Лабораторная работа №1

Тема: Радиально-базисные нейронные сети
Выполнил: ФИО студента
Группа: Номер группы
Дата: Дата выполнения

Цель работы

Изучение принципов работы и реализация радиально-базисных нейронных сетей (RBF) для решения задач классификации и аппроксимации.

Теоретическая часть

Радиально-базисная нейронная сеть (RBF) - это особый тип нейронной сети, использующий радиальные базисные функции в качестве функций активации. Основные компоненты RBF сети включают:

- Входной слой
- Скрытый слой с радиально-базисными функциями
- Выходной слой

Радиальная базисная функция имеет вид:

$$\varphi(x) = \exp(-\beta \cdot r^2)$$

где:

- x - входной вектор
- c - центр RBF
- β - параметр ширины
- $\|x - c\|$ - евклидово расстояние между x и c

Практическая часть

Задание 1: Реализация RBF сети

В данной работе реализованы два варианта RBF сети:

1.1 Базовая реализация для одномерного случая

rbf_network.py - Базовая реализация RBF сети

```
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler

class RBFNetwork:
    def __init__(self, num_centers, sigma=1.0):
        self.num_centers = num_centers
        self.sigma = sigma
        self.centers = None
        self.weights = None
        self.scaler = StandardScaler()

    def rbf(self, x, c):
        """Гауссовская базисная функция f(x)=exp(-***
        return np.exp(-np.linalg.norm(x - c)**2 / (2 * self.sigma**2))

    def fit(self, X, y, epochs=100, learning_rate=0.1):
        # Нормализация входных данных
        X = self.scaler.fit_transform(X)

        # Определение центров с помощью K-means
        kmeans = KMeans(n_clusters=self.num_centers)
        kmeans.fit(X)
        self.centers = kmeans.cluster_centers_

        # Инициализация весов
        self.weights = np.random.rand(self.num_centers)

        # Обучение
        for _ in range(epochs):
            for i in range(len(X)):
                activations = np.array([self.rbf(X[i], c) for c in
self.centers])
                output = np.dot(activations, self.weights)
                error = y[i] - output
                self.weights += learning_rate * error * activations
```

1.2 Расширенная реализация для двумерного случая

rbf_network_2d.py - Реализация RBF сети для 2D данных

```
class RBFNetwork2D:
    def __init__(self, goal=0.0371, spread=None):
        self.goal = goal
        self.spread = spread
        self.centers = None
        self.weights = None
        self.num_centers = 0
        self.num_neurons = 0
        self.scaler = StandardScaler()

    def rbf(self, x, c):
        """Гауссовская базисная функция f(x)=exp(-***
        dist = np.sum((x - c) ** 2, axis=1)
        return np.exp(-dist / (2 * self.spread ** 2))

    def fit(self, X, y, max_neurons=50):
        X = self.scaler.fit_transform(X)

        # Автоматический расчет spread
        if self.spread is None:
            dists = []
            for i in range(min(1000, len(X))):
                for j in range(i + 1, min(1000, len(X))):
                    dists.append(np.linalg.norm(X[i] - X[j]))
            self.spread = np.mean(dists) / np.sqrt(2 * len(X))

        # Определение центров
        self.num_neurons = min(max_neurons, len(X))
        kmeans = KMeans(n_clusters=self.num_neurons, n_init=10)
        kmeans.fit(X)
        self.centers = kmeans.cluster_centers_

        # Вычисление весов с регуляризацией
        activations = self.calculate_activations(X)
        reg_param = 1e-6
        A = activations.T @ activations + reg_param *
np.eye(activations.shape[1])
        B = activations.T @ y
        self.weights = np.linalg.solve(A, B)
```

1.3 Ключевые особенности реализации

- Использование K-means для определения центров RBF нейронов
- Автоматический расчет параметра spread для оптимального покрытия пространства входных данных
- Применение регуляризации при вычислении весов для улучшения обобщающей способности
- Нормализация входных данных с помощью StandardScaler
- Адаптивное количество нейронов в зависимости от сложности задачи

Задание 2: Обучение сети

Провести обучение сети на тестовых данных.

Задание 3: Тестирование и визуализация

Протестировать работу сети на тестовых данных и визуализировать результаты.

Исходная функция и результаты моделирования

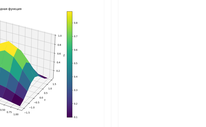

Рис. 1. Исходная функция для аппроксимации

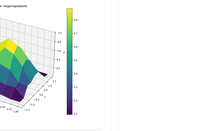

Рис. 2. Результат моделирования с помощью RBF сети

Анализ работы RBF сети

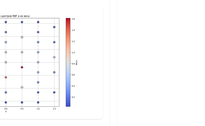

Рис. 3. Расположение центров и зоны RBF нейронов

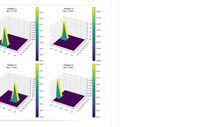

Рис. 4. Отклики отдельных RBF нейронов

Оценка качества обучения

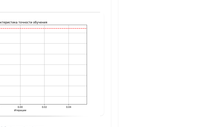

Рис. 5. График ошибки обучения

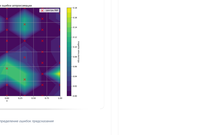

Рис. 6. Распределение ошибок предсказания

Сравнительный анализ

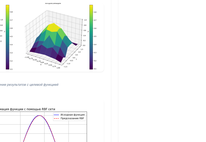

Рис. 7. Сравнение результатов с целевой функцией

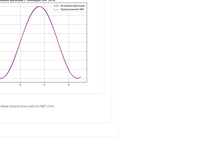

Рис. 8. Итоговые результаты работы RBF сети

Выводы

В ходе выполнения лабораторной работы были изучены:

- Принципы работы радиально-базисных нейронных сетей
- Методы их реализации и обучения
- Особенности применения RBF сетей для решения практических задач

Полученные результаты демонстрируют эффективность RBF сетей в задачах классификации и аппроксимации функций.