wind_v = wind_data['phi_w'].values
    wave = wind_data['swh'].values
    return wind_u, wind_v, wave
def apply_mask_to_wave_data(wave_data, id_msk, mask_value=3):
    id_msk_broadcasted = np.repeat(id_msk[np.newaxis, :, :], wave.shape[0],
axis=0)
    # Create a mask where id_msk != mask_value
    mask = id_msk_broadcasted != mask_value
    # Apply the mask by setting wave data to NaN where the condition is True
    masked_wave_data = wave_data.copy()  # Make a copy to preserve the original
data
    masked_wave_data[mask] = np.nan      # Set values to NaN where the condition
is met
        return masked_wave_data
# Custom Masking Layer
class NaNMaskingLayer(Layer):
    def __init__(self, **kwargs):
        super(NaNMaskingLayer, self).__init__(**kwargs)
    def call(self, inputs):
        # Create a mask where valid data is True and NaN is False
        mask = tf.math.is_finite(inputs)
        # Mask out invalid regions by setting them to zero
        masked_data = tf.where(mask, inputs, tf.zeros_like(inputs))
        return masked_data
def tf_nanmean(tensor):
    """Computes the mean while ignoring NaNs, equivalent to MATLAB's
nanmean."""
```

```
    mask = tf.math.is_finite(tensor)  # Create mask for valid values
    sum_valid = tf.reduce_sum(tf.where(mask, tensor, tf.zeros_like(tensor)))  # Sum of valid values
    count_valid = tf.reduce_sum(tf.cast(mask, tf.float32))  # Count of valid values
    return sum_valid / count_valid  # Compute mean only over valid values
def nan_mse_loss(y_true, y_pred):
    """Mean Squared Error loss while ignoring NaNs in y_true."""
    mask = tf.math.is_finite(y_true)  # Mask NaN values
    y_true = tf.where(mask, y_true, tf.zeros_like(y_true))  # Replace NaNs with 0
    y_pred = tf.where(mask, y_pred, tf.zeros_like(y_pred))  # Replace NaNs with 0
    mse = tf.square(y_true - y_pred)  # Compute squared error
    return tf_nanmean(mse)  # Compute mean only over valid values
def nan_mae(y_true, y_pred):
    mask = tf.math.is_finite(y_true)  # Create mask for valid values
    y_true_masked = tf.where(mask, y_true, tf.zeros_like(y_true))
    y_pred_masked = tf.where(mask, y_pred, tf.zeros_like(y_pred))
    mae = tf.abs(y_true_masked - y_pred_masked)
    return tf_nanmean(mae)
# Prepare data for ConvLSTM
def prepare_data(wind_u, wind_v, wave, seq_length=3, pred_length=1):
    if len(wave.shape) == 3:  # (time, lat, lon)
        wave = wave[np.newaxis, ...]  # Shape becomes (1, time, lat, lon)
    samples, timesteps, lat, lon = wave.shape
    X, y = [], []
    for t in range(seq_length, timesteps - pred_length):
        # Extract slices of wave and wind data
        wave_slice = wave[:, t - seq_length:t]
        wind_u_slice = wind_u[t - seq_length:t, :, :]
        wind_v_slice = wind_v[t - seq_length:t, :, :]
```

```
        # Ensure wind slices have a batch/sample axis
        wind_u_slice = wind_u_slice[np.newaxis, ...]
        wind_v_slice = wind_v_slice[np.newaxis, ...]
        # Ensure all slices have matching shapes
        if wave_slice.shape[1:] != wind_u_slice.shape[1:]:
            raise ValueError(f"Shape mismatch: wave_slice={wave_slice.shape},
wind_u_slice={wind_u_slice.shape}")
        # Stack wave and wind components along the last axis
        X_seq = np.stack([wind_u_slice[0], wind_v_slice[0]], axis=-1)
        y_seq = wave[:, t:t + pred_length]  # Target wave heights
        X.append(X_seq)
        y.append(y_seq)
    X = np.array(X)  # Final shape: (samples, seq_length, lat, lon, features)
    y = np.array(y)  # Final shape: (samples, pred_length, lat, lon)
    return X, y
# Build ConvLSTM model with Masking Layer
def build_model(input_shape):
    model = Sequential([
        Input(shape=input_shape),  # Define the input shape here
        NaNMaskingLayer(),  # Mask NaN values
        ConvLSTM2D(filters=64, kernel_size=(3,3), activation='relu',
                   return_sequences=True, padding='same'),
        ConvLSTM2D(filters=32, kernel_size=(3,3), activation='relu',
return_sequences=True, padding='same'),


        ConvLSTM2D(filters=16, kernel_size=(3,3), activation='relu',
return_sequences=False, padding='same'),
        Reshape((1, input_shape[1], input_shape[2], 16)),
        Conv3D(filters=8, kernel_size=(3, 3, 3), activation='relu', padding='same')
    ])
```

```
    optimizer = Adam(learning_rate=0.0001, clipvalue=1.0)
    model.compile(optimizer=optimizer, loss=nan_mse_loss, metrics=[nan_mae])
    return model
# Train model
def train_model(model, X_train, y_train, X_val, y_val, epochs, batch_size):
    early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience = 10, restore_best_weights=True)
    history = model.fit(X_train, y_train,
                        validation_data=(X_val, y_val),
                        epochs=epochs,
                        batch_size=batch_size,
                        callbacks=[early_stopping], verbose=0, workers=4, use_multiprocessing=True)
    return history
# Predict and evaluate
def predict(model, X_val):
    y_pred = model.predict(X_val)
    return y_pred
# Load data: Data should incloud wind speed, wind direction, or u and v components of wind
# and swh. NOTE: swh and wind field must have the same spatial resolution.
wind_file_1 = 'H:\\for Maria\\codes\\era5_full_2016_2023.nc'
mask_file_path = 'H:\\for Maria\\codes\\mask.mat'
wind_u, wind_v, wave = loaddata(wind_file_1)
mask = loadmat(mask_file_path)
id_msk = mask['id_msk']
wave = apply_mask_to_wave_data(wave, id_msk)
wind_u = apply_mask_to_wave_data(wind_u, id_msk)
wind_v = apply_mask_to_wave_data(wind_v, id_msk)
# prepare data for the model with 4 hours temporal resolusion (just to not occupy RAM)
```

```
u_wind_seq = wind_u[1::4]

v_wind_seq = wind_v[1::4]

swh_seq = wave[1::4]

del wind_v, wind_u, wave

# Prepare the data for training

X, y = prepare_data(u_wind_seq, v_wind_seq, swh_seq, seq_length=3,
pred_length=1)

X = X.reshape((-1, X.shape[1], X.shape[2], X.shape[3], X.shape[4]))  # Ensure X
has the right shape

y = y.reshape((-1, 1, y.shape[3], y.shape[4], 1))  # For 1 time step

# Check shapes

print("Input data shape (X):", X.shape)

print("Target data shape (y):", y.shape)

# Build and compile the model

input_shape = X.shape[1:]  # Shape without batch dimension

model = build_model(input_shape)

model.summary()


# Train, evaluate, and test

X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3,
random_state=42)

X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.1,
random_state=42)

scaler = MinMaxScaler()

scaler.fit(y_train.reshape(-1, y_train.shape[-1]))

# Train the model

history = train_model(model, X_train, y_train, X_val, y_val, epochs=30,
batch_size=16)

# Predict on validation data

y_pred = predict(model, X_val)

y_test_pred = predict(model, X_test)

test_loss, test_mae = model.evaluate(X_test, y_test)
```

```
print(f"Test Loss: {test_loss}, Test MAE: {test_mae}")
# Plot training and validation loss
plt.figure()
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.savefig('training_loss.png')  # Save instead of show
plt.close()  # Close the figure
# Plot predicted and true values
time_step = 1  # Adjust the time step as needed
y_val_grid = y_val[0, 0, :, :, 0]
y_pred_grid = y_pred[0, 0, :, :, 0]
fig, axes = plt.subplots(1, 3, figsize=(12, 6))
im_pred = axes[0].imshow(y_pred_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[0].set_title("Predicted SWH")
axes[0].axis("off")
fig.colorbar(im_pred, ax=axes[0], label="SWH")
im_true = axes[1].imshow(y_val_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[1].set_title("True SWH")
axes[1].axis("off")
fig.colorbar(im_true, ax=axes[1], label="SWH")
im_diff = axes[2].imshow(y_val_grid - y_pred_grid, cmap="viridis", aspect="auto", vmin=-0.5, vmax=0.5)
axes[2].set_title("Difference SWH")
axes[2].axis("off")
fig.colorbar(im_diff, ax=axes[2], label="True - Predicted")
plt.tight_layout()
```

```
plt.savefig('swh_predictions.png')  # Save instead of show
plt.close()  # Close figure
y_val_grid = y_test[0, 0, :, :, 0]
y_pred_grid = y_test_pred[0, 0, :, :, 0]
fig, axes = plt.subplots(1, 3, figsize=(12, 6))
im_pred = axes[0].imshow(y_pred_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[0].set_title("test Predicted SWH")
axes[0].axis("off")
fig.colorbar(im_pred, ax=axes[0], label="SWH")
im_true = axes[1].imshow(y_val_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[1].set_title("True SWH")
axes[1].axis("off")
fig.colorbar(im_true, ax=axes[1], label="SWH")
im_diff = axes[2].imshow(y_val_grid - y_pred_grid, cmap="viridis", aspect="auto", vmin=-0.5, vmax=0.5)
axes[2].set_title("Difference SWH")
axes[2].axis("off")
fig.colorbar(im_diff, ax=axes[2], label="True - Predicted")
plt.tight_layout()
plt.savefig('swh_predictions_test.png')  # Save instead of show
plt.close()  # Close figure
# Save results
file_name = 'matlab_out.mat'
savemat(file_name, {'y_val':y_val, 'y_pred':y_pred, 'X_val':X_val, 'X_train':X_train, 'X_test':X_test, 'y_test':y_test, 'X_temp':X_temp, 'y_temp':y_temp})
model.save("ConvLSTM_model.h5")
joblib.dump(model, 'ConvLSTM_model.pkl')
loss = history.history['loss']
val_loss = history.history['val_loss']
```

```
mae = history.history['nan_mae']
val_mae = history.history['val_nan_mae']
history_dict = {
    'loss': loss,
    'val_loss': val_loss,
    'mae': mae,
    'val_mae': val_mae
}
savemat('history.mat', {'history':history_dict})
# Save timestamp
endtime = time.time()
print('calculation time (minites)',(endtime - start)/60)
# calculation time (minites) on 07.02.2025 was 3194.6058395465216 minutes for ERA5 data (2016.01.01 00-2023.12.31 23)
import tensorflow as tf
import os
os.environ["OMP_NUM_THREADS"] = "4"  # Limits OpenMP to 4 threads
os.environ["TF_NUM_INTEROP_THREADS"] = "4"
os.environ["TF_NUM_INTRAOP_THREADS"] = "4"
tf.config.threading.set_inter_op_parallelism_threads(4)
tf.config.threading.set_intra_op_parallelism_threads(4)
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'
##from tensorflow.keras.mixed_precision import set_global_policy
##set_global_policy('float16')
import numpy as np
import xarray as xr
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import ConvLSTM2D, BatchNormalization, Conv3D, Dropout, Input, Layer, Reshape
from tensorflow.keras.optimizers import Adam
```

```
import matplotlib.pyplot as plt
import time
from sklearn.preprocessing import MinMaxScaler
from scipy.io import savemat, loadmat
import joblib
from tensorflow.keras.models import load_model


def load_and_interpolate_data(wind_file, starttime, endtime):
    # Load the datasets
    wind_data = xr.open_dataset(wind_file)
    subset = wind_data.isel(time=slice(starttime, endtime))
    wind_u = subset['U10'].values
    wind_v = subset['phi_w'].values
    wave = subset['swh'].values
    return wind_u, wind_v, wave
def create_sequences(data, seq_length=3):
    sequences = []
    for i in range(len(data) - seq_length + 1):
        sequences.append(data[i : i + seq_length])
    return np.array(sequences)
def apply_mask_to_wave_data(wave_data, id_msk, mask_value=3):
    id_msk_broadcasted = np.repeat(id_msk[np.newaxis, :, :], wave.shape[0],
axis=0)
    # Create a mask where id_msk != mask_value
    mask = id_msk_broadcasted != mask_value
    # Apply the mask by setting wave data to NaN where the condition is True
    masked_wave_data = wave_data.copy()  # Make a copy to preserve the original
data
    masked_wave_data[mask] = np.nan      # Set values to NaN where the condition
is met
    return masked_wave_data
```

```
# Custom Masking Layer
class NaNMaskingLayer(Layer):
    def __init__(self, **kwargs):
        super(NaNMaskingLayer, self).__init__(**kwargs)
    def call(self, inputs):
        # Create a mask where valid data is True and NaN is False
        mask = tf.math.is_finite(inputs)
        # Mask out invalid regions by setting them to zero
        masked_data = tf.where(mask, inputs, tf.zeros_like(inputs))
        return masked_data
def tf_nanmean(tensor):
    """Computes the mean while ignoring NaNs, equivalent to MATLAB's nanmean."""
    mask = tf.math.is_finite(tensor)  # Create mask for valid values
    sum_valid = tf.reduce_sum(tf.where(mask, tensor, tf.zeros_like(tensor)))  # Sum of valid values
    count_valid = tf.reduce_sum(tf.cast(mask, tf.float32))  # Count of valid values
    return sum_valid / count_valid  # Compute mean only over valid values
def nan_mse_loss(y_true, y_pred):
    """Mean Squared Error loss while ignoring NaNs in y_true."""
    mask = tf.math.is_finite(y_true)  # Mask NaN values
    y_true = tf.where(mask, y_true, tf.zeros_like(y_true))  # Replace NaNs with 0
    y_pred = tf.where(mask, y_pred, tf.zeros_like(y_pred))  # Replace NaNs with 0
    mse = tf.square(y_true - y_pred)  # Compute squared error
    return tf_nanmean(mse)  # Compute mean only over valid values
def nan_mae(y_true, y_pred):
    mask = tf.math.is_finite(y_true)  # Create mask for valid values
    y_true_masked = tf.where(mask, y_true, tf.zeros_like(y_true))
    y_pred_masked = tf.where(mask, y_pred, tf.zeros_like(y_pred))
    mae = tf.abs(y_true_masked - y_pred_masked)
    return tf_nanmean(mae)
```

```
def prepare_data(wind_u, wind_v, wave, seq_length=3, pred_length=1):
    if len(wave.shape) == 3:  # (time, lat, lon)
        wave = wave[np.newaxis, ...]  # Shape becomes (1, time, lat, lon)
    samples, timesteps, lat, lon = wave.shape
    X, y = [], []
    for t in range(seq_length, timesteps - pred_length):
        # Extract slices of wave and wind data
        wave_slice = wave[:, t - seq_length:t]
        wind_u_slice = wind_u[t - seq_length:t, :, :]
        wind_v_slice = wind_v[t - seq_length:t, :, :]
        # Ensure wind slices have a batch/sample axis
        wind_u_slice = wind_u_slice[np.newaxis, ...]
        wind_v_slice = wind_v_slice[np.newaxis, ...]
        # Ensure all slices have matching shapes
        if wave_slice.shape[1:] != wind_u_slice.shape[1:]:
            raise ValueError(f"Shape mismatch: wave_slice={wave_slice.shape},
wind_u_slice={wind_u_slice.shape}")
        # Stack wave and wind components along the last axis
        X_seq = np.stack([wind_u_slice[0], wind_v_slice[0]], axis=-1)
        y_seq = wave[:, t:t + pred_length]  # Target wave heights
        X.append(X_seq)
        y.append(y_seq)
    X = np.array(X)  # Final shape: (samples, seq_length, lat, lon, features)
    y = np.array(y)  # Final shape: (samples, pred_length, lat, lon)
    return X, y
wind_file_1 = 'H:.\\era5_full_2016_2023.nc'
mask_file_path = '.\\mask.mat'  # Replace with the actual path to your .mat file
time_strt = 70121-8544
time_end = 70121
wind_u, wind_v, wave = load_and_interpolate_data(wind_file_1, time_strt,
time_end)
```

```
mask = loadmat(mask_file_path)
id_msk = mask['id_msk']
wave = apply_mask_to_wave_data(wave, id_msk)
wind_u = apply_mask_to_wave_data(wind_u, id_msk)
wind_v = apply_mask_to_wave_data(wind_v, id_msk)
swh_seq = wave
wind_data = np.stack([wind_u, wind_v], axis=-1)
X_new = create_sequences(wind_data, seq_length=3)
y_new = create_sequences(wave, seq_length=3)
import joblib
# Load the trained model
model = joblib.load('ConvLSTM_model2.h5');
#load_model("ConvLSTM_model_1.h5", custom_objects={'nan_mse_loss': nan_mse_loss,'nan_mae': nan_mae, 'NaNMaskingLayer': NaNMaskingLayer})
# Use it for predictions
y_pred = model.predict(X_new)
time_step = 1  # Adjust the time step as needed
y_val_grid = y_new[0, 0, :, :]
y_pred_grid = y_pred[0, 0, :, :, 0]
fig, axes = plt.subplots(1, 3, figsize=(12, 6))
im_pred = axes[0].imshow(y_pred_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[0].set_title("Predicted SWH")
axes[0].axis("off")
fig.colorbar(im_pred, ax=axes[0], label="SWH")
im_true = axes[1].imshow(y_val_grid, cmap="viridis", aspect="auto", vmin=0, vmax=4)
axes[1].set_title("True SWH")
axes[1].axis("off")
fig.colorbar(im_true, ax=axes[1], label="SWH")
```

```
im_diff = axes[2].imshow(y_val_grid - y_pred_grid, cmap="viridis",
aspect="auto", vmin=-0.5, vmax=0.5)

axes[2].set_title("Difference SWH")

axes[2].axis("off")

fig.colorbar(im_diff, ax=axes[2], label="True - Predicted")

plt.tight_layout()

#plt.savefig('swh_predictions.png')  # Save instead of show

plt.show()  # Close figure

file_name = f"matlab_out_pred_{time_strt}_{time_end}.mat"

savemat(file_name, {'X_new':X_new, 'y_new':y_new, 'time_strt':time_strt,
'time_end':time_end,

                'y_pred':y_pred})
```

ПРИЛОЖЕНИЕ Б

Код для расчета модели XGBoost

```
import scipy.io as sio
import numpy as np
import os
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error
import matplotlib.pyplot as plt
import joblib
import time
model_name = 'xgboost_model_swh_test_10m.pkl'
start = time.time()
# Define file paths
file_paths = [
    ('.\\', 'JASON1_data.mat'),
    ('.\\', 'JASON2_data.mat'),
    ('.\\', 'JASON3_data.mat'),
    ('.\\', 'TOPEX_data.mat'),
    ('.\\', 'GFO_data.mat'),
    ('.\\', 'ERS_data.mat'),
    ('.\\', 'ENVI_data.mat'),
    ('.\\', 'SENTINEL3A_data.mat'),
    ('.\\', 'SENTINEL6A_data.mat')
]
# Initialize lists to store combined data
ALT_lat, ALT_lon, ALT_time, ALT_wnd_cal, mod_wdr, ALT_swh_ku_cal = [], [], [], [], [], []
# Load and combine data from all files
```

```
for folder, file in file_paths:
    mat_data = sio.loadmat(os.path.join(folder, file))
    # Extract structured array (e.g., 'JASON1_data')
    key = next(k for k in mat_data.keys() if not k.startswith('__'))
    struct_data = mat_data[key]
    ALT_lat.extend(struct_data['ALT_lat'][0, 0].flatten())
    ALT_lon.extend(struct_data['ALT_lon'][0, 0].flatten())
    ALT_time.extend(struct_data['ALT_time'][0, 0].flatten())
    ALT_wnd_cal.extend(struct_data['ALT_wnd_cal'][0, 0].flatten())
    mod_wdr.extend(struct_data['mod_wdr'][0, 0].flatten())
    ALT_swh_ku_cal.extend(struct_data['ALT_swh_ku_cal'][0, 0].flatten())
# Convert lists to NumPy arrays
X = np.column_stack((ALT_lat, ALT_lon, ALT_time, ALT_wnd_cal, mod_wdr))
y = np.array(ALT_swh_ku_cal)
# Remove NaN or Inf values
valid_mask = np.isfinite(y) & np.all(np.isfinite(X), axis=1)
X, y = X[valid_mask], y[valid_mask]
# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
y_train = np.where(y_train == 0, 1e-6, y_train)  # Replace zeros with a small
number
y_test = np.where(y_test == 0, 1e-6, y_test)
dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)
params = {
    'objective': 'reg:squarederror', #'reg:pseudohubererror'
    'learning_rate': 0.01,      # Reduced for stability
    'max_depth': 25,            # Reduced to avoid overfitting
    'subsample': 1,           # Prevents overfitting
    'colsample_bytree': 1,    # Uses x% of features
```

```
    'alpha': 1,               # L1 regularization
    'eval_metric': ['rmse', 'mae', 'mape']
}
evals = [(dtrain, 'train'), (dtest, 'test')]
evals_result = {}
bst = xgb.train(params, dtrain, num_boost_round=1000, evals=evals,
evals_result=evals_result, early_stopping_rounds=100, verbose_eval=10)
predictions = bst.predict(dtest
rmse = np.sqrt(mean_squared_error(y_test, predictions))
mae = mean_absolute_error(y_test, predictions)
si = rmse / np.mean(y_test)  # Scatter Index
print(f'RMSE: {rmse}')
print(f'MAE: {mae}')
print(f'SI: {si:.4f}')
# Save the model
joblib.dump(bst, model_name)
endtime = time.time()
print('calculation time (minites)',(endtime - start)/60)
##xgb.plot_importance(bst, importance_type='weight', max_num_features=10)
##plt.title('Feature Importance')
##plt.show()
# Scatter plot of true vs predicted values
plt.figure(figsize=(8, 6))
plt.scatter(y_test, predictions, color='blue', alpha=0.5)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', lw=2)
plt.xlabel('True Values')
plt.ylabel('Predictions')
plt.title('True Values vs Predictions')
plt.show()
```

```
# Plot loss curve
plt.figure(figsize=(10, 6))
plt.plot(evals_result['train']['rmse'], label='Train RMSE')
plt.plot(evals_result['test']['rmse'], label='Test RMSE')
plt.xlabel('Boosting Round')
plt.ylabel('RMSE')
plt.title('XGBoost Training and Validation metrics')
plt.legend()
plt.show()
plt.figure(figsize=(10, 6))
plt.plot(evals_result['train']['mape'], label='Train mape')
plt.plot(evals_result['test']['mape'], label='Test mape')
plt.xlabel('Boosting Round')
plt.ylabel('MAPE')
plt.title('XGBoost Training and Validation metrics')
plt.legend()
plt.show()
plt.figure(figsize=(10, 6))
plt.plot(evals_result['train']['mae'], label='Train mae')
plt.plot(evals_result['test']['mae'], label='Test mae')
plt.xlabel('Boosting Round')
plt.ylabel('MAE')
plt.title('XGBoost Training and Validation metrics')
plt.legend()
plt.show()
# Save data for MATLAB
##sio.savemat('combined_altimeter_data.mat', {'X': X, 'y': y, 'X_train': X_train, 'X_test': X_test, 'y_train': y_train, 'y_test': y_test})
import numpy as np
import xgboost as xgb
```

```
import joblib
import xarray as xr
import matplotlib.pyplot as plt
import scipy.io as sio  # MATLAB file handling
from datetime import datetime, timedelta
# Load trained model
bst = joblib.load('xgboost_model_swh_test_1m.pkl')
# Load ERA5 data
wind_file_1 = '.\\era5_full_2016_2023.nc'
dataset = xr.open_dataset(wind_file_1)
# Define time range
tstart = 61577 #70115#61577
tend = 70120
era5_time_units = dataset['time']  # Usually "hours since 1970-01-01"
time_origin = datetime(1970, 1, 1)  # MATLAB's datenum('1970-01-01')
time_values = np.array([time_origin + timedelta(days=t) for t in
dataset['time'].values])  # Convert to datetime
# Get lat/lon grid
lon, lat = np.meshgrid(dataset['lon'], dataset['lat'])
[nx, my] = lon.shape
[ht] = era5_time_units[tstart:tend+1].shape
era5_swh_t = np.zeros([nx, my, ht])
XGb_swh = np.zeros([nx, my, ht])
diff_swh = np.zeros([nx, my, ht])
for t in range(tstart, tend + 1):  # Iterate over time steps
    print(f'Processing time step {t}')
    current_date = time_values[t].strftime("%Y-%m-%d %H:%M")  # Convert to
string for title
    print(f'Processing time step {t}, Date: {current_date}')
    # Extract data at time step
    era5_swh = dataset['swh'].isel(time=t)
```

```
    mod_wsp = dataset['U10'].isel(time=t)
    mod_wdr = dataset['phi_w'].isel(time=t)
    # Prepare input features for the model
    time_step_array = np.full(lon.shape, t)  # Add time as an input feature
    X_pred = np.column_stack((lat.ravel(), lon.ravel(), time_step_array.ravel(),
mod_wsp.values.ravel(), mod_wdr.values.ravel()))
    dpred = xgb.DMatrix(X_pred)
    predictions = bst.predict(dpred)
    # Reshape predictions back to grid
    y_pred_grid = predictions.reshape(era5_swh.shape)
    # Mask predictions where ERA5 SWH is NaN
    y_pred_grid = np.where(np.isnan(era5_swh), np.nan, y_pred_grid)
    # Compute absolute difference
    abs_diff = (y_pred_grid - era5_swh)/era5_swh*100
    # Mask absolute difference where ERA5 SWH is NaN
    abs_diff = np.where(np.isnan(era5_swh), np.nan, abs_diff)
    # Save results for this time step in MATLAB dictionary
    era5_swh_t[:,:,t-tstart] = era5_swh
    XGb_swh[:,:,t-tstart] = y_pred_grid
    diff_swh[:,:,t-tstart] = abs_diff
    # Create the figure and subplots
    fig, axs = plt.subplots(2, 2, figsize=(12, 10))
    fig.suptitle(f'Date: {current_date}', fontsize=14, fontweight='bold')
    # Wind Field (u, v) visualization
    ax = axs[0, 0]
    c=ax.pcolormesh(lon, lat, mod_wsp, cmap='viridis')
    fig.colorbar(c, ax=ax)
    ax.set_title(f'Wind Field')
    ax.set_xlabel('Longitude')
```

```
    ax.set_ylabel('Latitude')
    # ERA5 SWH
    ax = axs[0, 1]
    c = ax.pcolormesh(lon, lat, era5_swh, cmap='viridis')
    fig.colorbar(c, ax=ax)
    ax.set_title(f'ERA5 SWH')
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    # Predicted SWH
    ax = axs[1, 0]
    c = ax.pcolormesh(lon, lat, y_pred_grid, cmap='viridis')
    fig.colorbar(c, ax=ax)
    ax.set_title(f'Predicted SWH')
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    # Absolute Difference
    ax = axs[1, 1]
    c = ax.pcolormesh(lon, lat, abs_diff, cmap='coolwarm', vmin=-50, vmax=50)
    fig.colorbar(c, ax=ax)
    ax.set_title(f'Difference %')
    ax.set_xlabel('Longitude')
    ax.set_ylabel('Latitude')
    # Adjust layout and save figure
    plt.tight_layout()
    plt.savefig(f'prediction_t{t}.png')  # Save the plot for each time step
    plt.close()  # Close the figure to save memory
    # Create a dictionary to store results for MATLAB
OUT_data = {'lon': lon, 'lat': lat, 'time': era5_time_units[tstart:tend+1], 'era5_swh': era5_swh_t, 'XGb_swh': XGb_swh, 'diff_swh': diff_swh}
```

```
# Save results to MATLAB file
sio.savemat('xgboost_predictions_1m.mat', OUT_data)
dataset.close()
print('Processing complete. Results saved to xgboost_predictions.mat')
```