Основы обработки текстов

Лекция #2:
Разметка последовательности, нейронные сети

Лектор: м.н.с. ИСП РАН Андрианов Иван Алексеевич

План лекции

- Нейронные сети прямого распространения
- Рекуррентные нейронные сети
- Границы между сущностями
- Разметка последовательности
- Условные случайные поля
- Алгоритм Витерби

Распознавание именованных сущностей (NER)

- На входе: текст, разбитый на предложения и токены
- На выходе: множество сущностей (начало, конец, тип)

Александр Пушкин родился в Москве, столице России

личность город страна

Microsoft — один из крупнейших производителей ПО в мире

компания

Обработка текстов — область на стыке ИИ и лингвистики

научная дисциплина НД НД

Общая схема решения NER

Александр Пушкин погиб в результате дуэли с Дантесом

- Рассматриваются только метки «О» и «личность»

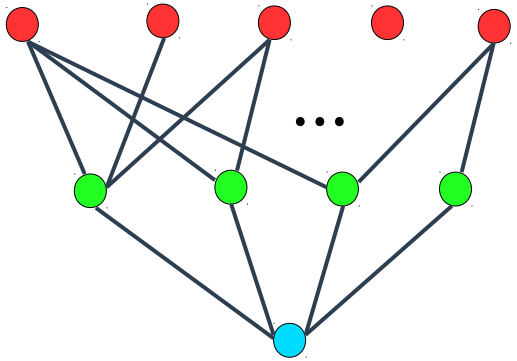
(<s>, <s>, Александр, Пушкин, погиб)	окно токенов (n=2)
(1, 1, 4, 2, 3)	окно индексов
(1,0,0,0|1,0,0,0|0,0,0,1|0,1,0,0|0,0,1,0)	one-hot признаки (x)
p(y=1|x;w;b) = σ((w, x) + b)	вероятность «лич»

Нейронные сети прямого распространения

- Логистическая регрессия $y = \sigma((w,x) + b)$ может работать только с линейно разделимыми классами
- Нейронная сеть прямого распространения — совокупность линейных и нелинейных преобразований

$\tanh(t) = (1 — e^{-2t}) / (1 + e^{-2t})$

$\sigma(t) = 1 / (1 + e^{-t})$, $\mathrm{softmax}(t)[i] = e^{t[i]} / \sum_j e^{t[j]}$



x

$z = \tanh(W_1 x + b_1)$

$y = \sigma(W_2 z + b_2)$

Обучение нейронных сетей

- Сеть: $z = \tanh(W_1 x + b_1)$, $y = \sigma(W_2 z + b_2)$
- Функция потерь «кросс-энтропия»

  $l = -\hat{y} \cdot \log y - (1 - \hat{y}) \cdot \log(1 - y)$
- Дифференцирование сложной функции (chain rule)

  $d\sigma/dt = \sigma(t) \cdot (1 - \sigma(t))$

  $d(\tanh)/dt = 1 - \tanh^2(t)$

  $dl/dW_2 = dl/dy \cdot dy/dW_2 = (-\hat{y} / y + (1 - \hat{y}) / (1 - y)) \cdot y \cdot (1 - y) \cdot z$

  $dl/dW_1 = dl/dy \cdot dy/dW_1 = dl/dy \cdot dy/dz \cdot dz/dW_1 =$
  $(-\hat{y} / y + (1 - \hat{y}) / (1 - y)) \cdot y \cdot (1 - y) \cdot W_2 \cdot (1 - z^2) \cdot x$

Обучение нейронных сетей

- Сеть: $z = \tanh(W_1 x + b_1)$, $y = \sigma(W_2 z + b_2)$
- Функция потерь: $l = f(y, \hat{y})$
- Обратное распространение ошибки (backpropagation)

  $\Delta W_1 = -\rho \cdot dl/dW_1$

  $\Delta W_2 = -\rho \cdot dl/dW_2$
- ρ – скорость обучения (learning rate)

  $10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$
- Низкий ρ – недообучение, высокий ρ – переобучение

Обучение нейронных сетей

- Mini-batch — множество примеров малого размера (2-32)
- Обновление параметров на основе потерь на одном примере приводит к нестабильному обучению
- Вместо этого можно усреднять потери на примерах из одного mini-batch
- Можно ли взять все данные в качестве batch? Если бы они представляли собой генеральную совокупность, это было бы оптимальным решением

Обучение нейронных сетей

- Стохастический градиентный спуск (stochastic gradient descent, SGD): $\Delta W = -\rho \cdot dl/dW$ нестабилен, т.к. не учитывает предыдущие изменения параметров
- Momentum (обычный или nesterov) сглаживает обновления, используя «историю»

  $\Delta W = -\rho \cdot A_W$ при $A_W = \mu \cdot A_W + dl/dW$, $\mu \sim 0.9$
- Адаптивные оптимизаторы (RMSProp, Adadelta, Adagrad, Adam, Nadam) динамически подстраивают ρ под каждый из параметров

Обучение нейронных сетей

- Для многих задач обработки текста ручная разметка — трудоемкий процесс, поэтому выборки не слишком велики
- Обучение на несколько эпох:

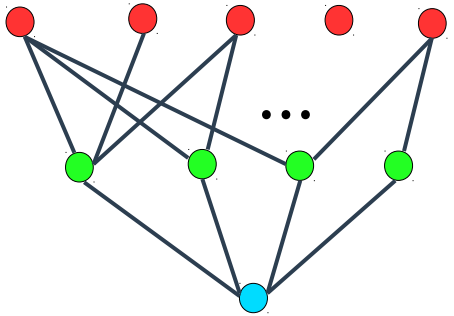
  $[(x_1,\hat{y}_1), \ldots, (x_N, \hat{y}_N)], [(x_1,\hat{y}_1), \ldots, (x_N, \hat{y}_N)], [(x_1,\hat{y}_1), \ldots, (x_N, \hat{y}_N)]$
- Рекомендация: ρ должен «затухать» на каждой эпохе

  $\rho = \rho_0 / (1 + \omega t)$

  $\rho = \rho_0 \cdot \omega^t$

Обучение нейронных сетей

- В отличие от линейных классификаторов нейронные сети способны строить нелинейные функции со сложными взаимодействиями признаков
- Однако на малых выборках легко переобучиться из-за переусложнения взаимодействий
- Dropout — метод регуляризации нейронных сетей: с вероятностью p заменяем значение нейрона на 0



x

$z = \tanh(W_1 x + b_1)$

$y = \sigma(W_2 z + b_2)$

Длинные зависимости в задаче NER

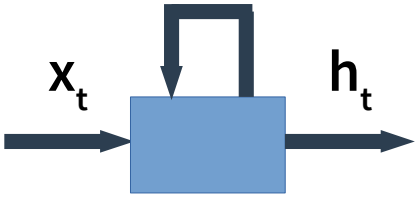
Капица после встречи с Хрущёвым вернулся на …

- Малого окна токенов недостаточно
- Увеличение размеров окна ведет к попаданию «шумных» слов, которые портят результаты
- Требуется более гибкая модель, способная анализировать контекст как полноценную последовательность

Рекуррентные нейронные сети (RNN)

- Моделируют последовательность $x_1, x_2, \ldots, x_T$

  $h_0 = 0, h_t = \sigma(W_x x_t + W_h h_{t-1} + b)$



- «Развертывание» сети

  $h_3 = \sigma(W_x x_3 + W_h \cdot \sigma(W_x x_2 + W_h \cdot \sigma(W_x x_1 + 0 + b) + b) + b)$

- Проблема угасающих градиентов (vanishing gradients)

  $dh_3/dW_x = h_3 \cdot (1 - h_3) \cdot (x_3 + W_h \cdot h_2 \cdot (1 - h_2) \cdot (x_2 + W_h \cdot h_1 \cdot (1 - h_1) \cdot x_1))$

  $dh_3/dW_h = h_3 \cdot (1 - h_3) \cdot (h_2 + W_h \cdot h_2 \cdot (1 - h_2) \cdot h_1)$

Длинная краткосрочная память (LSTM)

- Ячейка памяти, которая борется с угасанием градиентов

$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f)$ что забываем?

$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i)$ что запоминаем?

$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o)$ что выдаем?

$c_t = f_t * c_{t-1} + i_t * \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c)$ ячейка

$h_t = o_t * \tanh(c_t)$ выход

Двусторонние рекуррентные сети (BiLSTM)

- В обработке текстов важны и левый, и правый контексты

Александр Пушкин погиб в результате дуэли с Дантесом

- Можно применить два слоя LSTM:
  - по словам слева направо: кодировщик левого контекста
  - по словам справа налево: кодировщик правого контекста
- Далее их результаты объединяются или усредняются

Инструменты для реализации нейронных сетей

- Tensorflow (Python, *Java*)
- Theano (Python)
- Keras (Python)
- PyTorch (Python)
- Caffe (Python)
- DyNet (Python)
- Deeplearning4j (Java)

Границы между сущностями

Замеченный Алексеем Дмитрий Петров спешил на работу

O	лич	лич	лич	O	O	O

- Достаточно ли таких меток для определения границ сущностей?
- Требуется более сложная структура меток

BIO	O	B-лич	B-лич	I-лич	O	O	O
BIO2	O	I-лич	B-лич	I-лич	O	O	O
BILOU	O	U-лич	B-лич	L-лич	O	O	O

Разметка последовательности

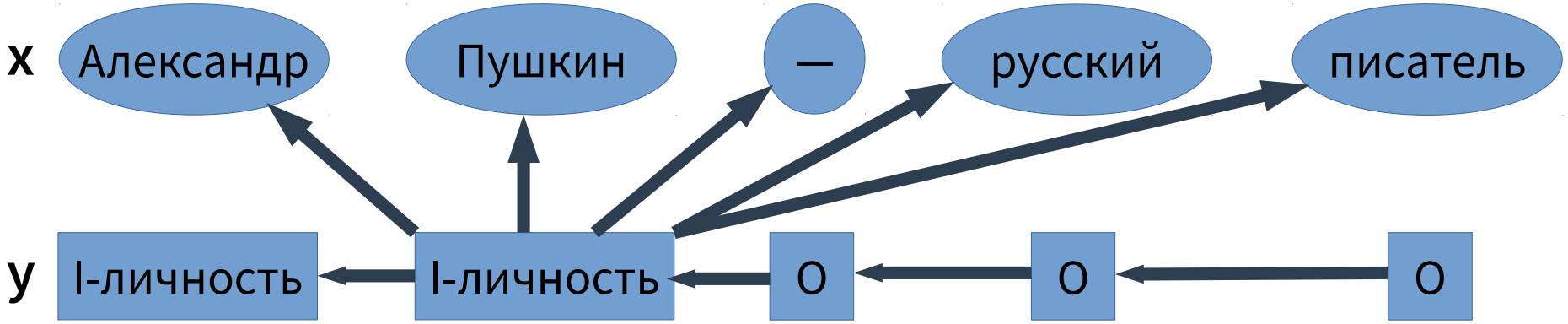
Сент-Китс и Невис — государство в Карибском море

страна геогр. объект

- При независимой классификации токенов мы не учитываем корреляции между их метками:
  - После «страна» наиболее вероятны «страна» и «O»
  - В BIO после «B-страна» наиболее вероятны «I-страна» и «O»
  - В BIO после «I-геогр.объект» невозможна «I-страна»
- Разметкой последовательности называется модель, присваивающая метки классов для всех элементов последовательности совместно

Условные случайные поля (CRF)

- Графическая модель
- В обработке текстов частный случай: linear-chain CRF



$p(y|x) = e^{score(y|x)} / \Sigma_{y'} e^{score(y'|x)}$ – аналог softmax

$score(y|x) = \Sigma_i \Sigma_j \lambda_j f_j(x, i, y_i, y_{i-1})$

Условные случайные поля (CRF)

- Типовой случай: для каждого слова есть вектор признаков x_i, построенный вручную или с помощью нейронной сети

  $\text{score}(y|x) = \Sigma_i \Sigma_j \lambda_j f_j(x, i, y_i, y_{i-1}) = \Sigma_i (\text{score}(y_i|x_i) + p(y_i|y_{i-1}))$

  $\text{score}(y_i|x_i) = (Wx_i + b)[y_i]$

- $p(y_i|y_{i-1})$ оценивается по обучающей выборке как частота следования одной метки за другой, например:
  - В BIO по любой выборке $p(y_i = \text{I-личность}|y_{i-1} = \text{I-город}) = 0$
  - В BIO2 по любой выборке $p(y_i = \text{B-личность}|y_{i-1} = \text{I-город}) = 0$

Алгоритм Витерби

- Определяет наиболее вероятную последовательность y на основе $p(y_i|x_i)$ и $p(y_i|y_{i-1})$ без перебора

  p(I|I)=0.3, p(O|I)=0.7, p(I|O)=0.4, p(O|O)=0.6

Александр	I=0.7, O=0.3	I=0.7, O=0.3
Пушкин	I=0.6, O=0.4	II=0.126, IO=0.196
—	I=0.2, O=0.8	IOI=0.016, IOO=0.094
русский	I=0.3, O=0.7	IOII=0.014, IOOO=0.039
писатель	I=0.4, O=0.6	IOOOI=0.00624, IOOOO=0.01404

Следующая лекция

- Синонимия: дистрибутивные векторные представления слов
- Лектор: Майоров Владимир Дмитриевич