УДК 004.032.26

Суворов Иван Сергеевич

магистрант направления подготовки 09.04.03 "Прикладная информатика", Сочинский государственный университет, Сочи

Макарова Ирина Леонидовна

кандидат технических наук, доцент, Сочинский государственный университет, Сочи

ИСПОЛЬЗОВАНИЕ ОТНОСИТЕЛЬНОГО ЕВКЛИДОВА РАССТОЯНИЯ В КАЧЕСТВЕ ФУНКЦИИ ПОТЕРЬ

Аннотация. В данной научной работе исследуется применение относительного евклидового расстояния (ОЕР), как функции потерь для задач машинного обучения. ОЕР представляет собой модификацию стандартной евклидовой метрики, которая учитывает различия между объектами через отношение их расстояний до центра кластера или среднего значения выборки. Введение этой меры позволяет более точно оценивать ошибки предсказания в условиях неоднородности данных.

Annotation. This scientific paper explores the application of relative Euclidean distance (RER) as a loss function for machine learning tasks. The OER is a modification of the standard Euclidean metric, which takes into account the differences between objects through the ratio of their distances to the cluster center or the average value of the sample. The introduction of this measure makes it possible to more accurately estimate prediction errors in conditions of heterogeneity of data.

Ключевые слова: нейронные сети; функция потерь; евклидово расстояние; относительное евклидово расстояние.

Keywords: neural networks; loss function; euclidean distance; relative euclidean distance.

Модификации функций потерь (или критериев оптимизации) являются актуальной задачей в области машинного обучения. Они играют ключевую роль в процессе обучения нейронной сети, так как они определяют, насколько хорошо модель предсказывает результаты по сравнению с реальными данными. Современные задачи машинного обучения, как правило, требуют использования глубоких нейронных сетей для решения задач распознавания изображений, обработки естественного языка, прогнозирования временных рядов и других сложных задач. Модели становятся все более сложными и это требует разработки новых подходов к оптимизации и выбору функций потерь, которые могут лучше учитывать специфику данных и архитектуру модели.

Основная цель модификации функции потерь состоит в том, чтобы сделать процесс обучения более эффективным и стабильным, а также соотвествовать требованиям конкретной задачи, обеспечивая при этом наилучшие возможные результаты на новых данных.

Оценка ошибок в машинном является ключевой метрикой в определении качества модели, а также ее способности делать точные прогнозы на новых данных. Существует множество методов оценки ошибок, каждый из которых подходит для определенных типов задач. Вот некоторые из них: Среднеквадратичная ошибка (MSE), Средняя абсолютная ошибка (MAE), Точность (Accuracy), F-мера, ROC-кривая и AUC, L-норма.

Для того, чтобы проиллюстрировать работу относительного евклидова расстояния в качестве функции потерь, проведём сравнительный анализ между несколькими смежными метриками. Для примера возьмём, ещё L-норму и абсолютное евклидово расстояние.

L-норма - это математическая мера, которая используется для того, чтобы измерить длину вектора или расстояния между двумя точками в n-мерном пространстве. Она определяется как [1]:

$$\|\boldsymbol{x}\|_{\mathbf{1}} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} |\boldsymbol{x_i}|$$

Асболютное Евклидово расстояние - является мерой расстояния между двумя точками в многомерном пространстве. Евклидово расстояние соответствует длине кратчайшего пути между двумя точками, если двигаться по прямой линии. Эта формула является частным случаем более общей формулы для L-p норм при p = 2. Оно рассчитывается по формуле [2]:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

Относительное Евклидово расстояние – это мера различия между двумя векторами, которая учитывает масштаб данных [3].

Формула для вычисления относительного Евклидова расстояния между двумя n-мерными векторами выглядит следующим образом:

$$\boldsymbol{d(p,q) = \frac{1}{\sqrt{n}} \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}}$$

Эксперимент проведём при помощи Google Colab. Google Colab (Collaborative Notebooks) – это облачная платформа от Google, которая предоставляет бесплатный доступ к Jupyter Notebook через браузер. Основная цель Colab заключается в упрощении процесса разработки, тестирования и совместного использования кода, особенно в области машинного обучения и анализа данных [4].

Также воспользуемся библиотекой PyTorch. PyTorch – это библиотека для языка программирования Python предназначенная для создания и обучения нейронных сетей, а также для выполнения других задач, связанных с машинным обучением [5].

С помощью трёх метрик функции потерь, решим задачу восстановления графика функции среди искусственного шума. Импортируем необходимые библиотеки:

import torch

import matplotlib.pyplot as plt

import math

К примеру, будем восстанавливать график такой функции:

$$y = 2^x * sin\,(2^{-x})$$

def target_function(x):

*return 2**x * torch.sin(2**-x) # задача востановить график данной функции среди шума*

Возьмём простую нейронную сеть с 3 слоями и функцией активации сигмоида. Нейронов в скрытом слое 200. Нейросеть имеет вид:

class RegressionNet(torch.nn.Module):

def __init__(self, n_hidden_neurons):

super(RegressionNet, self).__init__()

self.fc1 = torch.nn.Linear(1, n_hidden_neurons)

self.act1 = torch.nn.Sigmoid() # функция активации сигмоида

self.fc2 = torch.nn.Linear(n_hidden_neurons, n_hidden_neurons)

self.act2 = torch.nn.Sigmoid()

self.fc3 = torch.nn.Linear(n_hidden_neurons, 1)

def forward(self, x):

x = self.fc1(x)

x = self.act1(x)

x = self.fc2(x)

x = self.act2(x)

```
        x = self.fc3(x)
        return x

net = RegressionNet(200) # количество нейронов в скрытом слое
```

Создадим шум:

```
x_train =  torch.linspace(-10, 5, 100)
y_train = target_function(x_train)
noise = torch.randn(y_train.shape) / 20. # создание шума
y_train = y_train + noise
x_train.unsqueeze_(1)
y_train.unsqueeze_(1)

x_validation = torch.linspace(-10, 5, 100)
y_validation = target_function(x_validation)
x_validation.unsqueeze_(1)
y_validation.unsqueeze_(1)
# ------Dataset preparation end--------:
```

Оптимизатор – Adam :

```
optimizer = torch.optim.Adam(net.parameters(), lr=0.01) #используем оптимизатор адам "золотой стандарт"
```

Проинициализируем работу нейросети с количеством шагов 1000:

```
for epoch_index in range(1000):
    optimizer.zero_grad()

    y_pred = net.forward(x_train)
    loss_value = loss(y_pred, y_train)
    loss_value.backward()

    optimizer.step()
```

Будем использовать разные функции потерь.

1. L-норма (рис. 1)

```
def loss(pred, target):
  squares = abs(pred - target)
  return squares.mean()
```

2. Абсолютное Евклидово расстояние (рис. 2)

```
def loss(pred, target):
    squares = torch.sqrt((pred - target)**2)
    return squares.mean()
```

3. Относительное Евклидово расстояние (рис. 3)

```
def loss(pred, target):
    squares = (1/(math.sqrt(n)))*torch.sqrt((pred - target)**2)
    return squares.mean()
```

За n будем брать количество нейронов в скрытом слое. Все функции потерь с той или иной точностью, обучили нейросеть справляться с задачей восстановления графика функции среди шума.

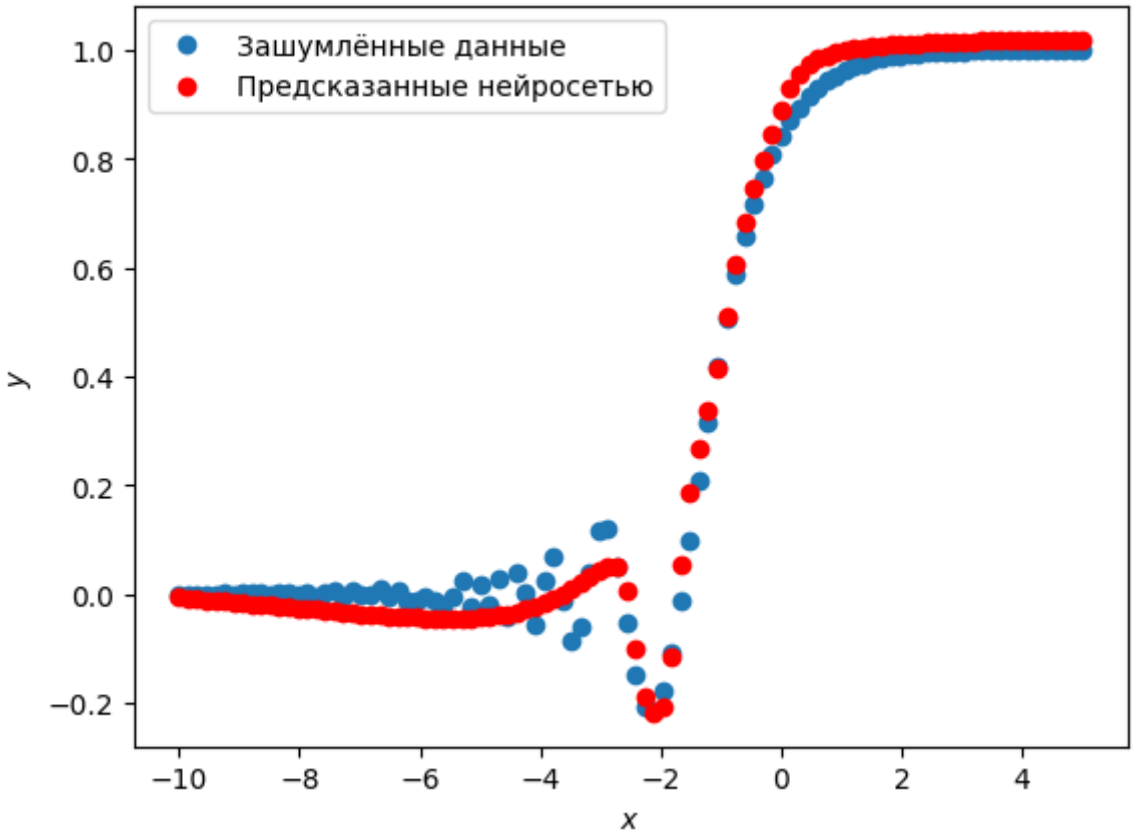

Рисунок 1. L-норма

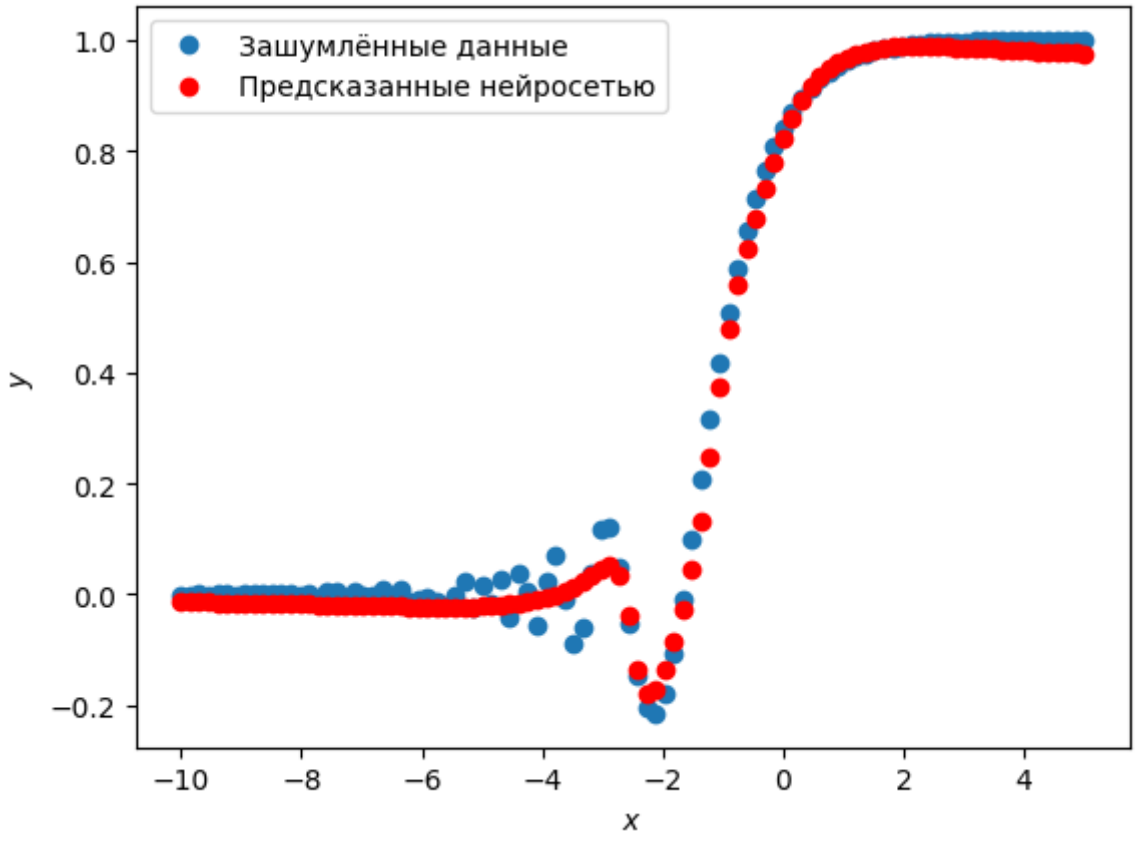

Рисунок 2. Абсолютное Евклидово расстояние

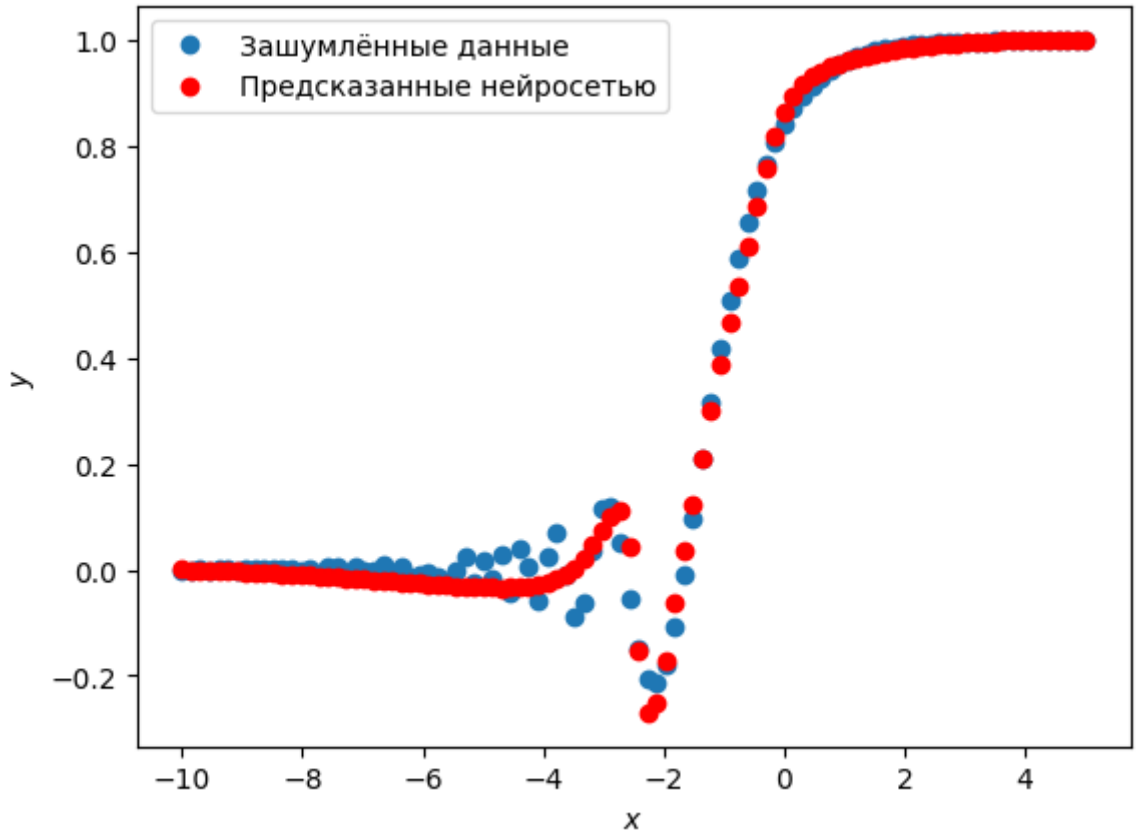

Рисунок 3. Относительное евклидово расстояние

Также для того, чтобы определить какая из метрик является эффективной, проведём обучение нейросети 10 раз по каждой функции потерь. Получим следующие результаты (Таблица 1):

Таблица 1. Данные экспериментов

L-норма	время работы	Абсолютное Евклидово расстояние	время работы	Относительное евклидово расстояние	время работы
0,0238028950989246	1,50	0,018648095	2,2	0,024164700880646706	2,6
0,0190999936312437	2,1	0,018394733	2,2	0,02674816	2,2
0,0237265136092901	2,6	0,024674751	2,2	0,02647137	2,9
0,0171601232141256	2,6	0,018965697	2,6	0,02007347	2,2
0,0214741248637437	2,1	0,023206495	2,2	0,01784649	3
0,0263147298246622	3,1	0,020065524	2,6	0,01949364	2,2
0,0316197387874126	2,1	0,019133974	2,2	0,02028721	2,7
0,0188079737126827	2,1	0,025001364	2,2	0,02247979	2,5
0,0200644973665475	2,1	0,028422356	2,7	0,02030828	3,2
0,0217852815985679	3,8	0,017140625	2,2	0,02028661	2,7
0,022386	2,41	0,021365361	2,33	0,01939950	2,62

На основании проведённого эксперимента было установлено, что среди трёх исследуемых функций потерь наилучшие результаты получены при использовании относительного евклидова расстояния. Эта метрика показала наиболее стабильные и точные прогнозы, что подтверждает её эффективность в контексте данной задачи.

L-норма и абсолютное евклидово расстояние также были протестированы, однако они уступили по качеству прогнозируемых значений. Использование L-нормы привело к повышенной чувствительности модели к выбросам, что негативно сказалось на общей точности. Абсолютное евклидово расстояние, хотя и продемонстрировало неплохие результаты, всё же оказалось менее эффективным по сравнению с относительным евклидовым расстоянием [6].

Таким образом, в рамках данного исследования рекомендуется использовать относительное евклидово расстояние в качестве функции потерь для достижения максимальной точности и стабильности модели [7].

В данной работе было проведено исследование использования относительного Евклидова расстояния в качестве функции потерь для задач классификации и регрессии. Анализ показал, что данный подход обладает рядом преимуществ по сравнению с традиционными методами, такими как L-норма. Относительное Евклидово расстояние позволяет учитывать не только величину ошибок, но и их распределение относительно истинных значений, что делает его особенно полезным в ситуациях, когда необходимо минимизировать влияние выбросов и обеспечить устойчивость модели к шумам.

Эксперименты на различных наборах данных подтвердили эффективность предложенного метода. Результаты показали улучшение качества предсказаний по сравнению с классическими функциями потерь, особенно в случаях, где данные содержат значительные выбросы или имеют асимметричное распределение. Это открывает новые перспективы для применения данного подхода в реальных приложениях, связанных с обработкой больших объемов данных и построением сложных моделей машинного обучения.

Таким образом, результаты исследования свидетельствуют о том, что использование относительного Евклидова расстояния в качестве функции потерь представляет собой перспективный метод оптимизации моделей, который заслуживает дальнейшего изучения и внедрения в практику.

Список литературы.

1. L-норма [Электронный ресурс]. https://ru.stackoverflow.com/questions/1124094/l1-%D0%B8-l2-%D1%80%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F-l1-%D0%B8-l2-%D0%BD%D0%BE%D1%80%D0%BC%D0%B0 (дата обращения 8.01.2026)
2. Абсолютное Евклидово Расстояние [Электронный ресурс]. https://wiki.loginom.ru/articles/euclid-distance.html (дата обращения 8.01.2026)
3. Относительное Евклидово расстояние [Электронный ресурс].https://science.fandom.com/ru/wiki/%D0%9E%D1%82%D0%BD%D0%BE%D1%81%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%B5%D0%B2%D0%BA%D0%BB%D0%B8%D0%B4%D0%BE%D0%B2%D0%BE_%D1%80%D0%B0%D1%81%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D0%B5 (дата обращения 9.01.2026)
4. JupyterNotebook [Электронный ресурс]. https://en.wikipedia.org/wiki/Project_Jupyter (дата обращения 9.01.2026)
5. PyTorch [Электронный ресурс]. https://pytorch.org/. (дата обращения 10.01.2026)
6. Об одной функции потерь для обучения нейросетевых моделей восстановления пропущенных значений многомерных временных рядов // Вестник Южно-Уральского гос. ун-та. Сер. Вычисл. матем. и информ. – 2023. – Т. 12, № 1. – С. 15–25.
7. Кахановский В. Б. Элементы теории нечётких множеств : учеб. пособие / В. Б. Кахановский. – Тамбов : ТГТУ, 2012. – 52 с. (относительное евклидово расстояние в нечётких множествах).