

Распознавание текста на основе скелетного представления линий с шириной и сверточных сетей

Петров Виктор Иванович

Московский физико-технический институт

Научный руководитель: Рейер Иван Александрович, к.т.н., ФИЦ ИУ РАН

- Распознавание текста — ключевая задача компьютерного зрения с приложениями в OCR, архивировании, документообороте и других сферах
- Традиционные подходы используют сверточные нейросети, обрабатывающие растровое изображение
- Однако они не используют структурную информацию: линии, скелеты, направление, ширину
- Векторизация и скелетизация — альтернативные способы представить изображение с выделением топологии символов
- Цель работы — исследовать, может ли скелетное представление с шириной улучшить распознавание текста

- Скелетное представление изображения — компактная и структурно насыщенная форма описания формы объектов
- Классические подходы к распознаванию текста используют пиксельные изображения, игнорируя геометрию и топологию символов
- Работы (Blum, 1967), (Ogniewicz Kübler, 1995), (Mestetskiy, 2008) показывают, что скелеты с радиальной функцией позволяют:
 - Сохранять устойчивость к шумам
 - Обеспечивать интерпретируемость через форму и толщину линий
 - Снижать размерность данных
- Скелетизация с шириной (radius function) дает больше информации, чем бинарный скелет — это перспективное направление для задач классификации и распознавания

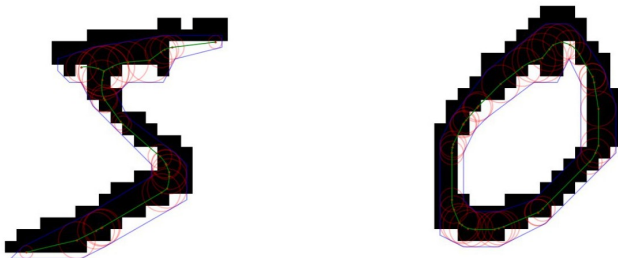
Постановка задачи

- Цель: улучшить распознавание рукописных цифр, используя структурное представление изображения
- Исходные данные:
 - Датасет MNIST
 - Бейзлайн — сверточная сеть LeNet (около 60k параметров)
- Предлагаемый подход:
 - Преобразовать изображения в скелетное представление с сохранением информации о ширине линий
 - Подать такое представление в графовую сверточную сеть
- Задача: сравнить точность моделей при равном (или меньшем) числе параметров и определить, даёт ли скелетизация преимущество

- Представление данных:
 - MNIST → бинаризация → скелетизация (C++)
 - Формирование JSON: узлы (nodes), рёбра (bones), контуры
 - Преобразование в граф: вершины — точки скелета, признаки — координаты и ширина
- Архитектура модели:
 - Графовая нейросеть (GCN) на PyTorch Geometric
 - Сравнение с LeNet по точности и числу параметров
- Технический стек:
 - C++ — быстрая реализация скелетизации и генерации JSON
 - Python (PyTorch, PyTorch Geometric) — обучение графовой модели
 - Вычисления: локальный ПК (CPU / GPU)

- Предобработка данных:
 - Выгрузка и бинаризация датасета MNIST (пороговое округление до 0 / 255)
 - Реализация загрузки изображений в C++
- Работа с библиотекой скелетизации:
 - Диагностика и исправление проблем совместимости старой C++ библиотеки (Местецкий, 2008)
 - Интеграция скелетизации в автоматический пайплайн
- Постобработка и подготовка данных:
 - Преобразование скелета в JSON-структуру (nodes, bones, contours)
 - Реализация визуализации для отладки и анализа качества скелетизации
 - Подбор гиперпараметров (регуляризация, pruning)
- Формирование обучающей выборки для графовой нейросети

Визуализация результата скелетизации



Пример визуализации скелетов для двух цифр

Результаты обучения модели

params: 40138

```
Epoch: 001, Train Loss: 1.2371, Train Acc: 0.5627, Test Acc: 0.8140
Epoch: 002, Train Loss: 0.5095, Train Acc: 0.8358, Test Acc: 0.8654
Epoch: 003, Train Loss: 0.3580, Train Acc: 0.8853, Test Acc: 0.8898
Epoch: 004, Train Loss: 0.2898, Train Acc: 0.9083, Test Acc: 0.9217
Epoch: 005, Train Loss: 0.2446, Train Acc: 0.9235, Test Acc: 0.9385
Epoch: 006, Train Loss: 0.2168, Train Acc: 0.9326, Test Acc: 0.9373
Epoch: 007, Train Loss: 0.2059, Train Acc: 0.9367, Test Acc: 0.9374
Epoch: 008, Train Loss: 0.1870, Train Acc: 0.9423, Test Acc: 0.9527
Epoch: 009, Train Loss: 0.1837, Train Acc: 0.9433, Test Acc: 0.9447
Epoch: 010, Train Loss: 0.1730, Train Acc: 0.9471, Test Acc: 0.9407
Epoch: 011, Train Loss: 0.1660, Train Acc: 0.9488, Test Acc: 0.9318
Epoch: 012, Train Loss: 0.1668, Train Acc: 0.9490, Test Acc: 0.9527
Epoch: 013, Train Loss: 0.1653, Train Acc: 0.9493, Test Acc: 0.9574
Epoch: 014, Train Loss: 0.1537, Train Acc: 0.9535, Test Acc: 0.9536
Epoch: 015, Train Loss: 0.1483, Train Acc: 0.9532, Test Acc: 0.9628
Epoch: 016, Train Loss: 0.1462, Train Acc: 0.9542, Test Acc: 0.9490
Epoch: 017, Train Loss: 0.1454, Train Acc: 0.9538, Test Acc: 0.9578
Epoch: 018, Train Loss: 0.1452, Train Acc: 0.9547, Test Acc: 0.9632
Epoch: 019, Train Loss: 0.1389, Train Acc: 0.9570, Test Acc: 0.9492
Epoch: 020, Train Loss: 0.1409, Train Acc: 0.9566, Test Acc: 0.9523
Best Test Accuracy: 0.9632
```


Выводы и достигнутые результаты

- Реализован полный пайплайн:
 - MNIST → скелетизация (C++) → граф (JSON) → обучение GNN
- Получена модель с **40,138 параметрами**, что в 1.5 раза меньше чем у LeNet
- **Лучшая точность на тестовой выборке: 96.32%**
- Модель показала сопоставимое или лучшее качество при меньшей архитектурной сложности

Перспективы:

- Проверка на более сложных датасетах (EMNIST, рукописные слова)
- Использование более продвинутых GNN (GAT, GraphSAGE)