АЛГОРИТМЫ ПОНИМАНИЯ ТЕКСТА МЕТОДАМИ ГЛУБОКОГО ОБУЧЕНИЯ НЕЙРОННЫХ СЕТЕЙ

Н.А. Кривошеев
Научный руководитель – В.Г. Спицын
Томский политехнический университет
nikola0212@mail.ru

Аннотация

В данной статье рассматривается задача классификации текстов с помощью многослойного персептрона и сверточной нейронной сети. Рассмотрена предобработка текстовых данных в виде посимвольного преобразования текста. Приведены результаты обучения и тестирования нейронных сетей различных топологий на тестовой выборке fetch_20newsgroups [1]. Все программы реализованы на языке Python, с использованием библиотеки keras.

Введение

На данный момент одной из наиболее популярных задач, является понимание текста. К данной задаче относятся: классификация, перевод, ответы на вопросы и др. В данной статье будет рассмотрена задача классификации текста методами глубокого обучения нейронных сетей. Задача классификации является одной из традиционных в машинном обучении, в связи с чем существуют тренировочные данные для обучения нейронных сетей. Существует множество решений данной задачи [3, 4, 5]. Далее будут рассмотрены возможные решения задачи классификации текстов, с приведением результатов тестирования на тестовых выборках fetch_20newsgroups [1].

Предобработка данных

Первичная предобработка заключается в замене больших букв маленькими, и в удалении малоинформативных и редких символов. К малоинформативным символам были отнесены символы: табуляция, перевод строки, длинная последовательность одинаковых символов (например, множество из трех и более звездочек) и др. К редким символам относятся символы, используемые во всей выборке не более нескольких десятков раз. В результате очистки текста в данной работе используется алфавит из 65 символов.

Следующим этапом предобработки является преобразование в вектора или числа (в случае использования слоя keras Embedding). Существуют различные методы предобработки текстовых данных [3, 5], в данной работе использовалось посимвольное преобразование текста в вектора.

Посимвольное преобразование текста позволяет преобразовать текст в вектор для нейронной сети с наименьшими затратами времени. Каждый символ заменяется на соответствующий ему вектор, для этого не требуется производить дополнительных вычислений.

В данной работе использовался слой Embedding, осуществляющий преобразование символов в вектора автоматически. Перед подачей данных на слой, необходимо заменить символы соответствующими им числами.

Во время предобработки или после, все тексты стандартизируется (обрезаются или заполняются) до заданной длины. В данной работе все тексты стандартизировались до длины текста в 2 000 символов.

Результаты обучения и тестирования нейронных сетей

Далее будут рассмотрены результаты тестирования нейронных сетей двух различных топологий:

- Многослойный персептрон прямого распространения;
- Сверточная нейронная сеть (как с использованием, так и без использования слоев Pooling);

Все топологии нейронных сетей, приведенные в таблицах ниже, обучались с помощью метода NADAM [2], с использованием категориальной функции потерь (categorical_crossentropy). Во всех скрытых слоях нейронной сети используется функция активации RELU. Выходной слой использует функцию активации softmax.

Программный код многослойного персептрона, где N количество нейронов, a tipe количество классов приведен в Таблица 1:

Таблица 1. Программный код многослойного персептрона на языке Python.

```
model = Sequential()
# Слой для векторного представления слов
model.add(Embedding(65, 64, input_length=2000))
model.add(SpatialDropout1D(0.1))
# Полносвязный слой
model.add(Flatten())
model.add(Dropout(0.25))
model.add(Dense(N, activation="relu"))
model.add(Dense(tipe, activation="softmax"))
# Копмилируем модель
model.compile(loss='categorical_crossentropy',
        optimizer='nadam',
        metrics=['accuracy'])
```

Программный код сверточной нейронной сети, где N и M количество нейронов, K ширина окна в символах, L ширина окна в символах слоя Pooling, a tipe количество классов приведен в Таблице 2.

Таблица 2. Программный код сверточной нейронной сети на языке Python.

```
model = Sequential()
# Слой для векторного представления слов
model.add(Embedding(65, 64, input_length=2000))
model.add(SpatialDropout1D(0.1))
# Слой сверточной нейронной сети
model.add(Conv1D(M, K))
model.add(MaxPooling1D(L))
# Полносвязный слой
model.add(Flatten())
model.add(Dropout(0.25))
model.add(Dense(N, activation="relu"))
model.add(Dense(tipe, activation="softmax"))
# Копмилируем модель
model.compile(loss='categorical_crossentropy',
        optimizer='nadam',
        metrics=['accuracy'])
```

Было проведено тестирование нейронных сетей на тестовой выборке fetch_20newsgroups [1], с использованием посимвольного преобразования текста. Использовались нейронные сети следующих топологий:

- Многослойный персептрон:
  1. N=25;
  2. N=50;
  3. N=100.
- Сверточная нейронная сеть:
  1. N=50, M=25, K=5, L=50;
  2. N=100, M=50, K=5, L=50;
  3. N=50, M=25, K=5, L=3.

Точность распознавания в процентах на тестовой выборке приведена в Таблице 3. Данные результаты были получены на основе не более трех тестов для каждой топологии. В результате точность нейронных сетей с топологиями, в которых меньше нейронов, может быть выше чем у нейронных сетей с большим количеством нейронов. Количество примеров в классе может сильно отличается от среднего количества примеров, часть из которых сильно различаются по смыслу. В результате это могло повлиять на точность распознавания третьей топологии сверточной нейронной сети. Классы из выборки для тестирования брались в алфавитном порядке. Список классов выборки: 'alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.windows.x', 'misc.forsale', 'rec.autos', 'rec.motorcycles', 'rec.sport.baseball'.

Таблица 3. Результаты тестирования нейронных сетей.

Топология нейронной сети		3 класса	10 классов	20 классов
Многослойный персептрон	1	51%	20.5%	5%
	2	50%	21%	9%
	3	52%	23%	16%
Сверточная нейронная сеть	1	81%	56%	50%
	2	81.25%	53%	52%
	3	57.5%	10%	14%

В результате тестирования на тестовом множестве fetch_20newsgroups [1] многослойный персептрон оказался не эффективен, а сверточная нейронная сеть показала наивысшую точность. По результатам тестирования следует заметить, что при увеличении количества используемых классов, точность распознавания значительно уменьшается.

Заключение

На основе полученных результатов можно сделать вывод, что использование сверточных нейронных сетей с широким окном пулинга (50 символов и более), значительно повышает точность классификации.

Работа поддержана грантом РФФИ № 18-08-00977 А.

Список использованных источников:

1. sklearn.datasets.fetch_20newsgroups [Электронный ресурс]. — Режим доступа: URL: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html (22.11.2018)
2. An overview of gradient descent optimization algorithms [Электронный ресурс]. — Режим доступа: URL: http://ruder.io/optimizing-gradient-descent/index.html#nadam (22.11.2018)
3. Общий взгляд на машинное обучение: классификация текста с помощью нейронных сетей и TensorFlow [Электронный ресурс]. — Режим доступа: URL: https://tproger.ru/translations/text-classification-tensorflow-neural-networks/ (21.11.2018)
4. Классификация предложений с помощью нейронных сетей без предварительной обработки [Электронный ресурс]. — Режим доступа: URL: https://habr.com/company/meanotek/blog/256593/ (21.11.2018)
5. Классификация текста с помощью нейронной сети на Java [Электронный ресурс]. — Режим доступа: URL: https://habr.com/post/332078/ (21.11.2018)