Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ Фундаментальные науки

КАФЕДРА Прикладная математика

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

Распознавание графиков решения одномерного линейного уравнения переноса

Студент ФН2-61Б (Группа) _____ (Подпись, дата) О. Д. Климов (И. О. Фамилия)

Руководитель курсовой работы _____ (Подпись, дата) М. П. Галанин (И. О. Фамилия)

2024 г.

Оглавление

Введение . . . 3

1. Постановка задачи . . . 3

1.1. Формулировка . . . 3

1.2. Пример . . . 3

2. Задача Коши для линейного одномерного уравнения переноса . . . 4

3. Численные методы решения задачи . . . 5

3.1. Описание методов . . . 5

3.2. Реализация . . . 6

4. Метод улучшения решения с помощью нейронной сети . . . 6

4.1. Модель сверточной нейронной сети . . . 6

4.2. Распознавание решения . . . 8

4.3. Результаты работы программы . . . 8

5. Актуальность и перспективы задачи . . . 8

Заключение . . . 8

Список использованных источников . . . 9

Введение

Необходимость распознавания графика функций возникает в совершенно разных прикладных задачах науки и техники. Например, она непосредственно связана с проблемой восстановления графика решений уравнений по неточно заданным данных о решениях. В силу нелинейности распознавания изображений нахождение точных алгоритмов для такой задачи испытывало ряд трудностей. Однако с развитием программирования и вычислительной техники стало возможным решать данную задачу методами нейронных сетей.

1. Постановка задачи

1.1. Формулировка

Необходимо изучить методы численного решения линейного одномерного уравнения переноса. Составить и отладить программу для нахождения численного решения задачи Коши для указанного уравнения. Использовать шесть различных разностных схем:

1) Явная схема с левой разностью на двух точках
2) Явная схема с левой разностью на трех точках
3) Неявная схема с левой разностью на двух точках
4) Неявная схема с левой разностью на трех точках
5) Схема Лакса
6) Схема Лакса-Вендрофа

Для всех схем использовать одинаковую систему тестов:

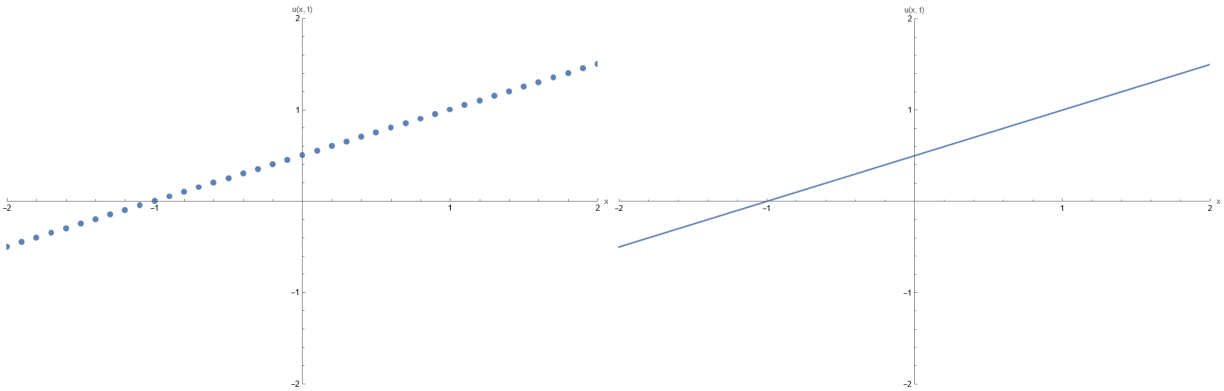
1) Левый треугольник
2) Правый треугольник
3) Прямоугольник
4) Синус
5) "Зуб"

Реализовать модель нейронной сети для распознавания решения на языке программирования Python. На основе модели разработать программу, которая по неточному решению возвращает более точное известное решение.

1.2. Пример

Основная задача работы состоит в том, чтобы создать программу, которая получала бы на вход неточное решение уравнения переноса и выдавала более точное соот-

ветствующее решение. На вход программы подается изображение с численным решением уравнения переноса, которое является неточным (рис. 1a). Программа должна дать на выходе точное решение исходной задачи (рис. 1b).



(a) График неточного решения (b) График точного решения

Рис. 1. Иллюстрация задачи

2. Задача Коши для линейного одномерного уравнения переноса

Уравнение переноса является одним из фундаментальных уравнений математической физики, которое широко используется для описания движения сплошной среды. В то же время оно является простейшим представителем семейства гиперболических уравнений, которые определяются наличием действительных характеристик, число которых совпадает с числом неизвестных для системы уравнений первого порядка. Рассмотрим задачу Коши для уравнения переноса следующего вида:

$$\begin{cases} \dfrac{\partial u}{\partial t} + \dfrac{\partial (au)}{\partial x} = 0, \\ u(x,0) = u_0(x), \end{cases} \quad \text{где } u = u(x,t), \quad a = const > 0, \quad t \in (0,T), \quad x \in (-\infty, +\infty). \tag{1}$$

Приведем аналитическое решение. Уравнение можно записать в виде:

$$u_t + a u_x = 0$$

Запишем и решим характеристическое уравнение:

$$\frac{dt}{1} = \frac{dx}{a} = \frac{du}{0} \quad \Longrightarrow \quad \begin{cases} u = C_1, \\ x = at + C_2 \end{cases} \quad \Longrightarrow \quad \begin{cases} C_1 = u, \\ C_2 = x - at \end{cases} \quad \Longrightarrow \quad u = \psi(x - at).$$

Применим граничные условия:

$$u(x,0) = u_0(x), \quad \Longrightarrow \quad \psi(x) = u_0(x), \quad \Longrightarrow \quad u = u_0(x - at).$$

Получим аналитическое решение уравнения $u = u_0(x - at)$. Решение заключается в сносе неизменного профиля по характеристикам.

Важнейшим свойством рассматриваемого рещения будет являтся сохранение начального профиля: если начальное решение представляет собой, например, профиль буквы M , то оно будет таким всегда.

3. Численные методы решения задачи

Целью построения методов численного решения данного уравнения является не собственно нахождения его решения, а исследование численного алгоритма на простейшем примере. Точное решение данного уравнения известно и тривиально. Однако в чрезвычайно большое количество математических моделей оператор переноса входит в качестве составной части. Это модели газодинамики, гидродинамики, переноса частиц и излучения, электродинамики и многие другие. Разработать для их численного решения численный метод можно только в том случае, если метод будет построен и успешно применен для данного простейшего уравнения, описывающего перенос.

3.1. Описание методов

Обозначим $\gamma = a\frac{\tau}{h}$ — число Куранта. Для численного решения задачи Коши для линейного одномерного уравнения переноса рассмотрим следующие схемы:

1) **Явная схема с левой разностью по двум точкам**:

$$\widehat{y} = (1 - \gamma)y + \gamma y_{-1}.$$

Погрешность аппроксимации схемы: $\psi_h = O(\tau + h)$.

Схема устойчива при $\gamma \leq 1$, причем при $\gamma = 1$ схема является точной.

2) **Неявная схема с левой разностью по двум точкам**:

$$\widehat{y} = \frac{\gamma}{1+\gamma}\widehat{y}_{-1} + \frac{1}{1+\gamma}y.$$

Погрешность аппроксимации схемы: $\psi_h = O(\tau + h)$.

Данная схема является безусловно устойчивой.

3) **Явная схема с левой разностью по трем точкам**:

$$\widehat{y} = (1 - \frac{3}{2}\gamma)y + \gamma(2y_{-1} - \frac{1}{2}y_{-2}).$$

Погрешность аппроксимации схемы: $\psi_h = O(\tau + h^2)$.

Схема является безусловно неустойчивой.

4) **Неявная схема с левой разностью по трем точкам**:

$$\widehat{y} = y - \frac{\gamma}{2}(-3y + 4y_{+1} - y_{+2})$$

5) **Схема Лакса**:

$$\widehat{y} = \frac{(y_{+1} + y_{-1}) - \gamma(y_{+1} - y_{-1})}{2}.$$

Погрешность аппроксимации схемы: $\psi_h = O(\tau + h + \frac{\tau}{h^2})$.

Схема устойчива при $\gamma \leq 1$ и сходится при стремлении $h^2 \to 0$ быстрее, чем τ.

6) **Схема Лакса-Вендрофа**:

$$\widehat{y} = y - \gamma(\frac{(y_{+1} + y) - \gamma(y_{+1} - y)}{2} - \frac{(y + y_{-1}) - \gamma(y - y_{-1})}{2}).$$

Погрешность аппроксимации схемы: $\psi_h = O(\tau^2 + h^2)$.

Схема устойчива при $\gamma \leq 1$, а при $\gamma = 1$ схема является точной.

3.2. Реализация

4. Метод улучшения решения с помощью нейронной сети

Основной целью данной работы является создание программы, которая по известному точному решению сможет улучшить входное неточное. Для реализации такой программы был выбран метод, заключающийся в распознавании изображения нейронной сетью. Алгоритм заключается в следующем: нейронной сети, обученной на исключительно точных решениях, подается на вход неточное, которое она классифицирует. Как результат, нейронная сеть выдает метку соответствующего точного решения, которое и будет возвращать программа.

4.1. Модель сверточной нейронной сети

Сверточная нейронная сеть [1] - модель нейронной сети, предложенная в 1988 году Яном Лекуном, которая и сегодня является одной из самых эффективных для распознавания образов на изображении. Структура сети однонаправленная и для обучения в большинстве случаев используется метод обратного распространения ошибки.

Основной особенностью сверточной нейронной сети является наличие комбинаций слоев свертки и пуллинга. Преимущества модели, заключается в способности улавливать пространственные взаимосвязи на изображении, что позволяет классифицировать одни и те же объекты независимо от их расположения и размера на изображении. Не менее значимым является меньшее количество настраиваемых весов, так как один набор фильтров как весы используется целиком для всего изображения, вместо того, чтобы создавать для каждого пикселя входного изображения свои коэффициенты как это делает перцептрон. Благодаря такой особенности, сверточная нейронная сеть требует меньшее количество времени для обучения. Также данная модель устойчива к повороту и сдвигу распознаваемого изображения.

Для реализации своей модели выберем библиотеку $TensorFlow$ в силу ее простоты и полноты документации. Опишем основные шаги при создании модели сверточной нейросети с помощью библиотеки TensorFlow.

Модель нейросети представляет собой объект класса из TensorFlow, который мы назовем $Model$. При создании модели необходимо указать ее тип как $sequental$, что означает, что процесс вычислений будет последовательно идти по слоям. После создания модели необходимо добавить в нее слои, что можно сделать с помощью метода $.add$. Добавляя нужное нам количество уже встроенных в библиотеку слоев свертки $Conv2D$, указываем количество фильтров, их размерность и функцию активации. Добавляем слои пуллинга $MaxPooling2D$, указывая в параметрах слоя необходимую размерность подвыборки. Последними слоями будут полносвязные слои $Dense$, где мы указываем количество нейронов и также функцию активации. Количество вариантов ответа нейросети обозначим $output$. Наконец, соберем нашу модель с помощью метода $.compile$, в котором необходимо указать параметры функции оптимизатора, потерь и метрику обучения. Для использования модели остается только ее скомпилировать с помощью метода $.compile$.

Библиотека позволяет сохранить параметры скомпилированной модели, что делает возможным встраивать уже созданный алгоритм распознавания в другие программы. Таким образом довольно практично можно создать свою модель нейронной сети для совершенно разных задач.

Листинг 1. Модель нейронной сети

```
1 import tensorflow as tf
2 from tensorflow.keras.models import Sequential
3 from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense
4
5 model = Sequential()
```

```
6
7  model.add(Conv2D(32, (4, 4), activation='relu', input_shape=(512, 512, 1)))
8  model.add(MaxPooling2D(pool_size=(4, 4)))
9
10 model.add(Conv2D(64, (2, 2), activation='relu'))
11 model.add(MaxPooling2D(pool_size=(2, 2)))
12
13 model.add(Flatten())
14 model.add(Dense(output + 50, activation='relu'))
15 model.add(Dense(output, activation='softmax'))
16
17 model.compile(optimizer='adam', loss='sparse_categorical_crossentropy',
       metrics=['accuracy'])
```

4.2. Распознавание решения

4.3. Результаты работы программы

5. Актуальность и перспективы задачи

Заключение

Таким образом в ходе работы были изучены численные методы решения уравнения переноса, проведено тестирование численных методов на 5 тестах, а также созданы несколько программ. Была создана программа, которая по неточному решения уравнения переноса возвращает точное. Была реализована программа на языке программирования C++17 для численного решения уравнения переноса различными схемами, а также реализованы 5 тестов. Реализована модель сверточной нейронной сети для распознавания решений уравнения на языке программирования Python 3.9 с помощью библиотеки машинного обучения TensorFlow 8.0. Изображения графиков решения создавались с помощью математического пакета Wolfram Mathematica 14.

Как итог, реализована программа, которая с помощью модели сети улучшает неточное решение данного уравнения до точного.

Список использованных источников

1. Галанин М.П., Савенков Е.Б. Методы численного анализа математических моделей. М.: Изд-во МГТУ им. Н.Э. Баумана. 2018. 592 с.
2. Tariq Rashid Make Your Own Neural Network // CreateSpace Independent Publishing Platform; 1st edition SAND96-0583 (March 31, 2016)
3. TensorFlow documentation. URL: https://www.tensorflow.org/?hl=ru (Дата обращения 17.01.2024)