

Инкрементное машинное обучение мягких пневматических приводов с симметричными камерами

Ю.Н. КОЖУБАЕВ ✉, Е.Н. ОВЧИННИКОВА, В.Ю. ИВАНОВ, С.Ю. КРотова

Санкт-Петербургский горный университет, Санкт-Петербург, Россия

✉ kozhubaev_yun@pers.spmi.ru

Ссылка для цитирования оригинальной статьи: Kozhubaev, Y.; Ovchinnikova, E.; Viacheslav, I.; Krotova, S. Incremental Machine Learning for Soft Pneumatic Actuators with Symmetrical Chambers. Symmetry 2023, 15, 1206. <https://doi.org/10.3390/sym15061206>

Аннотация: Мягкая робототехника – это специализированная область робототехники, которая занимается проектированием, производством и управлением роботами из мягких материалов, в отличие от роботов из жестких звеньев. Одной из основных проблем будущего использования непрерывных или гиперредундированных робототехнических систем в промышленной и медицинской технике является разработка подходящих подходов к моделированию и управлению. Из-за сложного нелинейного поведения мягких материалов и непредсказуемости движения исполнительных механизмов задача моделирования сложных мягких исполнительных механизмов является очень трудоемкой. В связи с этим в более ранних работах проводились исследования безмодельных методов управления мягкими исполнительными механизмами. В последние годы широкое распространение в исследованиях получили методы машинного обучения (ML). Адаптивность модели ML к нелинейной системе мягкого привода, а также изменяющееся со временем поведение мягких приводов в результате изменения характеристик материала и эксплуатационных требований являются ключевым основанием для включения модели ML. Для работы с нелинейной системой требуется оперативное обновление ML-модели. Последовательные данные, собранные на испытательном стенде и преобразованные в гипотезу, используются для осуществления инкрементного обучения. Эти методы называются пожизненным обучением и прогрессивным обучением. Обучение потоку данных в реальном времени сочетается с инкрементным обучением (IL) и нейросетевая модель настраивается последовательно для каждого входного сигнала. В данной статье предлагается метод интеллектуального управления мягкими пневматическими приводами на основе алгоритма инкрементного обучения. Мягкий пневматический привод подвергался трем различным условиям испытаний в контролируемой тестовой среде в течение определенного времени сбора данных. Кроме того, сбор данных осуществлялся с помощью моделирования методом конечных элементов. Собранные данные использовались для инкрементного обучения нейронной сети, а полученная модель анализировалась на предмет ошибок как при обучении, так и при тестировании. Сравнение ошибок обучения и тестирования проводилось для различных алгоритмов инкрементного обучения (IL), включая K-ближайших соседей, дерево решений, линейную регрессию и нейронную сеть. Была проверена возможность безмодельного интеллектуального управления мягкими пневматическими приводами на основе алгоритма инкрементного обучения, что позволило решить проблему управления программными исполнительными механизмами.

Ключевые слова: мягкая робототехника; машинное обучение; многоцелевая регрессия; инкрементное обучение; нейронная сеть; пневматический привод.

Incremental Machine Learning for Soft Pneumatic Actuators with Symmetrical Chambers

Yuriy N. KOZHUBAEV ✉, Elena N. OVCHINNIKOVA, Viacheslav Yu. IVANOV and Svetlana Yu. KROTOVA

Saint Petersburg Mining University, St. Petersburg, Russia

✉ kozhubaev_yun@pers.spmi.ru

Abstract: Soft robotics is a specialized field of robotics that focuses on the design, manufacture, and control of robots made of soft materials, as opposed to those made of rigid links. One of the primary challenges for the future use of continuous or hyper-redundant robotics systems in industrial and medical technology is the development of suitable modeling and control approaches. Due to the complex non-linear behavior of soft materials and the unpredictable motion of actuators, the task of modeling complex soft actuators is very time-consuming. As a result, earlier studies have undertaken research into model-free methods for controlling soft actuators. In recent years, machine learning (ML) methods have become widely popular in research. The adaptability of an ML model to a non-linear soft drive system alongside the varying actuation behavior of soft drives over time as a result of material characteristics and performance

* Перевод подготовлен для повышения интереса русскоязычной аудитории к данной статье.

© 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

© Санкт-Петербургский горный университет императрицы Екатерины II (перевод), 2024.

requirements is the key rationale for including an ML model. The system requires the online updating of the ML model in order to work with the non-linear system. Sequential data collected from the test bench and converted into a hypothesis are used to perform incremental learning. These methods are called lifelong learning and progressive learning. Real-time data flow training is combined with incremental learning (IL), and a neural network model is tuned sequentially for each data input. In this article, a method for the intelligent control of soft pneumatic actuators based on an incremental learning algorithm is proposed. A soft pneumatic actuator was subjected to three distinct test conditions in a controlled test environment for a specified duration of data gathering. Additionally, data were collected through finite element method simulations. The collected data were used to incrementally train a neural network, and the resulting model was analyzed for errors with both training and test data. The training and testing errors were compared for different incremental learning (IL) algorithms, including K-nearest neighbors, a decision tree, linear regression, and a neural network. The feasibility of the modulo-free intelligent control of soft pneumatic actuators based on an incremental learning algorithm was verified, solving the problem of the control of software actuators.

Keywords: soft robotics; machine learning; multi-objective regression; incremental learning; neural network; pneumatic actuator

1. Введение

Робототехника широко используется в промышленном производстве, геологоразведке, медицинском обслуживании, военной разведке и других областях, что имеет большое значение для национальной экономики и оборонного строительства страны. С точки зрения используемых основных материалов современные роботы можно разделить на два типа: роботы с жестким корпусом и роботы с мягким корпусом. Традиционные роботы с жестким корпусом состоят в основном из жестких кинематических пар на основе твердых материалов (таких как металлы, пластмассы и т.д.), которые позволяют быстро и точно выполнять рабочие задачи. Наиболее широко используемые в промышленном производстве роботы с жестким корпусом являются нередуцируемыми по кинематике [1]. Множество кинематических сочленений такого робота жестко связаны между собой, и каждое сочленение обеспечивает роботу одну степень свободы вращательного или линейного движения. Достижимый диапазон всех степеней свободы составляет рабочее пространство робота, а также определяет все положения, в которые может попасть захватное устройство. Однако традиционный робот обладает ограниченной гибкостью и низкой приспособляемостью к окружающей среде, а также может работать только в структурированной среде, что ограничивает применение жестких роботов в динамичных, неизвестных и неструктурированных сложных средах, таких как военная разведка, ликвидация последствий стихийных бедствий, научная разведка и т.д. При дальнейшем увеличении числа сочленений жесткого робота появляется избыточность или даже сверхизбыточность, что значительно повышает его ловкость. Адаптивность к окружающей среде у такого робота повышается по сравнению с роботом без избыточности, однако его корпус по-прежнему состоит из жестких материалов, а размеры и габариты не могут быть изменены произвольно. При его применении в конкретной среде требуется предварительная информация, например, о форме и размерах препятствий или линейного перемещения.

По сравнению с роботами с жестким телом роботы с мягким телом имеют больше степеней свободы и могут реализовать больше режимов движения, что дает роботам с мягким телом определенные преимущества с точки зрения гибкости, но при этом возникают новые проблемы, такие как управление восприятием положения роботами с мягким телом, улучшение управляемости роботов с мягким телом, повышение несущей способности мягких конструкций и т.д. За последние 10-20 лет исследователи по всему миру разработали целый ряд мягких роботов, в которых реализованы некоторые функции, труднодостижимые для традиционных роботов, такие как ползание по щели, непрерывное плавание, гибкое хватание и т.д. Обычно источником вдохновения для создания мягких роботов служат организмы или биологические ткани, встречающиеся в природе: гусеницы, морские звезды, щупальца осьминогов и т.д. Это связано с тем, что моллюски или мягкие ткани животных прошли естественную эволюцию в течение сотен миллионов лет и обладают такими характеристиками, как большие деформации, малый вес и высокая плотность мощности, что позволяет повысить их эффективность в сложных природных условиях за счет изменения формы тела. Изучая особенности движения, строение эпидермиса и способы сокращения мышц моллюсков, исследователи использовали такие методы, как физика, химия и свойства материалов, для реализации биологических движений [2].

В настоящее время робототехника и автоматизация широко интегрированы во все технологические процессы, включая управление движением транспортных средств [3], автоматическую локализацию и картографирование внутри и вне помещений [4, 5], системы компьютерного зрения [6-8], машинное обучение [9, 10] и нейронные сети [11, 12], и, безусловно, повышают эффективность технологических процессов.

Мягкая робототехника – это область, занимающаяся созданием и управлением упругими роботами, в основе которой лежат идеи мягкотелых животных. Конструкции мягких роботов базируются на анатомии мягкотелых организмов, таких как осьминоги, хоботы слонов, змеи и т.д., что позволяет использовать влияние природы и создавать очень сложные и совершенные экологические конструкции [13]. Многочисленные конструкции изучаются и разрабатываются при проектировании и создании мягких роботов (МР), способных решать любые задачи, возникающие в повседневной жизни. Благодаря своей мягкости такие роботы могут использоваться в опасных условиях [14].

Область мягкой робототехники развивалась с помощью многочисленных стратегий проектирования и управления и в настоящее время борется с проблемами управления, вызванными вязкоупругой природой этих роботов [15]. Благодаря своей хрупкости и простоте изготовления силиконовые формы являются основой архитектуры МР.

В последние годы МР нашли применение и в области медицины [16,17], включая хирургию и имплантацию протезов инвалидам. Два основных направления исследований МР – проектирование и управление, причем проектирование оказывается более сложным, чем в случае традиционных роботов из жестких материалов. Для многих образовательных и промышленных учреждений привлекательной является инновационная система управления, расширяющая границы возможностей управления мягкими роботами [18-20]. Основным компонентом, который должен присутствовать в МР, является исполнительная система, которая помогает производить выходной сигнал, необходимый для взаимодействия с окружающей средой. Существует множество методов, позволяющих продемонстрировать эффективность МР на основе его вязкоупругого поведения.

Учитывая, что электрические приводы обычно дороже и тяжелее пневматических и тепловых, они также имеют меньшую удельную мощность. Кроме того, электрические и тепловые приводы могут нагреваться в процессе работы и требуют охлаждения или отвода тепла, что усложняет систему, в то время как пневматические приводы обычно проще, дешевле и легче электродвигателей. Кроме того, они могут обеспечивать более высокую удельную мощность и скорость по сравнению с электродвигателями. Кроме того, пневматический привод может использоваться для работы в опасных или неблагоприятных условиях, таких как высокие температуры, радиация, химически агрессивные среды и т.д. В данном эксперименте был выбран пневматический привод.

Нелинейное и гистерезисное поведение мягких электроприводов кратко рассмотрено в литературе [21]. Под нелинейностью понимается связь между входными и выходными характеристиками системы. Гистерезис, смещение и дополнительные степени свободы вносят свой вклад в нелинейность системы, усложняя управление приводом. Методы машинного обучения доказали свою эффективность в решении этих проблем, позволяя находить решения для сложных систем в различных областях. Были предложены различные методы управления на основе обучения, объяснены проблемы, связанные с временным поведением [22], а также конкретные модели машинного обучения, используемые в современных методах управления. Было установлено, что по сравнению с методами, основанными на якобиане, для мягких приводов нейронная сеть с прямолинейным движением (FNN) превосходит по точности системы. Однако сложность адаптации систем к нелинейности остается, поскольку статическое машинное обучение не подходит для систем с изменяющимся поведением. Поэтому в области мягкой робототехники все большее распространение получают подходы, основанные на обучении, такие как обучение с подкреплением и онлайн-инкрементное обучение. В данной статье мы попытались использовать онлайн-инкрементное обучение для решения проблем управления мягкими приводами, сравнить ошибки обучения и тестирования алгоритмов K-ближайшие соседи, дерева решений, линейной регрессии и нейронной сети, а также сравнить полученные результаты с помощью сравнительного анализа. Проведенные эксперименты подтвердили возможность достижения заданного эффекта безмодельного управления.

2. Материалы и методы

В этой главе в основном рассматривается проектирование модели системы, метод обучения модели и метод тестирования модели. В ней также рассматриваются целесообразность, преимущества и недостатки четырех методов проектирования модели. Различия между четырьмя методами обучения модели показывают структуру и рациональность тестирования системы. В целом в этой главе мы сначала анализируем причины и цели выбора регрессора К- ближайшие соседи, регрессора дерева решений, линейной регрессии и нейронной сети, затем теоретически анализируем преимущества и недостатки пакетного и инкрементного обучения и выбираем наиболее подходящий для данного эксперимента метод обучения с использованием инкрементного обучения, а затем проводим тесты с предварительным и непредварительным обучением с использованием метода инкрементного обучения. В итоге мы получаем ошибки обучения четырех моделей в группе обучения и ошибки тестирования в группе тестирования, которые включены в следующий анализ результатов, представляющий собой основу данных.

2.1. Алгоритмы, связанные с моделью

В данной статье для адаптивного обучения необходимо было использовать метод инкрементного обучения. Среди алгоритмов, подходящих для инкрементного обучения, алгоритм К- ближайшие соседи является достаточно быстрым для обработки новых данных. Он не требует явного процесса обучения и может выполнять классификацию, регрессию, кластеризацию и т.д. Кроме того, алгоритм К-ближайшие соседи может адаптироваться к различным распределениям данных и обладает хорошей устойчивостью к нелинейным данным. Алгоритм «дерево решений» может работать с наборами данных с множеством категорий и множеством признаков, а также обладает хорошей интерпретируемостью и наглядностью. Деревья решений также способны работать с недостающими и зашумленными данными и быстро обрабатывать большие массивы данных. Линейная регрессия позволяет предсказывать непрерывные значения, решать проблемы множественной регрессии, а также обладает хорошей устойчивостью для высокоразмерных наборов данных и объясняет взаимосвязи между каждым признаком в этих наборах данных и целевыми переменными. Нейросетевой алгоритм может работать со сложными нелинейными моделями, обладает лучшей производительностью для высокоразмерных наборов данных и может выполнять сквозное обучение. Кроме того, нейросетевой алгоритм обладает хорошей способностью к обобщению и может работать с отсутствующими и зашумленными данными. Все четыре рассмотренных выше метода соответствуют нелинейности, прерывистости данных и шумовым характеристикам экспериментальной модели.

2.1.1. Регрессор «К-ближайшие соседи»

Парадигма контролируемого обучения «К-ближайшие соседи» (KNN) [23,24] является непараметрической, и выходом регрессора KNN является значение свойства объекта, которое представляет собой среднее значение KNN. В основе метода KNN лежит расстояние, вычисляемое от наблюдаемых «k» точек до заданной точки в наборе данных. Количество окружающих точек, которые модель будет учитывать при оценке новой точки, обозначается в алгоритме буквой «k». Регрессор KNN использует расстояние между точками для определения выходного значения. Существует множество подходов, которые часто используются, в том числе метод манхэттенского расстояния и метод евклидова расстояния. Выбор метода почти наверняка будет зависеть от конкретного случая использования, и каждый метод будет влиять на данные более высокой размерности. Благодаря своей эффективности расстояние Минковского является наиболее популярной метрикой расстояний. Уравнение показано ниже.

$$d(X, y) = \sum_{i=0}^n (|x_i - y_i|^p)^{1/p} \quad (1)$$

В зависимости от значения p приведенная формула представляет собой как манхэттенское, так и евклидово расстояние: если p равно 1, то это манхэттенское расстояние, а если p равно 2, то евклидово расстояние. На основании результатов голосования каждому прогнозу присваивается вес. Близкие точки будут иметь больший вес, чем удаленные. Алгоритм по-прежнему учитывает каждого из k ближайших соседей, но более близким соседям он приписывает больший вес. Аналогичным образом алгоритм решает проблему MTR.

2.1.2. Регрессор дерева решений

Первичные входные выборки разбиваются на соответствующие однородные значения различных признаков с помощью регрессора в виде дерева решений (DT). Сегментируя входные признаки на небольшие подмножества с сохранением связности, DT строит модель в виде ветвящегося дерева, оставляя только узлы принятия решений и конечные узлы. Данные делятся по признаку, дающему наибольший прирост информации, начиная с корневого узла. Для решения проблемы перегрузки модели шаг разбиения повторяется в каждом дочернем узле до тех пор, пока из DT не будут исключены листовые узлы.

Для оптимизации алгоритма обучающего дерева должна существовать функция стоимости, позволяющая разделять узлы по наиболее информативным признакам. Как видно из приведенного ниже уравнения, цель состоит в том, чтобы максимизировать информацию, полученную от каждого разбиения

$$IG(D_p, f) = I(D_p) - \left(\frac{N_{left}}{N_p} I(D_{left}) + \frac{N_{right}}{N_p} I(D_{right}) \right) \quad (2)$$

В приведенном уравнении f обозначает функцию для выполнения разбиения, а D_p , D_{left} и D_{right} – функции родительского и дочернего узлов соответственно. $I(D_p)$, $I(D_{left})$ и $I(D_{right})$ – меры примесей, измеряющие наилучшее разделение между родительским узлом и последующим узлом. N слева и N справа означают количество выборок в дочернем узле, а N_p – общее количество выборок в разделяемом узле (родительском узле). DT, используемый в данном эксперименте, работает в соответствии с той же теорией, но с дополнительными целевыми предсказаниями.

2.1.3. Линейная регрессия

Для решения базовых задач наиболее популярным и эффективным подходом является линейная регрессия. Статистической парадигмой для определения взаимосвязи между характеристиками с независимыми входами и целями с зависимыми выходами является регрессионный анализ. Приведенное ниже уравнение представляет собой более широкий регрессионный анализ при моделировании n точек данных с одной независимой переменной.

$$y = \beta_0 + \beta_1 x_i + \epsilon_i, \quad i = 1, 2, \dots, n \quad (3)$$

Пересечение линий представлено через ϵ_i , где β – параметр, а x_i – независимая переменная. Экспериментальная модель использует ту же теорию, но имеет много выходных целей, а ее функция переводит эти многочисленные цели во входные атрибуты.

2.1.4. Нейронная сеть

Нейронная сеть – это взаимосвязанная группа нейронов, которые действуют как единицы хранения информации, решая сильно нелинейные задачи с меньшей сложностью [24]. Архитектура нейросетевой модели адаптирована из предыдущего исследования мягкого пневматического привода [18]. Сеть состоит из 2 скрытых слоев с 45 и 50 нейронами и по 3 нейрона на входе и выходе. Активацией для входного и скрытого слоев является функция активации гиперболический тангенс, а в выходном слое – линейная функция активации. На рис. 1 ниже показан подсчет параметров слоевой структуры нейронной сети.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_3 (Dense)	(None, 1, 45)	180
dense_4 (Dense)	(None, 1, 50)	2300
dense_5 (Dense)	(None, 1, 3)	153
Total params: 2,633		
Trainable params: 2,633		
Non-trainable params: 0		

Рис. 1. Архитектура нейросетевой модели

2.2. Выбор типа обучения модели

Обучение нейронной сети играет значительную роль в эффективности модели машинного обучения, поэтому обучение с использованием достаточного количества данных и подходящих методов приводит к получению непревзойденного результата. Основную роль в обучении модели играет градиентный спуск, при этом обновление весов происходит в соответствии с ошибкой между фактическими и предсказанными нейронной сетью значениями. Существует множество методов обучения обычной нейронной сети, из которых ниже рассматриваются методы пакетного и инкрементного обучения.

По сравнению с пакетным обучением инкрементное обучение позволяет в первую очередь экономить вычислительные ресурсы. Инкрементное обучение требует обновления модели только при поступлении новых данных, без переобучения всей модели. Это позволяет экономить вычислительные ресурсы. Данные в данном эксперименте существуют не постоянно, что соответствует методу инкрементного обучения, который позволяет непосредственно обновлять модель на основе существующей модели, не дожидаясь поступления всех данных перед выполнением пакетного обучения. Соответствующая скорость объемного обучения выше, а пакетное обучение подходит для статических наборов данных. Как правило, ошибка обучения чрезвычайно мала, но обобщающая способность слаба. Инкрементное обучение является противоположностью. Его можно постоянно обновлять и обучать. Кроме того, обобщающая способность выше, что подходит для динамических данных. Таким образом, в данном эксперименте метод инкрементного обучения имеет больше преимуществ, чем метод пакетного обучения.

2.3. Методы обучения

На производительность модели машинного обучения существенно влияет процесс обучения нейронной сети, при этом обучение на достаточном количестве данных и с использованием правильных методов позволяет получить непревзойденные результаты [25-27]. Обучение модели происходит преимущественно методом градиентного спуска, а обновление весов происходит на основе разницы между фактическим значением и значением, предсказанным нейронной сетью. Существует несколько методов обучения традиционных нейронных сетей, среди которых подробно рассматриваются методы пакетного и инкрементного обучения.

2.3.1. Пакетное обучение

Ошибка для каждой выборки в обучающих примерах вычисляется с помощью метода градиентного спуска, известного как пакетное обучение, однако вес обновляется только в конце каждой обучающей партии (эпохи). Стохастический градиентный спуск (СГС) [28, 29] – это вариант градиентного спуска, используемый в Python-библиотеке River Online Learning. Алгоритм пакетного градиентного спуска позволяет минимизировать эмпирический риск. Если скорость обучения γ_t положительна, а L – размер партии, то получается непрерывная оценка w_t оптимального параметра. Если событие z_t представляет собой событие, указывающее на наборы данных в партии, а событие w_t – оптимизируемый вес, то $Q(z_t, w_t)$ представляет собой функцию потерь.

$$w_{t+1} = w_t - \gamma_t \frac{1}{L} \sum_{i=1}^L \nabla_w Q(z_i, w_t) \quad (4)$$

Данный подход сходится к локальному минимуму эмпирического риска при малой скорости обучения γ_t . Скорость обучения γ_t может быть изменена с помощью соответствующей положительной матрицы для увеличения скорости сходимости. На этапе обучения происходит лишь несколько обновлений весов, что создает более стабильный градиент ошибки. Стабильный градиент ошибки также может привести к более высокой скорости сходимости и большей обобщенности набора данных. Если данные доступны на самых ранних этапах разработки модели, часто используется подход пакетного обучения.

2.3.2. Дополнительное обучение

Для вычисления ошибки каждой выборки в обучающих примерах и изменения веса модели используется пошаговое обучение, представляющее собой разновидность подхода онлайн градиентного спуска. Поскольку преобразование данных и обучение модели при пошаговом обучении выполняются постепенно, алгоритмы необходимо перестраивать с учетом изменений, как это было описано во введении. С помощью бегущей статистики, где среднее и стандартное отклонение инкрементно обновляются, оцениваются масштабирующие признаки для потоковых данных. x_i , $i = 1, m$. При m наблюдениях, x_i , $i = m + 1, m + n$, а также при поступлении нового набора наблюдений. С учетом m - среднего эмпирического значения в данный момент времени и n – среднего эмпирического значения последнего всплеска данных, ниже приводится формула рекурсивного обновления среднего значения:

$$\mu = \frac{m}{m+n}\mu_m + \frac{n}{m+n}\mu_n \quad (5)$$

Эмпирическая дисперсия задается через

$$\sigma^2 = \frac{m}{m+n}\sigma_m^2 + \frac{n}{m+n}\sigma_n^2 + \frac{mn}{m+n}(\mu_m - \mu_n)^2 \quad (6)$$

Реализация проста за счет менее сложного обновления весов, а зашумленные данные и смещения также стремятся создать плохую модель, избегая глобальных минимумов. Функция стоимости не всегда является выпуклой, как показано в примере на рис. 2 (несколько минимумов), и стохастический градиентный спуск обрывается, чтобы пропустить глобальные минимумы, а шаги, предпринимаемые для достижения минимумов, также являются шумными из-за зашумленных данных. На каждом шаге вычисленный градиент не является реальным, а представляет собой лишь его приближение, поэтому в функции стоимости возникают значительные флуктуации, что приводит к снижению эффективности модели [10].

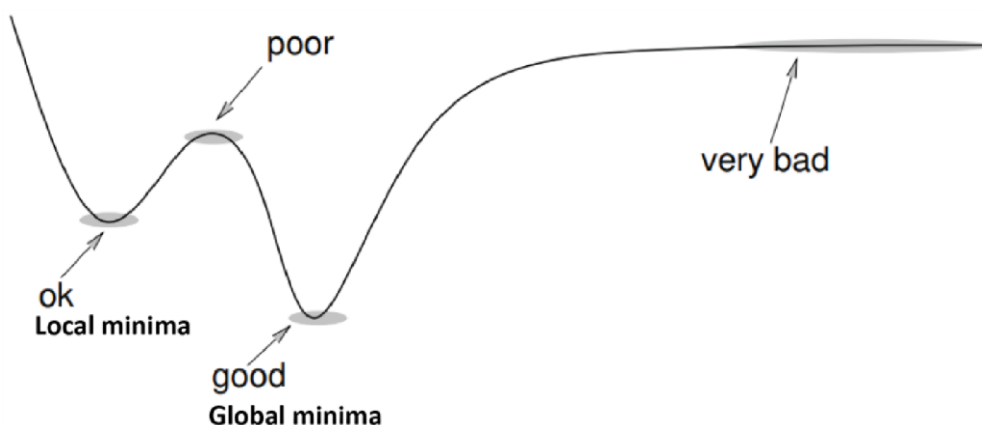


Рис. 2. Иллюстрация функции затрат

2.3.3. Метод обучения и тестирования

В данном эксперименте использовалось инкрементное обучение, при котором весовые коэффициенты обновляются для каждой выборки обучающих данных. Обучение является важным компонентом эффективной модели. Обученная модель оценивалась по средней абсолютной ошибке в каждом направлении движения конечного устройства (x, y, z). Поскольку она обеспечивает точный показатель ошибки, связанный с реальным выходом, в данной статье использовалась матрица средней абсолютной ошибки. Как показано на рис. 3, в процессе обучения использовалось множество сценариев. Целью данного эксперимента было создание модели, способной предсказывать местоположение конечного исполнительного механизма по будущим данным путем инкрементного обучения и сохранения модели, которая затем может быть использована для планирования курса мягкого пневматического исполнительного механизма. Обучение было разделено на две категории: предтренировочное и не предтренировочное.

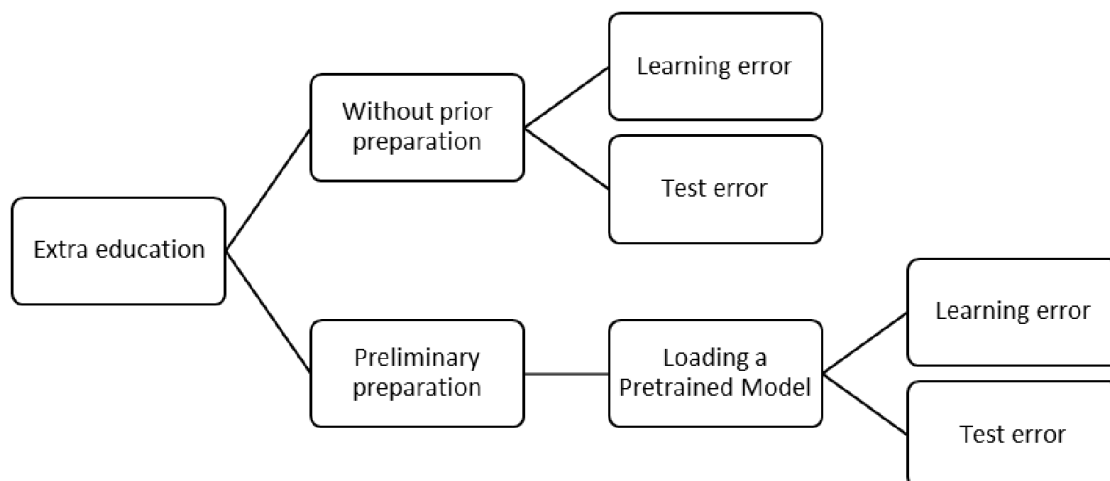


Рис. 3. Методология обучения. Модель обучается на исходных данных, прогнозы делаются на потоке данных, а для отслеживания отклонений от истинного поведения генерируются ошибки обучения. Обученная модель записывается в pickle-файл и используется для прогнозирования вновь обнаруженных данных. Затем рассчитываются и сравниваются ошибки обучения в реальном времени и ошибки тестирования для необнаруженных данных. Таким образом, оценивается способность модели удерживать информацию. Метод проиллюстрирован блок-схемой на рис. 4

2.4. С предтренировочным обучением

При сравнении обучения с предыдущей частью были замечены незначительные различия. С помощью трансферного обучения в данном эксперименте оценивался прогресс в реальном поведении модели. Сначала было проведено пакетное обучение модели на симулированных данных МКЭ, после чего были записаны веса в файл формата pickle. Затем сохраненная модель использовалась для дополнительного обучения после вставки в объект моделирования. Проводилось сравнение и изучение ошибок, при этом ошибка реального времени, ошибка обучения и ошибка тестирования генерировались с использованием латентных данных.

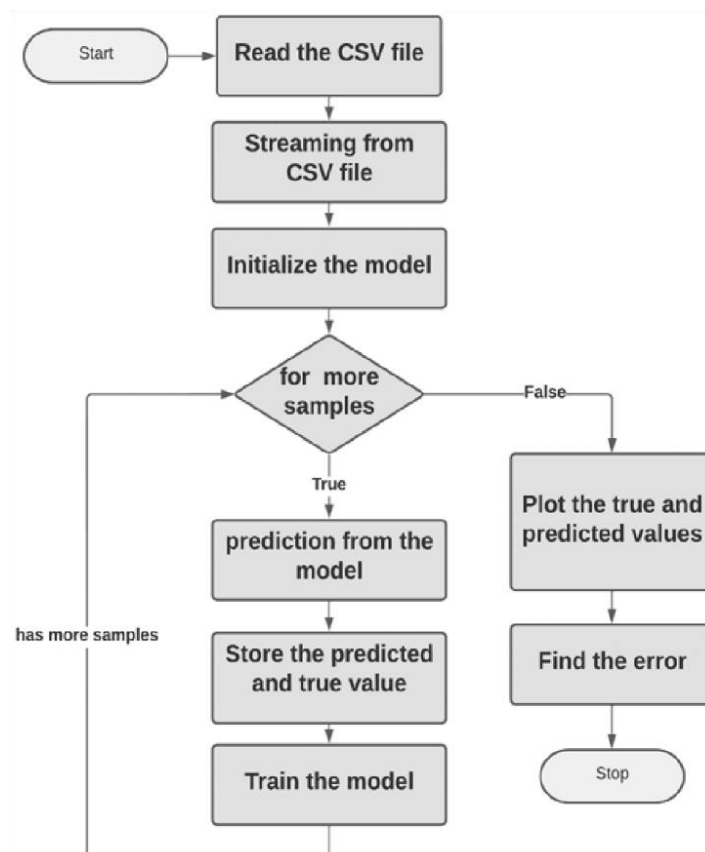


Рис. 4. Блок-схема алгоритма

2.5. Эргономические методологии моделирования и управления процессом изготовления мягких роботов

- Проектирование для сборки (DFA): DFA – это методология, позволяющая спроектировать компоненты таким образом, чтобы их можно было легко и быстро собрать. Этот метод учитывает весь производственный процесс и направлен на сокращение количества этапов, компонентов и трудозатрат, необходимых для производства конечного продукта.
- Автоматизированное проектирование (CAD): CAD используется для создания 3D-моделей мягких роботов, позволяя инженерам визуализировать и манипулировать своими работами до начала производства. Использование этого инструмента позволяет производителям быстро выявлять потенциальные проблемы и вносить необходимые изменения в конструкцию.
- Моделирование процессов: моделирование технологических процессов – это метод, позволяющий имитировать поведение производственного процесса. Этот метод позволяет инженерам предсказать, как различные параметры повлияют на конечный продукт, и внести необходимые коррективы.
- Управление на основе имитационного моделирования: управление на основе моделирования предполагает использование компьютерного моделирования для тестирования и оптимизации систем управления мягкими роботами в процессе производства. Использование этого метода позволяет инженерам убедиться в том, что роботы способны выполнять поставленные задачи с минимальными ошибками и сбоями.

2.6. Реализация на языке Python

Язык Python используется для реализации машинного обучения, поскольку он имеет доступ к большой коллекции библиотек, используемых в науке о данных и машинном обучении [30, 31]. Благодаря тому, что модель может адаптироваться к изменениям в реальном мире, инкрементное обучение недавно заменило парадигму пакетного обучения. Идея инкрементного обучения требует создания библиотеки с нуля. В связи с этим желанием были разработаны библиотеки *steme* и *scikit-multiflow*, ориентированные на инкрементное обучение.

Creme и *scikit-multiflow* объединились для создания новой библиотеки, известной как *River*. Она предоставляет возможность потоковой обработки данных из файлов и поддерживает практически все методы машинного обучения и глубокого обучения. Кроме того, новая библиотека предлагает классы по обнаружению смещений, выявлению аномалий и многое другое. Библиотеку можно использовать в любой виртуальной среде на базе Python, а также в IDE. Для обучения и построения графиков данных кодирование выполняется в блокноте Google Colab, поскольку без GPU это заняло бы слишком много времени. Блок-схема программы приведена на рисунке ниже. Для инкрементного программирования на основе глубокого обучения используется библиотека Google Keras, которая отличается удобством использования и менее сложной структурой кода по сравнению с другими библиотеками, такими как TensorFlow и PyTorch [32, 33].

3. Экспериментальные установки

3.1. Привод

Изометрический вид и вид в поперечном сечении мягкого пневматического привода показаны на рис. 5. Привод, использовавшийся в эксперименте, представлял собой цилиндрическую конструкцию с тремя камерами давления, равномерно расположенными по окружности и перпендикулярными осевому направлению. Привод был изготовлен из различных силиконовых компонентов с использованием 3D-печатных пластиковых форм.

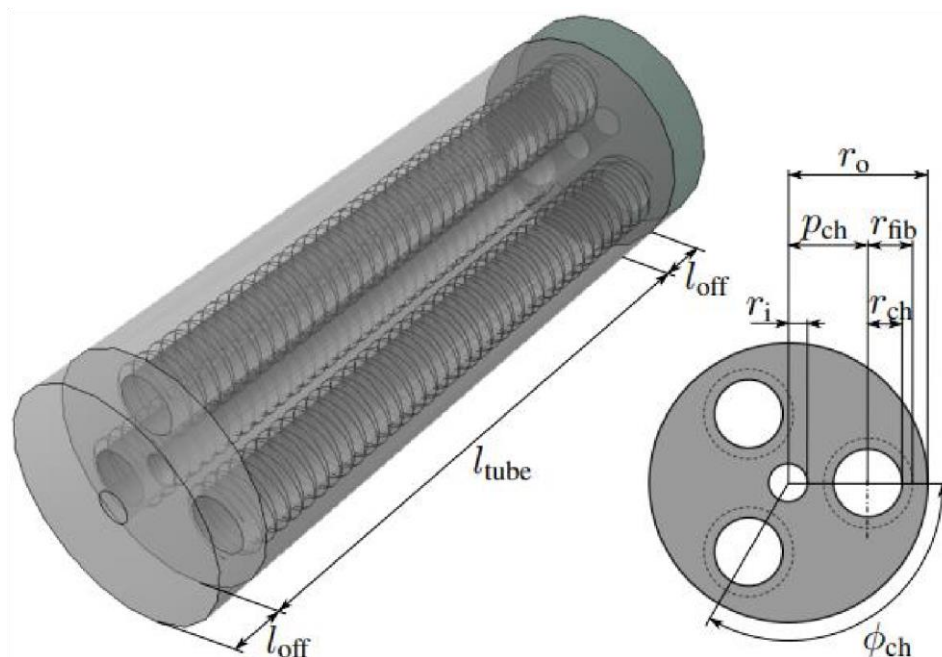


Рис. 5. Изометрический вид и вид в поперечном сечении мягкого пневматического привода

Камера привода изготовлена из Ecoflex-50 – более эластичного материала, способного выдерживать деформации изгиба. Для усиления камер использовалась переплетенная кевларовая проволока, уменьшающая радиальные деформации. Для формовки крышек торцов приводов использовался менее пластичный материал Dragon Skin 30. Один конец привода был закреплен, а на рабочем конце были установлены маркеры.

Размеры привода приведены ниже в табл. 1.

Таблица 1

Параметры мягкого пневматического привода

Параметр	l_{tube}	l_{off}	r_o	r_i
Значение	110 мм	10 мм	21,1 мм	2,9 мм
Параметр	r_{ch}	p_{ch}	r_{fib}	ϕ_{ch}
Значение	5,15 мм	12 мм	6,7 мм	1200

3.2. Испытательный стенд

На рис. 6 показан испытательный стенд. Мягкий пневматический стенд, вращающийся вокруг металлической рамы, обеспечивал все необходимые датчики входного давления и управления. Датчики, два компьютера и оптическая система слежения обеспечивали низкоуровневое управление. Поскольку калибровка проводилась для работы в критическом диапазоне, привод работал при максимальном давлении 1 бар. Вокруг испытательного стенда была установлена камера, а для отслеживания положения конечного исполнительного механизма использовалась оптическая система OptiTrack [19]. Система работала под управлением операционной системы Windows и преобразовывала визуальные данные в декартовы координаты, после чего передавала их в режиме реального времени на компьютер для дополнительных расчетов. Еще два компьютера были доступны для разработки системы и работы в режиме реального времени. Компьютер для работы в реальном времени также работал на той же архитектуре, что и компьютер для разработки, и использовал Secure Shell или Secure Socket Shell для мониторинга и управления системой. Компьютер для разработки работал под управлением операционной системы Linux и содержал MATLAB и Simulink, в которых находилась модель системы. Для управления PID-регулятором входное давление подавалось на вход через клапан, подключенный к датчику давления.

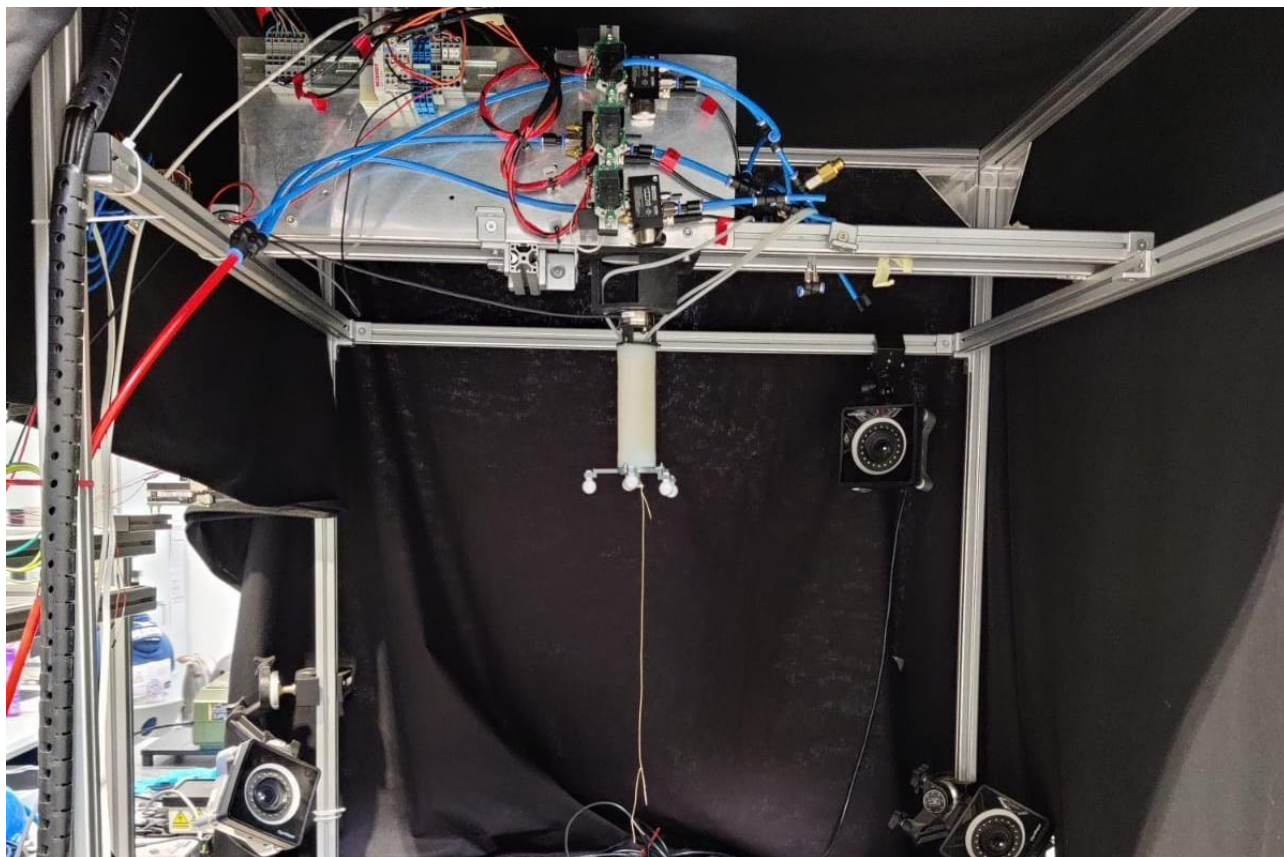


Рис. 6. Испытательный стенд с мягким пневматическим приводом

Данные с датчиков поступали по шине EtherCAT, которая, в свою очередь, передавала их на ПК реального времени по кабелю Ethernet, обеспечивая процедуру управления приводом. Система была создана с использованием сред MATLAB и Simulink на сложном ПК, а затем реализована на ПК реального времени.

3.3. Экспериментальные данные

Эксперимент проводился в течение определенного времени и при определенных условиях. В ходе эксперимента данные получались с помощью ПК реального времени, который, в свою очередь, рассчитывал необходимое заданное давление и собирал данные о положении

4. Результаты и обсуждение

Эксперименты проводились с использованием испытательного стенда для сбора данных о спиральных и круговых движениях конечного исполнительного механизма. Для эффективного обучения модели из набора данных были исключены исходные ложные сигналы о движении конечного захватного устройства. Основной целью эксперимента было предсказать эффективность инкрементно-обученной модели для будущего планирования траектории движения.

Эксперимент проводился в двух вариантах: сначала модель обучалась «с нуля», вычислялись ошибки реального времени или ошибки обучения, а также сохранялись выученные веса и прогнозировался выход в режиме предсказания с последующим вычислением ошибок. Во-вторых, использовался метод трансферного обучения. Модель обучалась на симулированных данных, затем проводилось дальнейшее инкрементное обучение с использованием данных и вычисленных ошибок реального времени или ошибок обучения. Обученные веса, полученные в результате обучения, сохранялись и загружались в объект модели для прогнозирования выхода, после чего производились расчеты ошибок. В табл. 2 приведены средние абсолютные ошибки различных алгоритмов ОД. Данные экспортировались в формат значений, разделенных запятыми (CSV), из матричного файла рабочей среды MATLAB. На рис. 7 показаны положения конечного захватного устройства SPA при спиральном движении с увеличением давления во всех трех камерах.

Входное давление рассчитывалось в соответствии с требуемыми положениями концевых захватных устройств. Для спирального движения давление на концевое захватное устройство увеличивалось со временем, а для кругового движения заданное давление изменялось синусоидально во всех трех камерах, которые были смещены друг относительно друга на 120 градусов. Данные состояли из трех входных признаков в виде трех значений давления и меток в виде выходных положений конечного захватного устройства. В зависимости от продолжительности эксперимента были собраны два типа данных о круговом движении. На рис. 8 показаны положения конечных захватных устройств SPA при круговом движении с изменением давления во всех трех камерах.

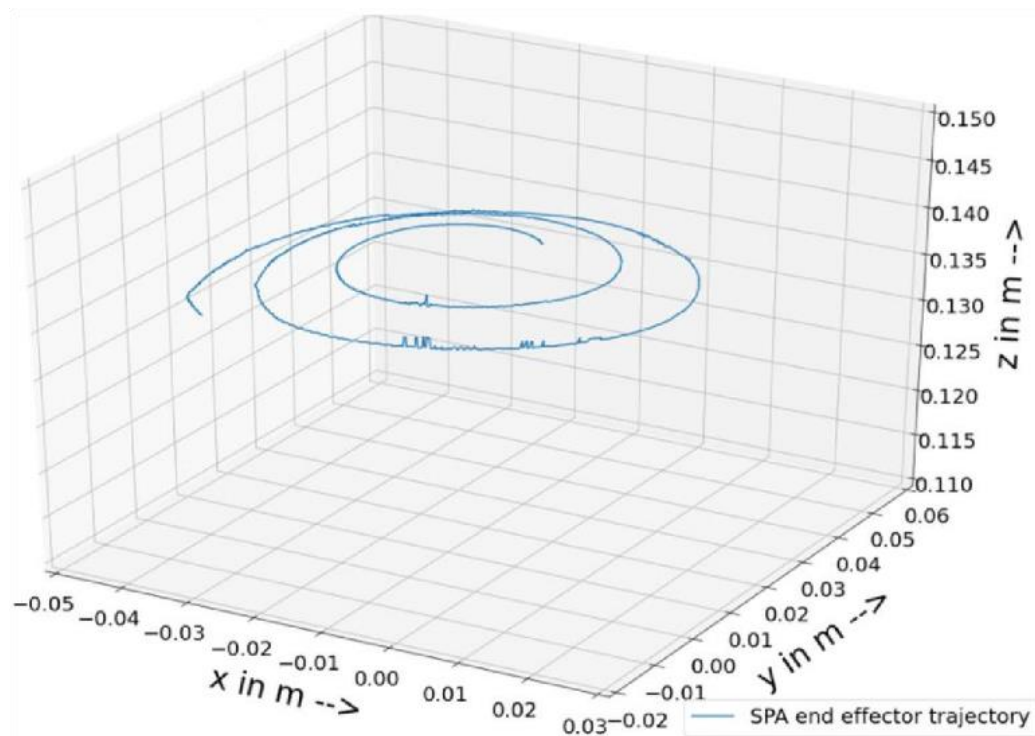


Рис. 7. Положения концевых захватных устройств при спиральном движении мягкого пневматического привода

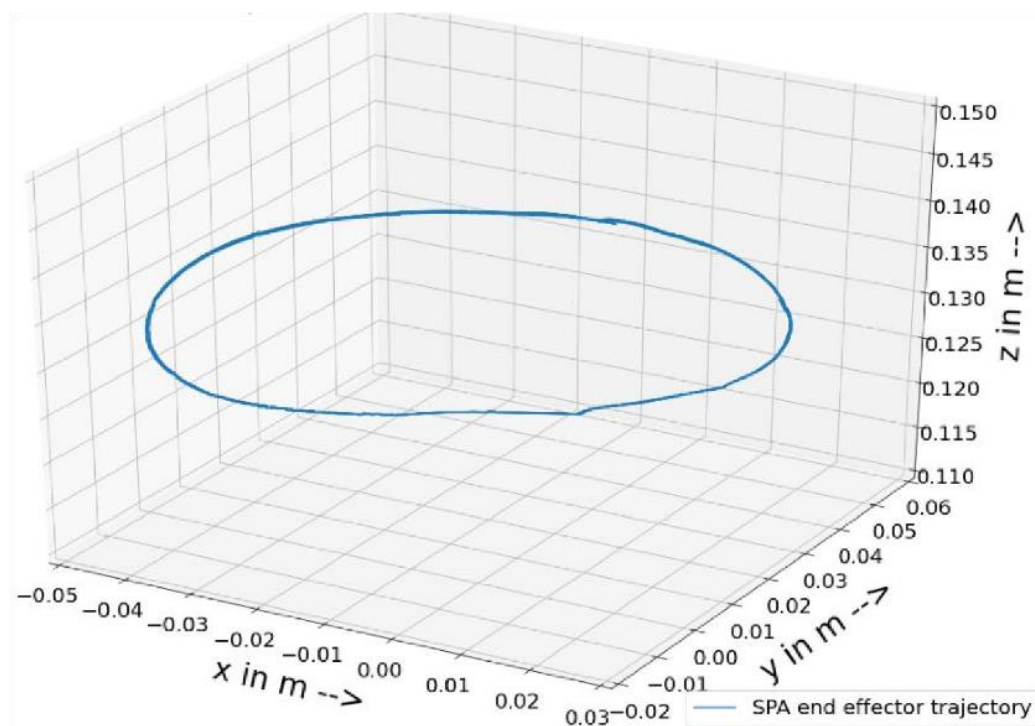


Рис. 8. Положения концевых захватных устройств при круговом движении мягкого пневматического привода

3.4. Смещение потока данных

Смещение данных – это явление изменения распределения набора данных в непрерывном потоке. Смещение потока данных является наиболее актуальной темой, так как его изучение затруднено из-за низкой квалификации, что иногда приводит к забвению ранее изученной гипотезы.

При онлайн-овом или инкрементном методе обучения данные всегда поступают одни за другими, поэтому в реальном мире они бесконечны. Предполагается, что реальный мир будет меняться из-за изменений в распределении, на которое влияет нелинейное поведение системы. Обычные рассуждения о смещении данных приводят к фиксации теоретической гипотезы [14], но в реальном мире изменение данных происходит неожиданно и непредсказуемо. Более того, это изменение принимает различные формы: меняются характеристики входных данных, изменяется связь между входными и целевыми данными и т.д. Существуют различные типы смещения, такие как ковариационный сдвиг, смещение концепции, распад модели и смещение данных [18]. Ковариационный сдвиг измеряет изменение распределения при изменении входных признаков. Смещение концепции означает изменение связи между входным признаком и выходным объектом. Распад модели снижает производительность ML-модели из-за смещения. Смещение данных может относиться к изменению любой из перечисленных концепций или их комбинации [15].

На рис. 9 показано смещение, обнаруженное в наборе данных circular1, где слева показаны выходные данные исполнительных механизмов X, Y и Z, а вертикальная линия показывает точки, учитываемые при определении смещения. Поскольку давление во всех трех камерах изменяется по синусоиде, положения конечных исполнительных механизмов рассматриваются для одинаковых значений давления, как показано на рис. 10. Смещение наблюдалось во всех трех направлениях движения концевой датчика. Смещение увеличивалось по осям x и z, а по оси y сначала наблюдался нисходящий, а затем восходящий тренд (Rev. 3, точка 2).

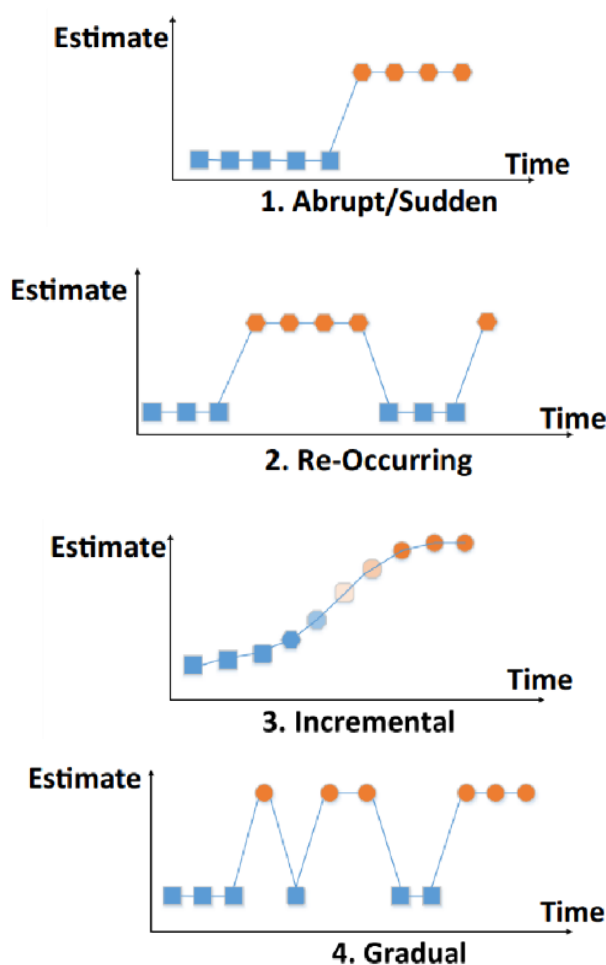


Рис. 9. Типы смещений, которые могут возникнуть в реальном сценарии [19]

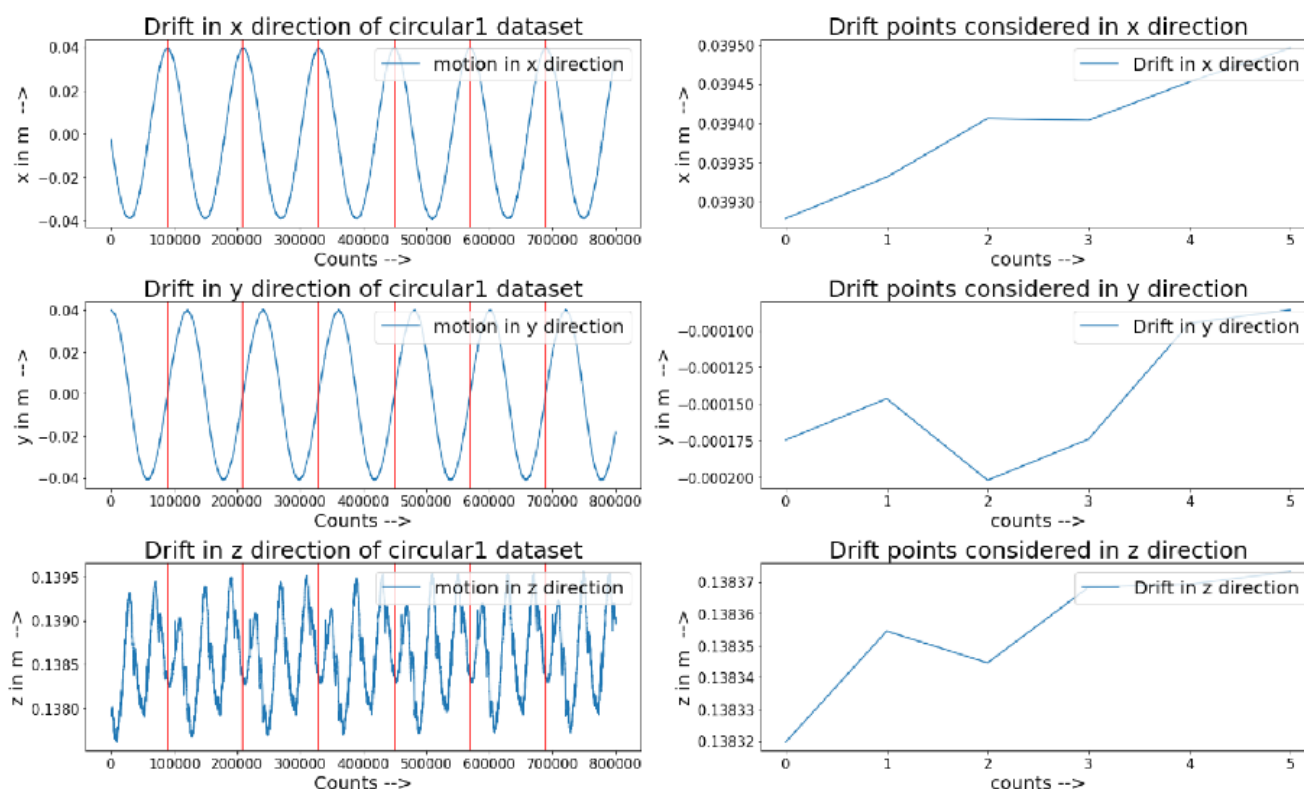


Рис. 10. Обнаружение смещений в наборе данных circular1

4. Результаты и обсуждение

Эксперименты проводились с использованием испытательного стенда для сбора данных о спиральных и круговых движениях конечного исполнительного механизма. Для эффективного обучения модели из набора данных были исключены исходные ложные сигналы о движении концевого захватного устройства. Основной целью эксперимента было предсказать эффективность инкрементно-обученной модели для будущего планирования траектории движения.

Эксперимент проводился в двух вариантах: сначала модель обучалась «с нуля», вычислялись ошибки реального времени или ошибки обучения, а также сохранялись выученные веса и прогнозировался выход в режиме предсказания с последующим вычислением ошибок. Во-вторых, использовался метод трансферного обучения. Модель обучалась на симулированных данных, затем проводилось дальнейшее инкрементное обучение с использованием данных и вычисленных ошибок реального времени или ошибок обучения. Обученные веса, полученные в результате обучения, сохранялись и загружались в объект модели для прогнозирования выхода, после чего производились расчеты ошибок. В табл. 2 приведены средние абсолютные ошибки различных алгоритмов ML.

Таблица 2

Сравнительная таблица средних абсолютных погрешностей

Средняя абсолютная ошибка					
Алгоритм	Наборы данных	Без предварительной подготовки		С предварительной подготовкой	
		Подготовка	Тестирование	Подготовка	Тестирование
KNN-регрессор	Спираль	0,00023086	0,010890343	1.76E-04	0,01089034
	Моделирование	4.25E-03	0,014948641	-	-
	Круг 1	3.82E-06	0,022515052	3.90E-06	0,02251506
	Круг 2	3.65E-06	0,021748876	3.90E-06	0,02226751

Средняя абсолютная ошибка					
Алгоритм	Наборы данных	Без предварительной подготовки		С предварительной подготовкой	
		Подготовка	Тестирование	Подготовка	Тестирование
Регрессор дерева решений	Спираль	2,85E+19	1.04E+19	1.90E+19	1.43E+19
	Моделирование	3.55E-03	0,006477737	-	-
	Круг 1	5.08E+19	3.80E+19	5.08E+19	4.95E+19
	Круг 2	5.08E+19	4.03E+19	5.08E+19	4.11E+19
Линейная регрессия	Спираль	8.80E-05	0,008816521	4.05E+11	5633802232
	Моделирование	1.07E-03	0,002352048	-	-
	Круг 1	2.23E-06	0,017343613	1.36E+09	2388458669
	Круг 2	2.07E-06	0,020323096	8.69E+08	874727820
Глубокое обучение	Спираль	7.91E-02	0,128890869	2.70E-02	0,13601386
	Моделирование	4.07E-02	0,011656918	-	-
	Круг 1	1.78E-03	0,023455277	1.05E-03	0,00437666
	Круг 2	1.79E-03	0.012201666	1.85E-03	0,00989459

Ошибки обучения для различных наборов данных различались в зависимости от используемого алгоритма. Алгоритмы выбирались по минимальным ошибкам в методе пакетного обучения, а для регрессионного анализа лучшие алгоритмы приведены в [34–38]. Все методы проверялись на наличие ошибок при обучении и тестировании. Средняя абсолютная ошибка использовалась из-за катастрофического забывания, которое наблюдалось в обучаемой модели, но при обучении предсказания ошибки были минимальными, за исключением дерева решений, которое показало низкие результаты. Четыре алгоритма IL были протестированы на четырех различных наборах данных.

4.1. Наборы данных

Обучение и тестирование проводилось путем последовательного разбиения набора данных на обучающий и тестовый. Например, набор данных *circular1* содержал 1 млн. точек данных, из которых первые 80 % были предназначены для обучения, а следующие 20 % – для тестирования (разделение на обучение и тестирование составляло 80-20 % для всех наборов данных). Точки данных последовательно разбивались для проверки того, что модель непрерывно предсказывает будущее на основе ранее усвоенных знаний, чтобы можно было осуществлять планирование траектории на основе инкрементного обучения. Было получено четыре набора данных: спираль, симуляция, «круг 1» и «круг 2». Все они были собраны на тестовом стенде, за исключением набора данных для моделирования.

4.2. К-тест ближайших соседей

Ошибки регрессии KNN оказались сравнительно меньше, чем у всех остальных сравниваемых алгоритмов, а метод пакетного обучения также привел к низким ошибкам. Ошибки обучения и ошибки прогнозирования в реальном времени варьировались в пределах нескольких микрометров. На рис. 11 показано сравнение ошибок между различными наборами данных с предварительным обучением и без предварительного обучения. В табл. 2 приведены ошибки алгоритма К-тест ближайших соседей в различных базах данных.

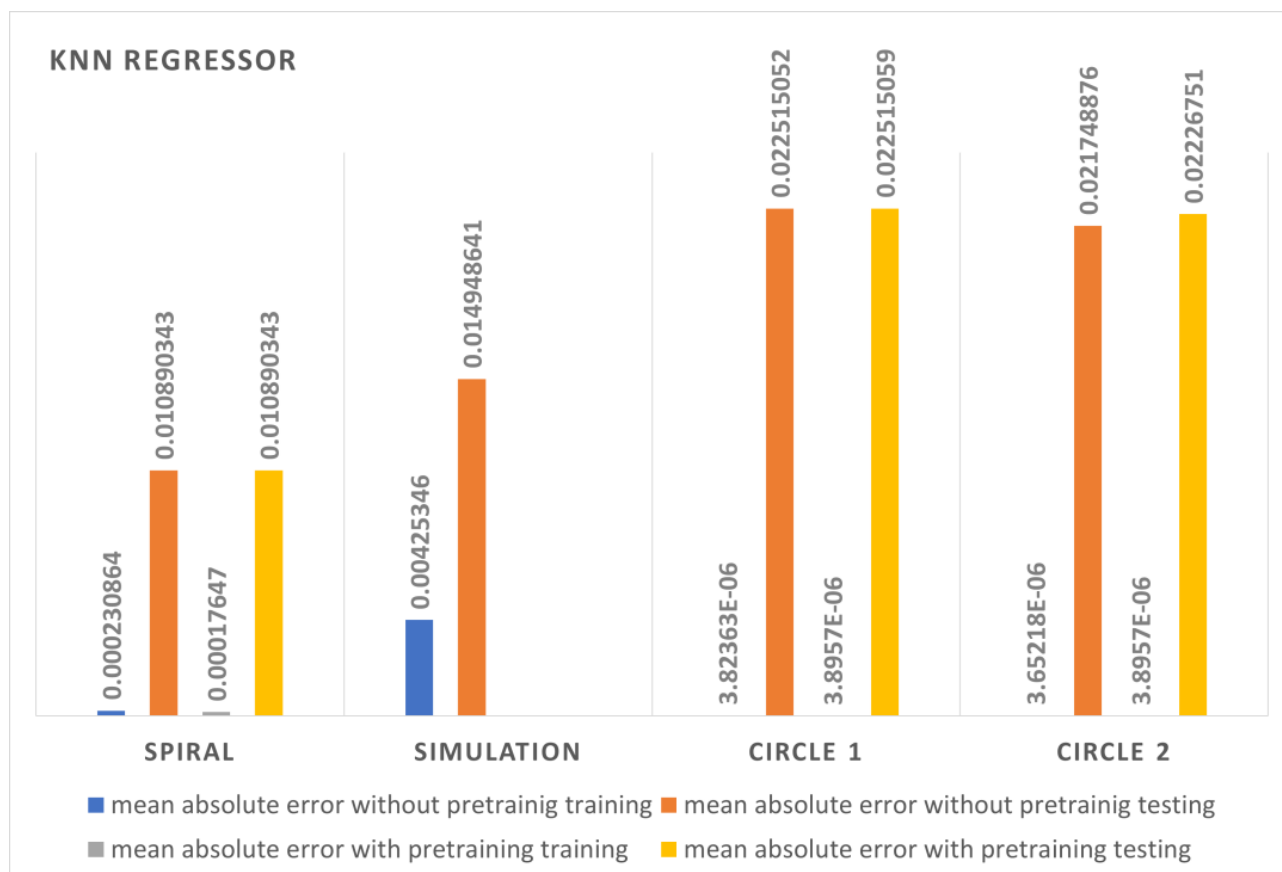


Рис. 11. График ошибок регрессора KNN при сравнении различных наборов данных с предварительным обучением и без него

Ошибки предсказания с ненаблюдаемыми данными приводили к несколько бóльшим ошибкам, поскольку модель обобщалась на обучающий набор данных и ранее полученные знания забывались. Ошибки обучения были ниже на наборе данных circular2 как с предварительным обучением, так и без него. На наборе данных «Спираль» ошибки тестирования были меньше, чем на всех остальных наборах данных как с предварительным обучением, так и без него. Алгоритм превзошел все остальные алгоритмы, однако было замечено, что KNN оказался более общим для обучающего набора данных, чем для непросматриваемых тестовых данных.

4.3. Тест «Дерево решений»

Алгоритм DT показал низкую производительность в методе инкрементного обучения, так как имел высокие показатели ошибок по сравнению со всеми остальными алгоритмами. Время обучения также было больше по сравнению с другими алгоритмами. Ошибки обучения также были высокими, и модель не была обобщающей для данных. На спиральных данных алгоритм DT показал среднюю производительность, но на имитационных данных результат оказался лучше, чем в других методах, поскольку данные были выбраны из рабочей области случайным образом. Ошибки, показанные на рис. 12, могут быть связаны с тем, что модель не обучается из-за высоких значений ошибок, а также с тем, что для каждого примера, который поступает, DT-модель требует построения модели заново. Следовательно, если данные не отражают структуру дерева, то они приведут к сбою.

4.4. Тест линейной регрессии

Тест линейной регрессии показал среднюю производительность по сравнению с регрессией KNN и нейросетевыми алгоритмами. Уровень ошибок находился между регрессором DT и нейронной сетью. На имитационном наборе данных ошибок было меньше, чем на других. Это объясняется случайностью данных. Спиральные данные также показали средний результат по сравнению с круговым набором данных, что свидетельствует о хорошем результате. Предварительно обученная модель вела себя хуже, чем модель с обычным обучением «с нуля». На рис. 13 показано сравнение

ошибок линейной регрессии для различных наборов данных с предварительным обучением и без него. Трансферное обучение оказалось неэффективным в качестве KNN-регрессора, а наоборот, отклонялось от реального движения конечного исполнительного механизма. При выводе данных моделирования было получено меньше ошибок по сравнению со всеми остальными алгоритмами. Круговые наборы данных дали больше ошибок, несмотря на большее количество точек данных.

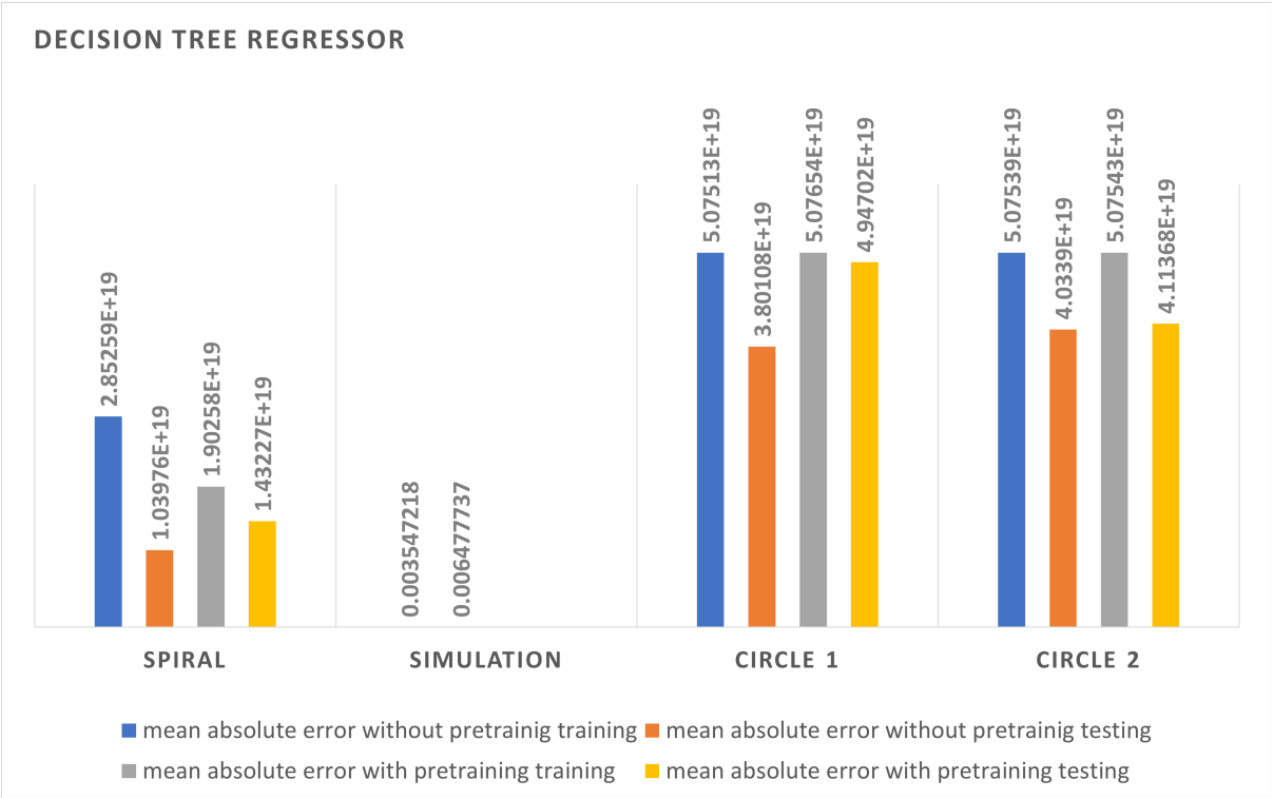


Рис. 12. График ошибок регрессора дерева решений при сравнении различных наборов данных с предварительным обучением и без него

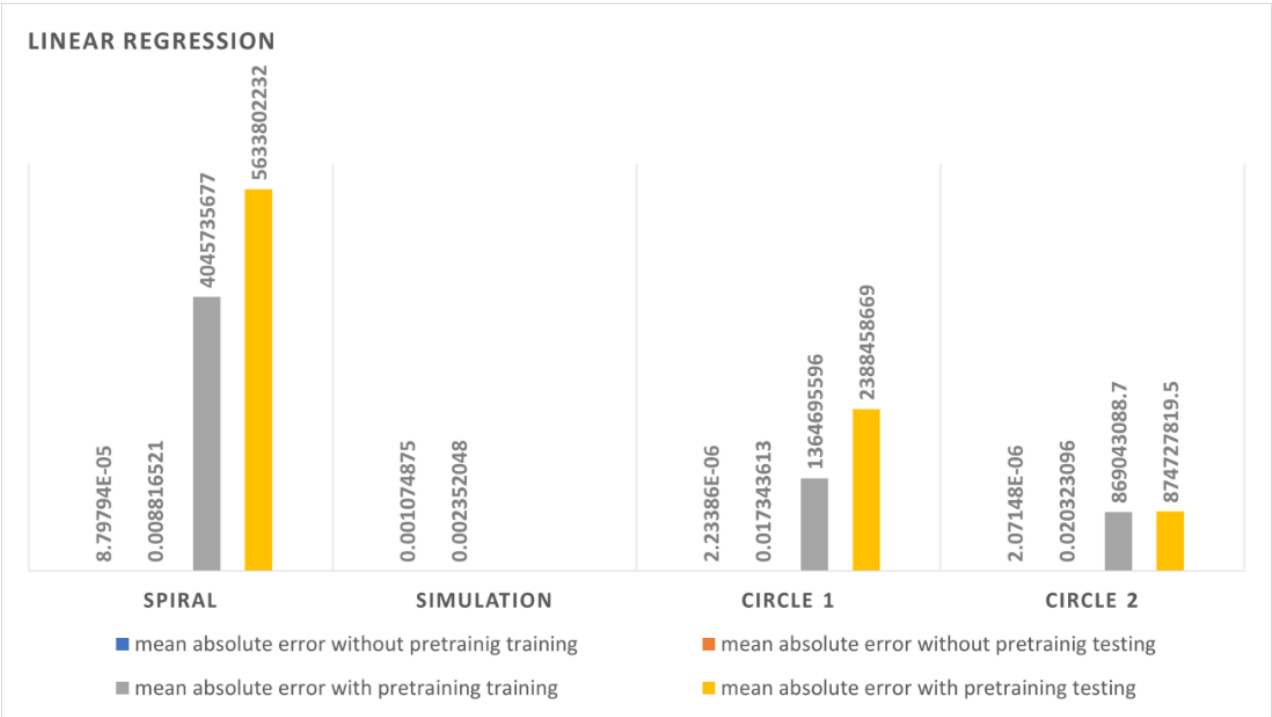


Рис. 13. График ошибок линейного регрессора при сравнении различных наборов данных с предварительным обучением и без него

4.5. Тест глубокой нейронной сети

Нейронная сеть показала хорошие результаты, но уровень ошибок был снижен по сравнению с методами DT и LR. Обучение проводилось на всех наборах данных, на которых была изображена нейронная сеть, поскольку ей требуется больше наборов данных для обучения по сравнению с другими алгоритмами. Как видно из рис. 14, уровень ошибок на спиральном наборе данных был выше, так как данных для обучения было меньше, и уменьшался по мере продвижения к набору данных «круг 2». Нейронная сеть была более обобщенной по отношению к обучающему набору данных. Нейронная сеть лучше работала с обоими круговыми наборами данных, так как имела достаточное количество данных для обучения. Предварительное обучение оказало большее влияние на нейронную сеть и показало меньшее количество ошибок. В целом нейронная сеть показала, что она может адаптироваться к изменениям в потоке данных, поскольку каждый вес в нейронных сетях адаптируется к каждому новому опыту, а веса также зависят от функции ошибки. Это позволяет нейронной сети сравнительно легко обновлять веса.

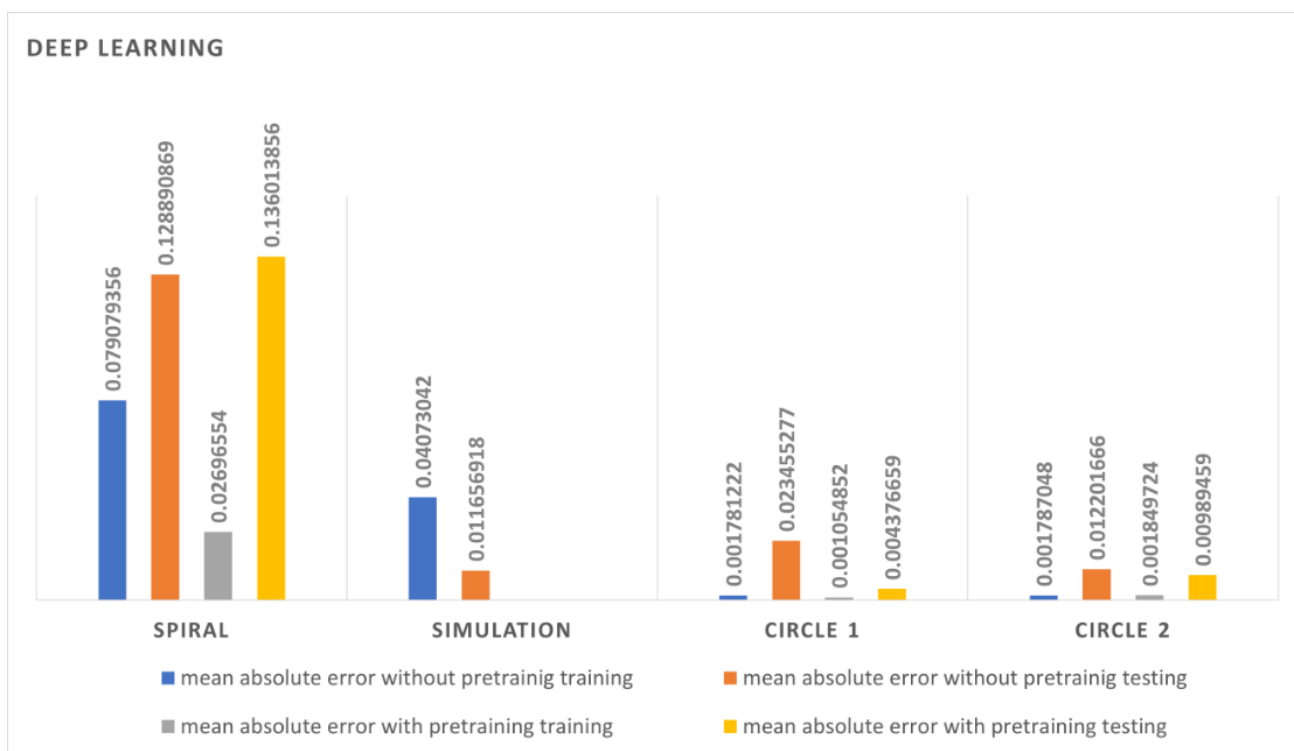


Рис. 14. График ошибок нейронной сети при сравнении различных наборов данных с предварительным обучением и без него

4.6. Влияние предварительного обучения на коэффициент ошибок

Обучение нейронной сети «с нуля» оказалось неэффективным для алгоритмов ML и хорошим для метода глубокого обучения, как следует из приведенного выше раздела. График ошибок отражает среднее значение ошибки в конце обучения модели. Оказалось, что трансферное обучение позволяет снизить уровень ошибок искусственной нейронной сети [15]. На рис. 15 показаны показатели ошибок в процессе обучения всех моделей.

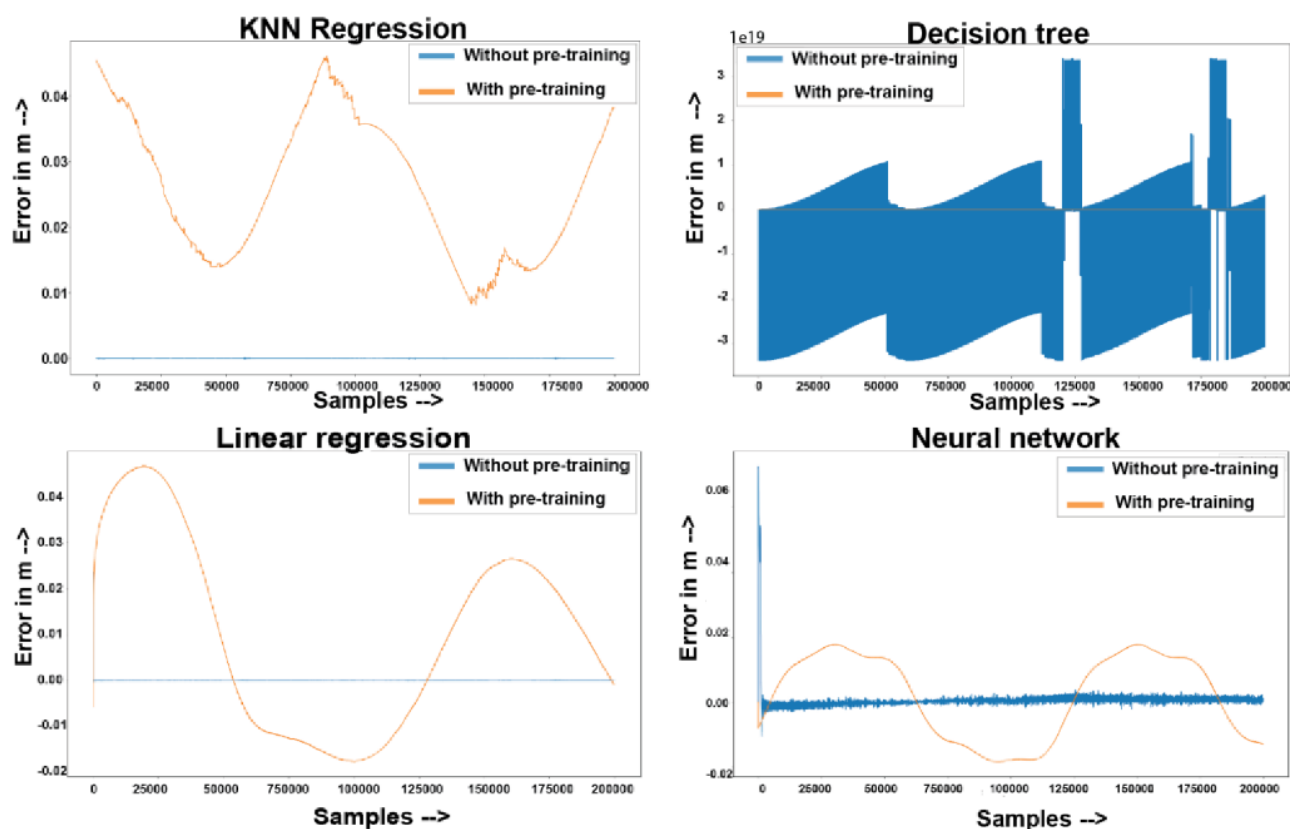


Рис. 15. График ошибок обучения для всех четырех алгоритмов IL с предварительным обучением и без него для набора данных circular1

Ошибки без предварительного обучения имели тенденцию к уменьшению, за исключением DT, где модель при инкрементном обучении старалась обновлять веса чаще, чем при пакетном методе. Метод трансферного обучения показал синусоидальное уменьшение ошибки. Это объясняется наличием уже обобщенных предварительно обученных весов и определенным рабочим пространством с различными движениями конечных захватных устройств. Начальные ошибки в KNN и LR были высокими, так как для обновления весов использовался метод стохастического градиентного спуска, который сходил быстрее, поэтому модели стремились быстрее слиться и имели тенденцию к накоплению отрицательных ошибок, и это продолжалось, при этом и KNN, и LR показывали снижение уровня ошибок для последовательных пиков и стремились к сближению. Нейронная сеть без предварительного обучения пыталась сходиться с первоначально высокой ошибкой. Нейронная сеть с предварительным обучением показала снижение ошибки по сравнению со всеми остальными алгоритмами. Это произошло потому, что трансферное обучение наиболее эффективно в нейронных сетях, использующих алгоритмы градиентного спуска. Градиентный спуск обновляет веса сети после каждой эпохи, но с исходными предварительно обученными весами модель сходится медленнее, чем со случайными весами, а иногда и вовсе не сходится. DT показал положительную реакцию на предварительное обучение, но уровень ошибок без предварительного обучения был высоким. В целом трансферное обучение оказало негативное влияние на инкрементное обучение с помощью KNN и LR. Однако нейронная сеть показала хорошие результаты в снижении начальной ошибки, хотя модель сходилась с трудом.

5. Выводы и дальнейшая работа

В данном эксперименте для прогнозирования движения конечного исполнительного механизма с давлением на входе и координатами в реальном времени на выходе были протестированы обнаружение смещения в потоковых данных и четыре модели инкрементного обучения для мягкого пневматического привода. Смещение было обнаружено при сравнении движений конечного исполнительного механизма и учета определенных точек давления из входного давления. Сначала были проведены два типа тестов без предварительного обучения моделей, а затем с предварительным

обучением, а также рассчитаны ошибки обучения и тестирования между метками и выходными данными моделей. Среди всех алгоритмов наилучшие результаты показал регрессор KNN с меньшей ошибкой при обучении и тестировании. Нейросетевая модель также показала лучшие результаты, чем DT и LR при переносе и обучении. При этом модель не была обобщена на обучающие данные, а ошибки тестирования нейронной сети по сравнению с ошибками других низкоэффективных алгоритмов были хорошими. Регрессор дерева решений показал отклонение от нормального поведения со значительными ошибками при обучении и тестировании. Алгоритм линейной регрессии показал хорошие результаты при обучении, но его обобщенность на обучающий набор данных была шире, поэтому были получены большие ошибки при тестировании. В целом регрессор KNN превзошел все остальные алгоритмы, хотя разница в ошибках тестирования по сравнению с предварительным обучением была больше. Ошибки тестирования нейронной сети были меньше и объяснялись попыткой удержать уже полученные знания.

Было обнаружено смещение в движении конечного звена привода, и необходимо провести дополнительную работу по изучению методов обнаружения смещения для выявления типов смещения. KNN и нейронная сеть показали отличные результаты в отношении ошибок тестирования, но, поскольку KNN плохо справлялась с неизвестными тестовыми данными из-за забывания, в дальнейшей работе следует рассмотреть нейронную сеть с новой средой, подходящей для инкрементного обучения.

Вклад авторов: Концептуализация, Ю.К.; Методология, Ю.К.; Программное обеспечение, Ю.К. и С.К.; Валидация, Ю.К.; Формальный анализ, Ю.К.; Исследование, Ю.К.; Ресурсы, И.В.; Курирование данных, Е.О. и С.К.; Написание первоначального варианта, Ю.К. и Е.О. Все авторы прочитали и согласились с опубликованной версией рукописи.

Финансирование: Данное исследование не получало внешнего финансирования.

Заявление Совета по институциональному надзору: Не применимо.

Положение об информированном согласии: Не применимо.

Заявление о доступности данных: Не применимо.

Конфликты интересов: Авторы заявляют об отсутствии конфликта интересов.

Сокращения

SRs	Мягкие роботы
ML	Машинное обучение
KNN	К-ближайшие соседи
DT	Дерево решений
LR	Линейная регрессия
NN	Нейронная сеть
CAD	Автоматизированное проектирование

ЛИТЕРАТУРА

1. Lutonin, A.; Shklyarskiy, J. Topology and control algorithms for a permanent magnet synchronous motor as a part of a vehicle with in-wheel motors. In Proceedings of the E3S Web of Conferences. EDP Sci. 2021, 266, 04001.
2. Holkar, K.; Waghmare, L.M. An overview of model predictive control. *Int. J. Control Autom.* 2010, 3, 47–63.
3. Brigadnov, I.; Lutonin, A.; Bogdanova, K. Error State Extended Kalman Filter Localization for Underground Mining Environments. *Symmetry* 2023, 15, 344. <https://doi.org/10.3390/sym15020344>
4. Yapar, C.; Levie, R.; Kutyniok, G.; Caire, G. Real-time outdoor localization using radio maps: A deep learning approach. *arXiv* 2021, arXiv:2106.12556. <https://doi.org/10.1109/TWC.2023.3273202>
5. Xu, W.; Cai, Y.; He, D.; Lin, J.; Zhang, F. Fast-lid2: Fast direct lidar-inertial odometry. *IEEE Trans. Robot.* 2022, 38, 2053–2073. <https://doi.org/10.1109/TRO.2022.3141876>
6. Kakani, V.; Nguyen, V.H.; Kumar, B.P.; Kim, H.; Pasupuleti, V.R. A critical review on computer vision and artificial intelligence in food industry. *J. Agric. Food Res.* 2020, 2, 100033. <https://doi.org/10.1016/j.jafr.2020.100033>
7. Sankowski, D.; Nowakowski, J. *Computer Vision in Robotics and Industrial Applications*; World Scientific: Singapore, 2014; Volume 3.
8. Tian, H.; Wang, T.; Liu, Y.; Qiao, X.; Li, Y. Computer vision technology in agricultural automation—A review. *Inf. Process. Agric.* 2020, 7, 1–19. <https://doi.org/10.1016/j.inpa.2019.09.006>
9. Bertolini, M.; Mezzogori, D.; Neroni, M.; Zammori, F. Machine Learning for industrial applications: A comprehensive literature review. *Expert Syst. Appl.* 2021, 175, 114820. <https://doi.org/10.1016/j.eswa.2021.114820>
10. Kashyap, P. Industrial applications of machine learning. In *Machine Learning for Decision Makers: Cognitive Computing Fundamentals for Better Decision Making*; Apress: Tokyo, Japan, 2017; pp. 189–233.

11. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* 2018, 4, e00938. <https://doi.org/10.1016/j.heliyon.2018.e00938>
12. Zhou, H.; Zhao, H.; Zhang, Y. Nonlinear system modeling using self-organizing fuzzy neural networks for industrial applications. *Appl. Intell.* 2020, 50, 1657–1672. <https://doi.org/10.1007/s10489-020-01645-z>
13. Chen, F.; Wang, M.Y. Design optimization of soft robots: A review of the state of the art. *IEEE Robot. Autom. Mag.* 2020, 27, 27–43. <https://doi.org/10.1109/MRA.2020.3024280>
14. Takishima, Y.; Yoshida, K.; Khosla, A.; Kawakami, M.; Furukawa, H. Fully 3D-printed hydrogel actuator for jellyfish soft robots. *ECS J. Solid State Sci. Technol.* 2021, 10, 037002.
15. Elango, N.; Faudzi, A.A.M. A review article: Investigations on soft materials for soft robot manipulations. *Int. J. Adv. Manuf. Technol.* 2015, 80, 1027–1037.
16. Wang, H.; Zhang, R.; Chen, W.; Wang, X.; Pfeifer, R. A cable-driven soft robot surgical system for cardiothoracic endoscopic surgery: Preclinical tests in animals. *Surg. Endosc.* 2017, 31, 3152–3158. <https://doi.org/10.1007/s00464-016-5340-9>
17. Runciman, M.; Darzi, A.; Mylonas, G.P. Soft robotics in minimally invasive surgery. *Soft Robot.* 2019, 6, 423–443.
18. Zakharov, L.A.; Martyushev, D.A.; Ponomareva, I.N. Predicting dynamic formation pressure using artificial intelligence methods. *J. Min. Inst.* 2022, 253, 23–32.
19. Zhukovskiy, Y.L.; Kovalchuk, M.S.; Batueva, D.E.; Senchilo, N.D. Development of an Algorithm for Regulating the Load Schedule of Educational Institutions Based on the Forecast of Electric Consumption within the Framework of Application of the Demand Response. *Sustainability* 2021, 13, 13801. <https://doi.org/10.3390/su132413801>
20. Romashev, A.; Iakovleva, T.; Mashevsky, G. Mining informational and analytical bulletin. *Sci. Tech. J.* 2022, 6, 175–188.
21. Luis, E.; Pan, H.M.; Sing, S.L.; Bastola, A.K.; Goh, G.D.; Goh, G.L.; Tan, H.K.J.; Bajpai, R.; Song, J.; Yeong, W.Y. Silicone 3D printing: Process optimization, product biocompatibility, and reliability of silicone meniscus implants. *3D Print. Addit. Manuf.* 2019, 6, 319–332.
22. Nagymate, G.; Kiss, R.M. Application of OptiTrack motion capture systems in human movement analysis: A systematic literature review. *Recent Innov. Mechatron.* 2018, 5, 1–9.
23. Kramer, O.; Kramer, O. K-nearest neighbors. In *Dimensionality Reduction with Unsupervised Nearest Neighbors*; IEEE: Honolulu, HI, USA, 2013; pp. 13–23.
24. Larose, D.T. K-nearest neighbor algorithm. In *Discovering Knowledge in Data: An Introduction to Data Mining*; John Wiley & Sons: Hoboken, NJ, USA, 2005; pp. 104–106. <https://doi.org/10.1002/0471687545.ch5>
25. Zemenkova, M.Y.; Chizhevskaya, E.L.; Zemenkov, Y.D. Intelligent monitoring of the condition of hydrocarbon pipeline transport facilities using neural network technologies. *J. Min. Inst.* 2022, 258, 933–944. <https://doi.org/10.31897/PMI.2022.105>
26. Alanazi, A.K.; Alizadeh, S.M.; Nurgalieva, K.S.; Nesic, S.; Grimaldo Guerrero, J.W.; Abo-Dief, H.M.; Eftekhari-Zadeh, E.; Nazemi, E.; Narozhnyy, I.M. Application of Neural Network and Time-Domain Feature Extraction Techniques for Determining Volumetric Percentages and the Type of Two Phase Flow Regimes Independent of Scale Layer Thickness. *Appl. Sci.* 2022, 12, 1336. <https://doi.org/10.3390/app12031336>
27. Ushakov, E.; Aleksandrova, T.; Romashev, A. Neural network modeling methods in the analysis of the processing plant's indicators. In *International Scientific Conference Energy Management of Municipal Facilities and Sustainable Energy Technologies EMMFT 2019*; Springer: Berlin/Heidelberg, Germany, 2021; Volume 2, pp. 36–45.
28. Bottou, L. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, 7th ed.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 421–436.
29. Amari, S.I. Backpropagation and stochastic gradient descent method. *Neurocomputing* 1993, 5, 185–196. [https://doi.org/10.1016/0925-2312\(93\)90006-O](https://doi.org/10.1016/0925-2312(93)90006-O)
30. Filippov, E.V.; Zaharov, L.A.; Martyushev, D.A.; Ponomareva, I.N. Reproduction of reservoir pressure by machine learning methods and study of its influence on the cracks formation process in hydraulic fracturing. *J. Min. Inst.* 2022, 258, 924–932. <https://doi.org/10.31897/PMI.2022.103>
31. Islamov, S.; Grigoriev, A.; Beloglazov, I.; Savchenkov, S.; Gudmestad, O.T. Research Risk Factors in Monitoring Well Drilling—A Case Study Using Machine Learning Methods. *Symmetry* 2021, 13, 1293. <https://doi.org/10.3390/sym13071293>
32. Montiel, J.; Halford, M.; Mastelini, S.M.; Bolmier, G.; Sourty, R.; Vaysse, R.; Zouitine, A.; Gomes, H.M.; Read, J.; Abdessalem, T.; et al. River: Machine learning for streaming data in python. *J. Mach. Learn. Res.* 2021, 22, 4945–4952.
33. Vasilev, I.; Slater, D.; Spacagna, G.; Roelants, P.; Zocca, V. *Python Deep Learning: Exploring Deep Learning Techniques and Neural Network Architectures with Pytorch, Keras, and TensorFlow*; Packt Publishing Ltd.: Birmingham, UK, 2019.
34. Kim, D.; Kim, S.H.; Kim, T.; Kang, B.B.; Lee, M.; Park, W.; Ku, S.; Kim, D.; Kwon, J.; Lee, H.; et al. Review of machine learning methods in soft robotics. *PLoS ONE* 2021, 16, e0246102. <https://doi.org/10.1371/journal.pone.0246102>
35. Sun, W.; Akashi, N.; Kuniyoshi, Y.; Nakajima, K. Physics-informed recurrent neural networks for soft pneumatic actuators. *IEEE Robot. Autom. Lett.* 2022, 7, 6862–6869. <https://doi.org/10.1109/LRA.2022.3178496>
36. Sultanbekov, R.; Beloglazov, I.; Islamov, S.; Ong, M.C. Exploring of the Incompatibility of Marine Residual Fuel: A Case Study Using Machine Learning Methods. *Energies* 2021, 14, 8422. <https://doi.org/10.3390/en14248422>
37. Brilliant, L.; Zavialov, A.; Danko, M.; Andronov, K.; Shpurov, I.; Bratkova, V.; Davydov, A. Integration of machine learning methods and geological and hydrodynamic modeling in field development design (Russian). *Oil Ind. J.* 2022, 2022, 48–53.
38. Romashev, A.O.; Nikolaeva, N.V.; Gatiatullin, B.L. Adaptive approach formation using machine vision technology to determine the parameters of enrichment products deposition. *J. Min. Inst.* 2022, 256, 677–685. <https://doi.org/10.31897/PMI.2022.77>