Обработка естественного языка, лекция 4: введение в ИНС и полносвязные сети

Пожидаев М. С.

1 декабря 2021 г.

Перцептрон

В 1960 году Фрэнк Розенблат представил нейрокомпьютер Mark I, который являлся аппаратной реализацией перцептрона. Представленный компьютер:

- был способен запоминать визуальные образы и далее распознавать их;
- наследовал идею нейросвязей в человеческом мозге, расширяя пространство значений нейронов от булевого до долей единицы;
- содержал сенсорные элементы, ассоциативные элементы и сумматор, которые на современном языке являются полносвязной сетью с одним скрытым слоем.

Полносвязная сеть

Полносвязная или линейная ячейка нейронной сети выражается следующим уравнением:

$$y = act(Wx + b)$$

- W — матрица весов (*kernel*);
- b — вектор сдвига (*bias*);
- $act()$ — функция активации.

Функция активации используется для отображения значений в интервал от нуля до единицы.

Обучение ИНС

Вперёд!

В процессе обучения разработчик предоставляет обучающие данные для векторов x и y. Нейросеть, подсчитав ошибку, проводит подстройку матрицы весов и вектора сдвига для её уменьшения.

Получилось ли?

Требуется достижение сходимости, отсутствие которой означает несоответствие ёмкости нейросети сложности обучающих данных.

Повторим!

Обучение проходит в несколько эпох с одними и теми же обучающими данными. Это повышает качество работы.

Примеры функций активации

Гиперболический тангенс:

$$th(x) = \frac{\exp(\frac{x}{\alpha}) - \exp(-\frac{x}{\alpha})}{\exp(\frac{x}{\alpha}) + \exp(-\frac{x}{\alpha})}$$

SoftMax может использоваться для целого вектора сразу:

$$\sigma(x)_i = \frac{\exp(x_i)}{\sum_{k=1}^{K} \exp(x_k)}$$

Способы представления слов

При работе с нейросетями следует подобрать способ представления слов в виде векторов. В простейшем случае можно составить словарь слов и использовать вектор с нулями, кроме элемента, соответствующему индексу слова, который содержит единицу (*one-hot-vector*).

Недостатки:

1. Все слова должны быть известны.
2. Над векторами невозможно производить математические операции.
3. Вектор получается слишком длинным, а это приводит к увеличению матриц весов.

Word embedding

По-новому!

Значительно более перспективной является идея представления слов в виде вершин в k-мерном векторном пространстве, в котором k часто равно 300, 500 или т. д.

Плюсы:

1. Появляется возможность производить алгебраические операции.
2. Размерность пространства существенно меньше, чем при использовании индексов в словаре.

Пример!

"Жизнь - любовь = быт" (серьёзно, результат на основе НКРЯ).

Word2vec

Word2vec — распространённый способ получения слов в векторном представлении (*word embeddings*). Вектора отражают лексический контекст слова.

Существует две формы работы:

- **Continuous Bag-of-Words:** угадать слово по контексту;
- **Skip-Gram:** предсказать контекст по слову.

Word2vec представляет собой мелкую (*shallow*) полносвязную сеть со скрытым слоем, размерность которого равна размерности пространства векторов. Матрица весов скрытого слоя и содержит в себе искомые векторные представления.

TensorFlow и Keras

TensorFlow — библиотека для алгоритмов машинного обучения от Google с основным интерфейсом разработки на Python.

1. Реализует вычислительную модель в виде ориентированного графа, по которому идут вычисления и определяется ошибка.
2. Поддерживает аппаратное ускорение на графических процессорах.
3. Имеет пополняемую коллекцию готовых моделей.
4. Предлагает Keras — упрощённый интерфейс определения моделей.

Пример работы с Keras

```
model = Sequential()
model.add(LSTM(100, activation='relu',\
  input_shape=(n_steps_in, n_features)))
model.add(RepeatVector(n_steps_out))
model.add(LSTM(100, activation='relu',\
  return_sequences=True))
model.add(TimeDistributed(Dense(1)))
model.compile(optimizer='adam', loss='mse')
```

PyTorch

PyTorch — библиотека для алгоритмов машинного обучения от Facebook с основным интерфейсом разработки на Python.

1. Поддерживает autograd — автоматический подсчёт градиента обратного спуска для обучения.
2. Поддерживает аппаратное ускорение на графических процессорах.

Пример работы с PyTorch

```
model = torch.nn.Sequential(
    torch.nn.Linear(3, 1),
    torch.nn.Flatten(0, 1)
)
loss_fn = torch.nn.MSELoss(reduction='sum')
learning_rate = 1e-6
loss = loss_fn(y_pred, y)
if t % 100 == 99:
    print(t, loss.item())
```

Спасибо за внимание!

Веб-сайт: https://marigostra.ru/
E-mail: msp@luwrain.org