Распознавание текста на основе скелетного представления толстых линий и сверточных сетей

Литовченко Л.А.

Московский физико-технический институт (Государственный университет)
litovchenko.la@phystech.edu

Аннотация Данная работа посвящена исследованию применения скелетного представления растрового изображения в задаче классификации печатных символов. Сравнивается 3 подхода к решению задачи: классический с использованием свёрточных нейросетей небольшой глубины, с построением скелета изображения и кодировской его в вектор, и их комбинация.

Ключевые слова: сверточные нейронные сети, распознавание текста, скелетное представление, graph embedding

1 Введение

Распознавание растровых изображений уже давно успело стать классической задачей компьютерного зрения. Сюда же относится и распознавание изображений с символами. Каждый год публикуются научные работы, улучшающие и без того высокое качество классификации (1) (2). Традиционным подходом к проектированию архитектур для решения таких задач является использование свёрточных слоев (3) (4). Обычно, входом таких алгоритмов является растровое изображение, однако, существуют подходы, которые производят построение некоторой графовой структуры путёв скелетизации изображения с последующим преобразованием в вектор и обучении на нём.

Скелетизация представляет из себя выделение графа изображенного объекта, форма которого может приближённо описать изначальный объект. В общем случае это можно представить как заполнения внутренностей объектов кругами, центры которых будут являться вершинами скелетного графа и будут соединяться ребрами графа. В литературе можно найти различные статьи, посвященные вопросам скелетизации: (5) описывает генерацию рукописного текста при помощи толстых линий; (6) сравнениет формы изображений при помощи скелетизации; (7) описывает альтернативные подходы к построению графа.

Получаемый после скелетизации граф объекта не позволяет непосредственно использовать его в существующих архитектурах нейронных сетей. В связи с этим необходимо произвести векторизацию графа с помощью его эмбеддинга.

Для проведения экспериментов использовался успевший стать классическим датасет MNIST (8). Он состоит из рукописных цифр, и большое количество работ, в том числе и современных, используют данную выборку для валидации и анализа архитектур (9) (10) (11).

2 Постановка задачи

Введем следующие обозначения:

- I - множество одноканальных изображений символов из набора A. На изображении содержится только один символ, для каждого элемента из I известна метка класса $y \in A$.
  Функция $a : I \to A$ однозначно сопоставляет каждому изображению из I его метку класса y.
- G - множество скелетных представлений изображений, где под скелетным представлением подразумевается неориентированный граф, каждой вершине которого сопоставлено некоторое число, называемое радиусом.
  Функция $g : I \to G$ однозначно сопоставляет каждому изображению из I его скелетное представление по алгоритму, описанному в (5).
- F - множество признаков, описывающих скелетное представление.
  Функция $f : I \to G$ однозначно сопоставляет каждому элементу из G множество его признаков.

Тогда, задачей данной работы, является построение таких функций

$$\hat{a_1} : I \to A$$

$$\hat{a_2} : G \to A$$

$$\hat{a_3} : F \to A$$

, что они минимизируют соответствующие функции потерь:

$$L(\hat{a_1}, a)$$

$$L(\hat{a_2}(g), a)$$

$$L(\hat{a_3}(f(g)), a)$$

, где $L(a, b)$ - функция кросс энтропии двух функций a и b на выборке изображений I.

3 Описание базового алгоритма

В качестве рассматриваемых подходов к решению задачи используются следующие:

– Свёрточная нейронная сеть небольшой глубины, описывающаяся следующим кодом библиотеки keras:

```
Conv2D(32, (3, 3), activation='relu', input_shape=input_shape)
Conv2D(64, (3, 3))
BatchNormalization()
Activation('relu')
MaxPooling2D(pool$_$size=(2, 2))
Dropout(0.3)
Conv2D(128, (3, 3))
BatchNormalization()
Activation('relu')
MaxPooling2D(pool$_$size=(2, 2))
Dropout(0.3)
Flatten()
Dense(256)
Activation('relu')
Dropout(0.5)
Dense(num$_$classes, activation='softmax')
```

– Реализация градиентного бустинга над решающими деревьями, представленная в библиотеке lightgbm.
– Скомбинированная модель, которая принимала на вход картинку и признаки, полученные с помощью градиентного бустинга. Признаки добавлялись перед скрытым слоем после прохождения картинкой свёрточных фильтров и преобразования в одномерный тензор. Нейросеть описывается следующим кодом библиотеки keras:

```
input_shape_cnn = (28, 28, 1)
model_cnn_input = Input(input_shape_cnn)
cnn_layer = Conv2D(32, (3, 3), activation='relu',
input_shape=input_shape_cnn)(model_cnn_input)

cnn_layer = Conv2D(64, (3, 3))(cnn_layer)
cnn_layer = BatchNormalization()(cnn_layer)
cnn_layer = Activation('relu')(cnn_layer)
cnn_layer = Dropout(0.3)(cnn_layer)
cnn_layer = MaxPooling2D(pool_size=(2, 2))(cnn_layer)

cnn_layer = Conv2D(128, (3, 3))(cnn_layer)
cnn_layer = BatchNormalization()(cnn_layer)
cnn_layer = Activation('relu')(cnn_layer)
cnn_layer = Dropout(0.3)(cnn_layer)
cnn_layer = MaxPooling2D(pool_size=(2, 2))(cnn_layer)
cnn_layer = Flatten()(cnn_layer)

input_dim_skelet = (160, )
```

```
features_input = Input(input_dim_skelet)
features_layer1 = Dense(32, input_dim=input_dim_skelet)(features_input)

merged = Concatenate()([cnn_layer, features_layer1])
layer1 = Dropout(0.5)(merged)
output_layer = Dense(10, activation='softmax')(layer1)

model = Model(inputs=[model_cnn_input, features_input], outputs=output_layer)
model.compile(loss="categorical_crossentropy",
optimizer="adam",
metrics=['accuracy', f1])
```

4 Планирование эксперимента

4.1 Описание датасета

Для проведения эксперимента используется выборка рукописных цифр MNIST. В выборке 70000 одноканальных изображений 28x28 каждое, каждому из которых соответствует метка класса. Дополнительно к каждому изображению построен его скелетный граф, методом, описанным в (5). Для того, чтобы построить скелет, картинку необходимо было бинаризовать. В этих данных бинаризация была следующая: все пиксели, цвет которых не черный, становятся белыми. Вершины графа описывались следующими четыремя числами: координатами по оси абсцисс и ординат, степенью вершины и радиальной функцией в этой точке, или же, радиусом вписанной в фигуру окружности в этой точке.

4.2 Свёрточная нейронная сеть

Для обучения нейросети было произведено разбиение выборки на обучение, валидацию и тестирование в соотношении 600000:5000:5000. Веса оптимизировались алгоритмом AdaDelta. Применялись стратегии раннего останова после 5 эпох при отсутствии улучшения точности на валидации и сохранения лучшей модели по этому же признаку.

4.3 Градиентный бустинг

При использовании градиентного бустинга, из скелетного графа извлекались следующие признаки:

- количество вершин в скелетном представлении
- среднее значение среди всех чисел, как представленных в описании графа, так и по каждой категории
- сумма как всех чисел, представленных в описании графа, так и по каждой категории

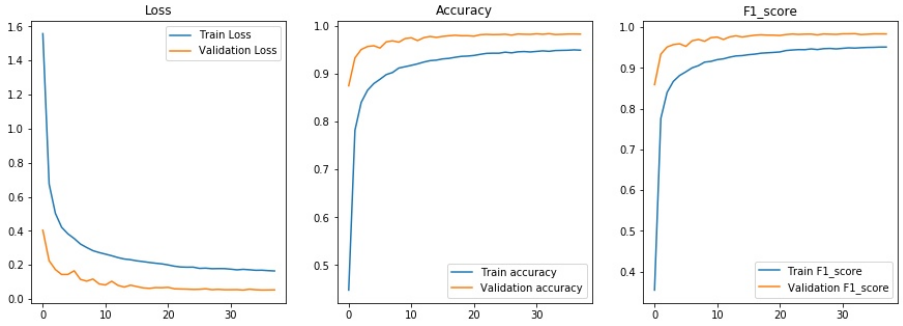
– дисперсия как среди всех чисел, представленных в описании графа, так и по каждой категории
– разность между максимальным и минимальным числом как среди всех чисел, представленных в описании графа, так и по каждой категории
– перцентили от 0.1 до 0.9 с шагом 0.1 как среди всех чисел, представленных в описании графа, так и по каждой категории
– для каждой из четырёх категорий самое частое значение и сколько раз оно встретилось

Для обучения и предсказания использовалась модель LGBMClassifier со следующими параметрами:

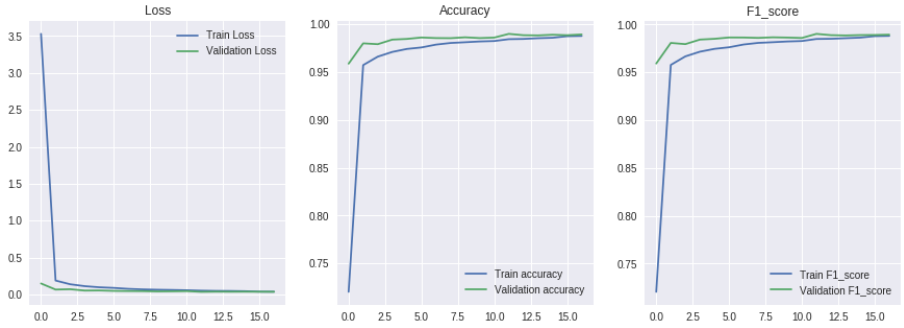
```
LGBMClassifier(boosting_type='gbdt', num_leaves=31, max_depth=-1,
learning_rate=0.1, n_estimators=100, subsample_for_bin=200000,
objective=None, class_weight=None, min_split_gain=0.0,
min_child_weight=0.001, min_child_samples=20, subsample=1.0,
subsample_freq=1, colsample_bytree=1.0, reg_alpha=0.0,
reg_lambda=0.0, random_state=None, n_jobs=-1, silent=True)
```

5 Анализ ошибки

В данном разделе приведены графики функции потери, а так же точности и F1 метрики на валидации в зависимости от эпохи обучения
Просто свёрточная нейронная сеть:



Комбинированный эксперимент:



6 Анализ структуры модели

6.1 Нейронная сеть

В ходе анализа структуры было выяснено, что добавление блоков нормализации по батчам значительно ускорило сходимость. Путем частичного перебора была подобрана итоговая конфигурация количества фильтров свёрточных слоёв.

В итоге имеем следующую струтуру нейронной сети:

Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 26, 26, 32)	320
conv2d_8 (Conv2D)	(None, 24, 24, 64)	18496
batch_normalization_5 (Batch	(None, 24, 24, 64)	256
activation_7 (Activation)	(None, 24, 24, 64)	0
max_pooling2d_5 (MaxPooling2	(None, 12, 12, 64)	0
dropout_7 (Dropout)	(None, 12, 12, 64)	0
conv2d_9 (Conv2D)	(None, 10, 10, 128)	73856
batch_normalization_6 (Batch	(None, 10, 10, 128)	512
activation_8 (Activation)	(None, 10, 10, 128)	0
max_pooling2d_6 (MaxPooling2	(None, 5, 5, 128)	0
dropout_8 (Dropout)	(None, 5, 5, 128)	0
flatten_3 (Flatten)	(None, 3200)	0
dense_5 (Dense)	(None, 256)	819456
activation_9 (Activation)	(None, 256)	0
dropout_9 (Dropout)	(None, 256)	0
dense_6 (Dense)	(None, 10)	2570

Total params: 915,466
Trainable params: 915,082
Non-trainable params: 384

6.2 Градиентный бустинг

Были проведены эксперименты с измененем параметров n_estimators и learning_rate, однако они не улучшели результат на тестовых данных, в связи с чем было решено оставить значения по умолчанию 100 и 0.1.

6.3 Комбинированная модель

Итоговая структура комбинированной нейронной сети:

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 28, 28, 1)	0	
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320	input_1[0][0]
conv2d_2 (Conv2D)	(None, 24, 24, 64)	18496	conv2d_1[0][0]
batch_normalization_1 (BatchNor	(None, 24, 24, 64)	256	conv2d_2[0][0]
activation_1 (Activation)	(None, 24, 24, 64)	0	batch_normalization_1[0][0]
dropout_1 (Dropout)	(None, 24, 24, 64)	0	activation_1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 12, 12, 64)	0	dropout_1[0][0]
conv2d_3 (Conv2D)	(None, 10, 10, 128)	73856	max_pooling2d_1[0][0]
batch_normalization_2 (BatchNor	(None, 10, 10, 128)	512	conv2d_3[0][0]
activation_2 (Activation)	(None, 10, 10, 128)	0	batch_normalization_2[0][0]
dropout_2 (Dropout)	(None, 10, 10, 128)	0	activation_2[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 5, 5, 128)	0	dropout_2[0][0]
input_2 (InputLayer)	(None, 160)	0	
flatten_1 (Flatten)	(None, 3200)	0	max_pooling2d_2[0][0]
dense_1 (Dense)	(None, 32)	5152	input_2[0][0]
concatenate_1 (Concatenate)	(None, 3232)	0	flatten_1[0][0] dense_1[0][0]
dropout_3 (Dropout)	(None, 3232)	0	concatenate_1[0][0]
dense_2 (Dense)	(None, 10)	32330	dropout_3[0][0]

Total params: 130,922
Trainable params: 130,538
Non-trainable params: 384

7 Выбор модели

В результате проведенных экспериментов были получены следующие результаты:

Модель	Точность на тесте	F1-score на тесте
Нейросеть:	**0.9938**	**0.9938**
Градиентный бустинг:	**0.9314**	**0.9304**
Комбинированный метод:	**0.9901**	**0.9901**

8 Результаты работы

В результате проведённого эксперимента были сравнены три подхода к классификации рукописных символов: классический с использованием свёрточных нейросетей небольшой глубины, с построением скелета изображения

и кодировской его в вектор, и их комбинация. Классический нейросетевой подход показал себя лучше прочих, достигнув ошибки на тестовых данных в 0.62 процента, в то время как комбинированный метод и градиентный бустинг оказались в 1.5 и в 11 раз хуже.

Литература

[1] Zou, Xianli Fast Convergent Capsule Network with Applications in MNIST— Advances in Neural Networks – ISNN 2018— Springer International Publishing— pp. 3–10

[2] Palvanov A., Im Cho Y Comparisons of deep learning algorithms for MNIST in real-time environment— International Journal of Fuzzy Logic and Intelligent Systems— 2018. – Т. 18. – №. 2. – С. 126-134.

[3] LeCun Y. et al. Convolutional networks for images, speech, and time series //The handbook of brain theory and neural networks. – 1995. – Т. 3361. – №. 10. – С. 1995.

[4] Ciresan D. C. et al. Convolutional neural network committees for handwritten character classification //Document Analysis and Recognition (ICDAR), 2011 International Conference on. – IEEE, 2011. – С. 1135-1139.

[5] Клименко С. В., Местецкий Л. М., Семенов А. Б. Моделирование рукописного шрифта с помощью жирных линий //Труды. – 2006. – Т. 16.

[6] Кушнир О. и др. Сравнение формы бинарных растровых изображений на основе скелетизации //Машинное обучение и анализ данных. – 2012. – Т. 1. – №. 3. – С. 255-263.

[7] Масалович А., Местецкий Л. Распрямление текстовых строк на основе непрерывного гранично-скелетного представления изображений //Труды Международной конференции «Графикон», Новосибирск.–2006.–4 с.

[8] LeCun Y., Cortes C., Burges C. J. MNIST handwritten digit database // Available: http://yann. lecun. com/exdb/mnist. – 2010. – Т. 2.

[9] Zhu D. et al. Negative Log Likelihood Ratio Loss for Deep Neural Network Classification //arXiv preprint arXiv:1804.10690. – 2018.

[10] Nair P., Doshi R., Keselj S. Pushing the limits of capsule networks //Technical note. – 2018.

[11] Hsieh P. C., Chen C. P. Multi-task Learning on MNIST Image Datasets. – 2018.