Структурированное обучение в задаче разметки последовательности.

Попов Артём, OzonMasters, осень 2022

Natural Language Processing

Постановка задачи разметки последовательности

Дано множество размеченных последовательностей (x, y):

- $x = (x_1, \dots, x_n)$ – входная последовательность (слова)
- $y = (y_1, \dots, y_n)$ – выходная последовательность (метки, теги)

Необходимо по входной последовательности предсказать элементы выходной последовательности.

1. Метка y_i соответствует слову x_i. Длины x, y из одной пары совпадают, но могут различаться с длинами других пар.
2. Две последовательности можно привести к одной длине дополнив короткую специальным <PAD> токеном.

Другие названия: sequence tagging, sequence labeling

Составные сущности. BIO-нотация.

Именованная сущность может состоять из нескольких токенов. В этом случае обычно используют BIO-нотацию:

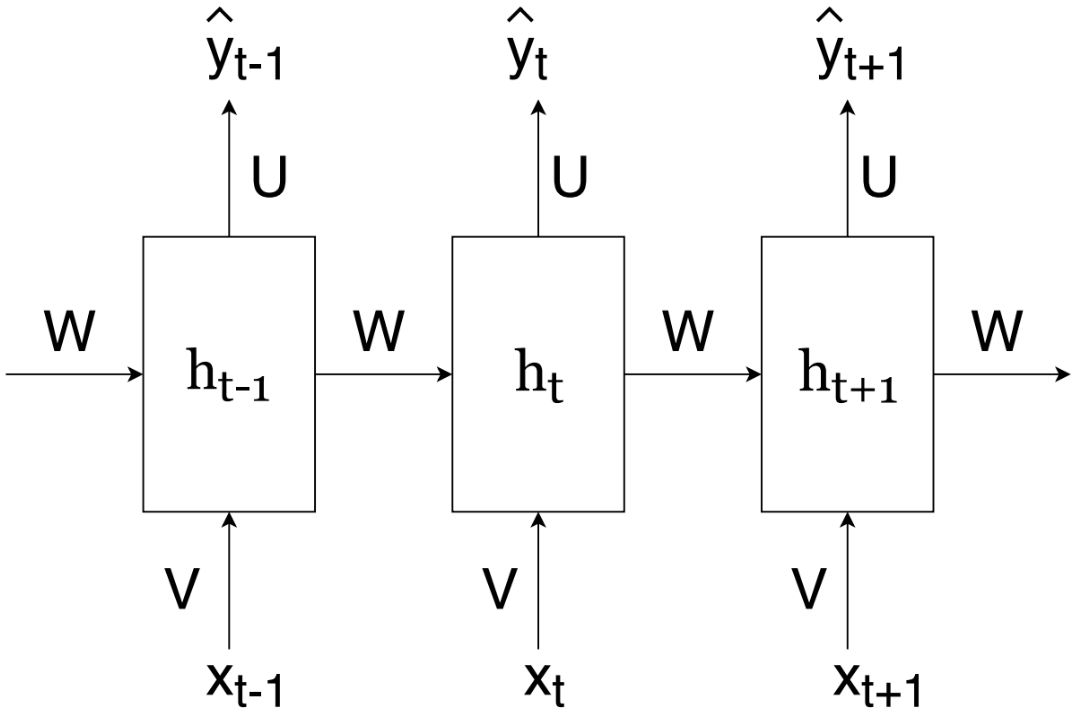
- B (Begin) – первое слово сущности
- I (Inside) – второе слово сущности
- O (Outside) – слово не входит ни в какую сущность

Betty	came	to	Los	Angeles	to	become	an	actress
B-PER	O	O	B-LOC	I-LOC	O	O	O	O

Подходы к задаче разметке

- Rule-based подход
- Классификатор на каждой позиции, использующий признаки контекста позиции
- Графические модели (HMM / MEMM / **CRF**)
- Нейронные сети (рекуррентные**,** трансфомеры, свёрточные)
- **Комбинация нейронных сетей и графических моделей**

Модель рекуррентной нейронной сети (RNN)



h_t – скрытое состояние сети в момент времени t

Принцип работы сети:

$$h_t = f(Vx_t + Wh_{t-1} + b)$$
$$\hat{y}_t = g\left(Uh_t + \hat{b}\right)$$

Обучение сети (backpropagation through time):

$$\sum_{t=1}^{n} \mathcal{L}(y_t, \hat{y}_t) \to \min_{V,W,U,b,\hat{b}}$$

[Rumelhart et al. Learning internal representations by error propagation (1985)](#)

Теггер на основе RNN

Этап 1. Вычисление вероятностей меток

$$h_t = GRU(h_{t-1,} x_t)$$
$$\hat{y}_t = softmax(Uh_t + \hat{b})$$

Этап 2. Вычисление меток из множества классов Y:

$$y_t = \arg\max_Y \hat{y}_t$$

Есть ли в этой схеме какие проблемы?

Нет никакой связи между предсказаниями соседних элементов.

Примеры. Возможные ошибки моделей

Некоторые ошибки исходят из несовершенности модели:

Давай	встретимся	на	библиотеке	имени	Ленина
O	O	O	B-LOC	B-LOC	I-LOC
O	O	O	B-LOC	I-LOC	B-PER

Некоторые ошибки прям противоречат постановке задачи:

Давай	встретимся	на	библиотеке	имени	Ленина
O	O	O	B-LOC	I-LOC	I-PER
O	O	O	I-LOC	I-LOC	I-PER

Напоминание. Мультиномиальная регрессия

$x \in X$ – объект, $y \in Y$ – метка класса

Вычисление вероятности класса ($\theta \in \mathbb{R}^{|Y| \times J}$):

$$a(x, y) = \sum_{j=1}^{J} \theta_{yj} f_j(x), \qquad p(y|x) = softmax_{y \in Y}\left(\{a(x, y)\}_{y \in Y}\right)$$

Вычисление метки класса: $\hat{y} = \arg\max_{y \in Y} p(y|x)$

Обучение модели:

$$\sum_{(x,y) \in D} \log p(y|x) \to \max_{\theta}$$

Модель линейного CRF (linear-CRF)

$x = (x_1, \dots, x_n), \qquad x_i \in X, \qquad y = (y_1, \dots, y_n), \qquad y_i \in Y$

Вычисление вероятности последовательности ($\theta \in \mathbb{R}^J$):

$$a(x, y) = \sum_{j=1}^{J} \theta_j F_j(x, y), \qquad \mathrm{p(y|x)} = softmax_{y \in Y^n}\left(\{a(x, y)\}_{y \in Y}\right)$$

Вычисление метки класса: $\hat{y} = \arg\max_{y \in Y^n} p(y|x)$

Обучение модели:

$$\sum_{(x,y) \in D} \log p(y|x) \to \max_{\theta}$$

Вычислительные проблемы модели

$x = (x_1, \dots, x_n), \qquad x_i \in X, \qquad y = (y_1, \dots, y_n), \qquad y_i \in Y$

Вычисление вероятности последовательности ($\theta \in \mathbb{R}^J$):

$$a(x, y) = \sum_{j=1}^{J} \theta_j F_j(x, y), \qquad \mathrm{p(y|x)} = softmax_{y \in Y^n}\left(\{a(x, y)\}_{y \in Y}\right)$$

Вычисление метки класса: $\hat{y} = \arg\max_{y \in Y^n} p(y|x)$

Обучение модели:

$$\sum_{(x,y) \in D} \log p(y|x) \to \max_{\theta}$$

Специальный вид признаков F

$$F_j(x, y) = \sum_{i=1}^{n} f_j(y_i,\ y_{i-1}, x, i)$$

$f_j(y_i, y_{i-1}, x, i)$ – информация о последовательности x, полезная для предсказания метки y_i в позиции i, когда в предыдущей позиции $(i-1)$ находится метка y_{i-1}

- $f_j(\dots, i)$ может зависеть от всего x, необязательно от x_i
- $f_j(\dots, i)$ не может зависеть от других меток кроме y_i и y_{i-1}
- последовательности (x, y) могут иметь любую длину, но количество признаков всегда равно J

Классические примеры индикаторных признаков

y_i = ADVERB и слово x_i оканчивается на «-ly»

Ожидаем $\theta_j > 0$, такие слова часто оказываются наречиями

y_i = PRONOUN и i = 1 и предложение оканчивается знаком «?»
Ожидаем $\theta_j > 0$, первое слово в вопросительных предложениях часто оказывается местоимением

y_i = NOUN и y_{i-1} = ADJECTIVE

Ожидаем $\theta_j > 0$, существительные часто следуют за прилагательным

y_i= PREPOSITION и y_{i-1}= PREPOSITION

Ожидаем $\theta_j < 0$, два предлога редко встречаются подряд

Нужно решить две задачи:

1. получение оптимальной разметки по обученной модели

2. вычисление градиента для обучения модели

Получение оптимальной разметки: алгоритм Витерби

Функция Φ – потенциал

$$\hat{y} = \arg\max_{y \in Y^n} p(y|x) = \arg\max_{y \in Y^n} a(x, y)$$

$$a(x, y) = \sum_{j=1}^{J} \theta_j F_j(x, y) = \sum_{j=1}^{J} \theta_j \sum_{i=1}^{n} f_j(y_i,\ y_{i-1}, x, i) \ =$$

$$= \sum_{j=1}^{J} \sum_{i=1}^{n} \theta_j f_j(y_i,\ y_{i-1}, x, i) \ = \sum_{i=1}^{n} \sum_{j=1}^{J} \theta_j f_j(y_i,\ y_{i-1}, x, i) \ =$$

$$= \sum_{i=1}^{n} \Phi_{\mathrm{i}}(y_i, y_{i-1}) = \sum_{i=1}^{n-1} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_n(y_n, y_{n-1})$$

Функция Q – промежуточный максимум

Функция $Q(i, v)$ зависит от двух аргументов:

- $i \in \{0, \dots, n\}$ – позиция в последовательности
- $v \in Y$ – последняя метка в промежуточном максимуме

$$Q(0, v) = 0, \qquad Q(k, v) = \max_{y_1, \dots, y_{k-1}} \left(\sum_{i=1}^{k-1} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_k(v, y_{k-1}) \right)$$

Для первых элементов:

$Q(1, v) = \Phi_1(v, y_0)$, $y_0 = < START >$

$$Q(2, v) = \max_{y_1}\big(\Phi_1(y_1, y_0), + \Phi_2(v, y_1)\big)$$

$$Q(3, v) = \max_{y_1, y_2}\big(\Phi_1(y_1, y_0) + \Phi_2(y_2, y_1) + \Phi_3(v, y_2)\big)$$

Функция Q – промежуточный максимум

Благодаря марковскому свойству признаков, Q может вычисляться итерационно.

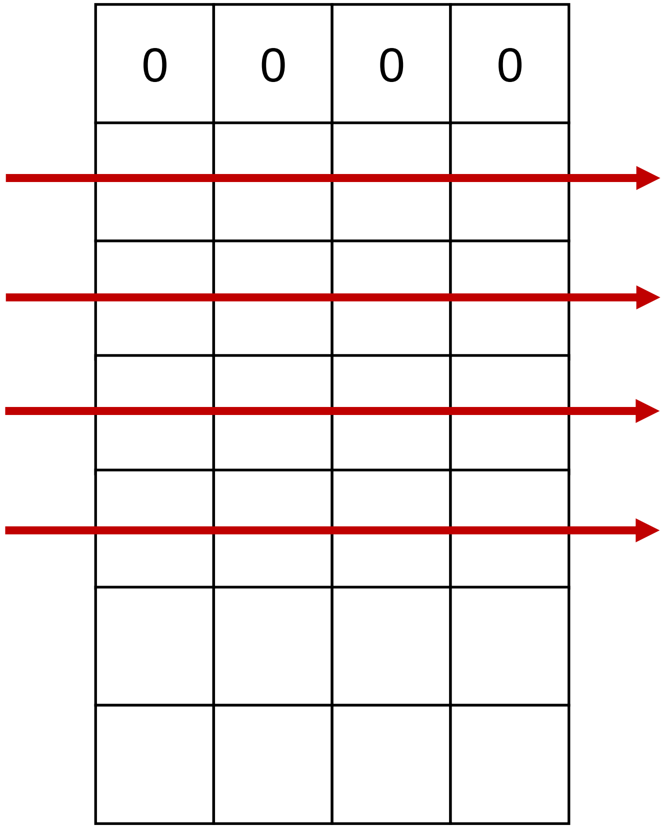
Хотим вывести рекуррентную формулу зависимости $Q(k, v)$ от элементов предыдущей строки $Q(k-1, u)$.

Нас интересует значение $Q(n, v)$:

$$Q(n, v) = \max_{y \in Y^{n-1}} a(x, y_{1:n-1} + [v])$$

$$\max_{y \in Y^n} a(x, y) = \max_{v} Q(n, v)$$

$v \in Y$



$i \in \{0, \dots, n\}$

Рекуррентный пересчёт функции Q

$$Q(k,v) = \max_{y_1,\ldots,y_{k-1}} \left(\sum_{i=1}^{k-1} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_k(v, y_{k-1}) \right) =$$

$$= \max_{y_1,\ldots,y_{k-1}} \left(\sum_{i=1}^{k-2} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_{k-1}(y_{k-1}, y_{k-2}) + \ \Phi_k(v, y_{k-1}) \right) =$$

$$= \max_{y_{k-1}} \max_{y_1,\ldots,y_{k-1}} \left(\sum_{i=1}^{k-2} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_{k-1}(y_{k-1}, y_{k-2}) + \ \Phi_k(v, y_{k-1}) \right) =$$

$$= \max_{y_{k-1}} \big(Q(k-1, y_{k-1}) + \Phi_k(v, y_{k-1}) \big)$$

Нахождение $\widehat{y}$

$$\hat{y}_n = \arg\max_{y_n} \max_{y_1,\dots,y_{n-1}} a(x,y) = \arg\max_{y_n} Q(n, y_n)$$

$$\hat{y}_k = \arg\max_{y_k} \max_{y_1,\dots,y_{k-1}} \max_{y_{k+1},\dots,y_n} a(x,y) =$$

$$= \arg\max_{y_k} \max_{y_1,\dots,y_{k-1}} \left(\sum_{i=1}^{k} \Phi_{\mathrm{i}}(y_i, y_{i-1}) + \Phi_{k+1}(\hat{y}_{k+1}, y_k) + \sum_{i=k+2}^{n} \Phi_{\mathrm{i}}(\hat{y}_i, \hat{y}_{i-1}) \right) =$$

$$= \arg\max_{y_k} \big(Q(k, y_k) + \Phi_{k+1}(\hat{y}_{k+1}, y_k) \big)$$

Алгоритм Витерби (оптимальная разметка)

Прямой ход – рекуррентное вычисление матрицы $Q \in \mathbb{R}^{n \times |Y|}$

$$Q(0, v) = 0$$

$$Q(k, v) = \max_{y_{k-1} \in Y} \big(Q(k-1, y_{k-1}) + \Phi_k(v, y_{k-1}) \big)$$

Обратный ход – вычисление оптимальной разметки $\hat{y} \in Y^n$

$$\hat{y}_n = \arg\max_{y_n} Q(n, y_n)$$

$$\hat{y}_k = \arg\max_{y_k} \big(Q(k, y_k) + \Phi_{k+1}(\hat{y}_{k+1}, y_k) \big)$$

Какая вычислительная сложность алгоритма? Прямой $O\big(n|Y|^2\tilde{J}\big)$, обратный $O\big(n|Y|^2\big)$, $\tilde{J}$ – количество ненулевых признаков для x.

Вычисление градиента функционала

$$\frac{\partial}{\partial \theta_j} log\, p_\theta\,(y|x) = \frac{\partial}{\partial \theta_j}\log\frac{\exp(\langle\theta, F(x,y)\rangle)}{Z(\theta)} = F_j(x,y) - \frac{\partial}{\partial \theta_j} log\, Z\,(\theta) =$$

$$= F_j(x,y) - \frac{1}{Z(\theta)}\frac{\partial}{\partial \theta_j}\sum_{y\in Y^n} exp\left(\sum_{j=1}^{J}\theta_j F_j(x,y)\right) =$$

$$= F_j(x,y) - \frac{1}{Z(\theta)}\left(\sum_{y\in Y^n} exp\left(\sum_{j=1}^{J}\theta_j F_j(x,y)\right)F_j(x,y)\right) =$$

$$= F_j(x,y) - \sum_{y\in Y^n} F_j(x,y) p_\theta(y|x)$$

Вычисление градиента функционала

Подставим в градиент выражение F_j через f_j:

$$\sum_{y \in Y^n} F_j(x, y) p_\theta(y|x) = \sum_{y \in Y^n} p_\theta(y|x) \sum_{i=1}^{n} f_j(y_i, y_{i-1}, x, i) =$$

$$= \sum_{i=1}^{n} \sum_{y_i \in Y} \sum_{y_{i-1} \in Y} p_\theta(y_{i-1}, y_i|x) f_j(y_i, y_{i-1}, x, i)$$

Мы избавимся от экспоненциальной сложности, если научимся эффективно вычислять $p_\theta(y_{i-1}, y_i|x)$.

Вычисление совместной вероятности двух меток

Разделим выражение внутри суммы на три множителя, первый и третий будут содержать элементы из одного суммирования:

$$p_\theta(y_{k-1},\, y_k|x) \;= \frac{1}{Z(\theta)} \sum_{y_1,\,\dots,\,y_{k-2}} \; \sum_{y_{k+1},\,\dots,\,y_n} exp\left(\sum_{i=1}^{n} \Phi_k(y_i,\, y_{i-1}) \right) =$$

$$= \frac{1}{Z(\theta)} \sum_{y_1,\,\dots,\,y_{k-2}} \; \sum_{y_{k+1},\,\dots,\,y_n} exp\left(\sum_{i=1}^{k-2} \Phi_i(y_i,\, y_{i-1}) \; + \; \Phi_{k-1}(y_{k-1},\, y_{k-2}) \right) \times$$

$$\times\, exp\big(\Phi_k(y_k, y_{k-1})\big) \times exp\left(\Phi_{k+1}(y_{k+1},\, y_k) + \sum_{i=k+2}^{n} \Phi_i(y_i,\, y_{i-1}) \right) =$$

Вычисление совместной вероятности двух меток

$$= \frac{\exp(\Phi_k(y_k, y_{k-1}))}{Z(\theta)} \times$$

$$\times \sum_{y_1,\,\dots,\,y_{k-2}} exp\left(\sum_{i=1}^{k-2} \Phi_i(y_i,\, y_{i-1}) \ + \ \Phi_{k-1}(y_{k-1},\, y_{k-2})\right) \times$$

$$\times \sum_{y_{k+1},\dots,y_n} exp\left(\Phi_{k+1}(y_{k+1},\, y_k) + \sum_{i=k+2}^{n} \Phi_i(y_i,\, y_{i-1})\right) =$$

$$= \frac{\exp(\Phi_k(y_k, y_{k-1}))}{Z(\theta)} \times \alpha(k-1, y_{k-1}) \times \beta(k, y_k)$$

Вектора “вперёд” и “назад” (forward & backward)

$\alpha(k, v)$ – “вперёд” вектор, ненормированная вероятность начала последовательности:

$$\alpha(k, v) = \sum_{y_1, \ldots, y_{k-1}} exp\left(\sum_{i=1}^{k-1} \Phi_i(y_i,\ y_{i-1})\ +\ \Phi_k(v,\ y_{k-1})\right)$$

$\beta(k, u)$ – “назад” вектор, ненормированная вероятность конца последовательности:

$$\beta(k, u) = \sum_{y_{k+1}, \ldots, y_n} exp\left(\Phi_{k+1}(y_{k+1},\ u) + \sum_{i=k+2}^{n} \Phi_i(y_i,\ y_{i-1})\right)$$

Пересчёт векторов «вперёд»

Для векторов “вперёд” и “назад” можно вывести рекуррентные формулы аналогичные формулам в алгоритме Витерби:

$$\alpha(k, v) = \sum_{y_1, \dots, y_{k-1}} exp\left(\sum_{i=1}^{k-1} \Phi_i(y_i,\, y_{i-1}) \;+\; \Phi_k(v,\, y_{k-1})\right) =$$

$$= \sum_{y_1, \dots, y_{k-2}} \sum_{y_{k-1}} exp\left(\sum_{i=1}^{k-2} \Phi_i(y_i,\, y_{i-1}) \;+\; \Phi_{k-1}(y_{k-1},\, y_{k-2})\right) \times$$

$$\times \exp\big(\Phi_k(v, y_{k-1})\big) = \sum_{y_{k-1}} \alpha(k-1, y_{k-1}) \exp\big(\Phi_k(v, y_{k-1})\big)$$

Пересчёт векторов “вперёд”

Пересчёт “вперёд” векторов:

$$\alpha(0, v) = \mathbb{I}[v = < START >]$$

$$\alpha(k, v) = \sum_{u \in Y} \alpha(k-1, u) \exp(\Phi_k(v, u))$$

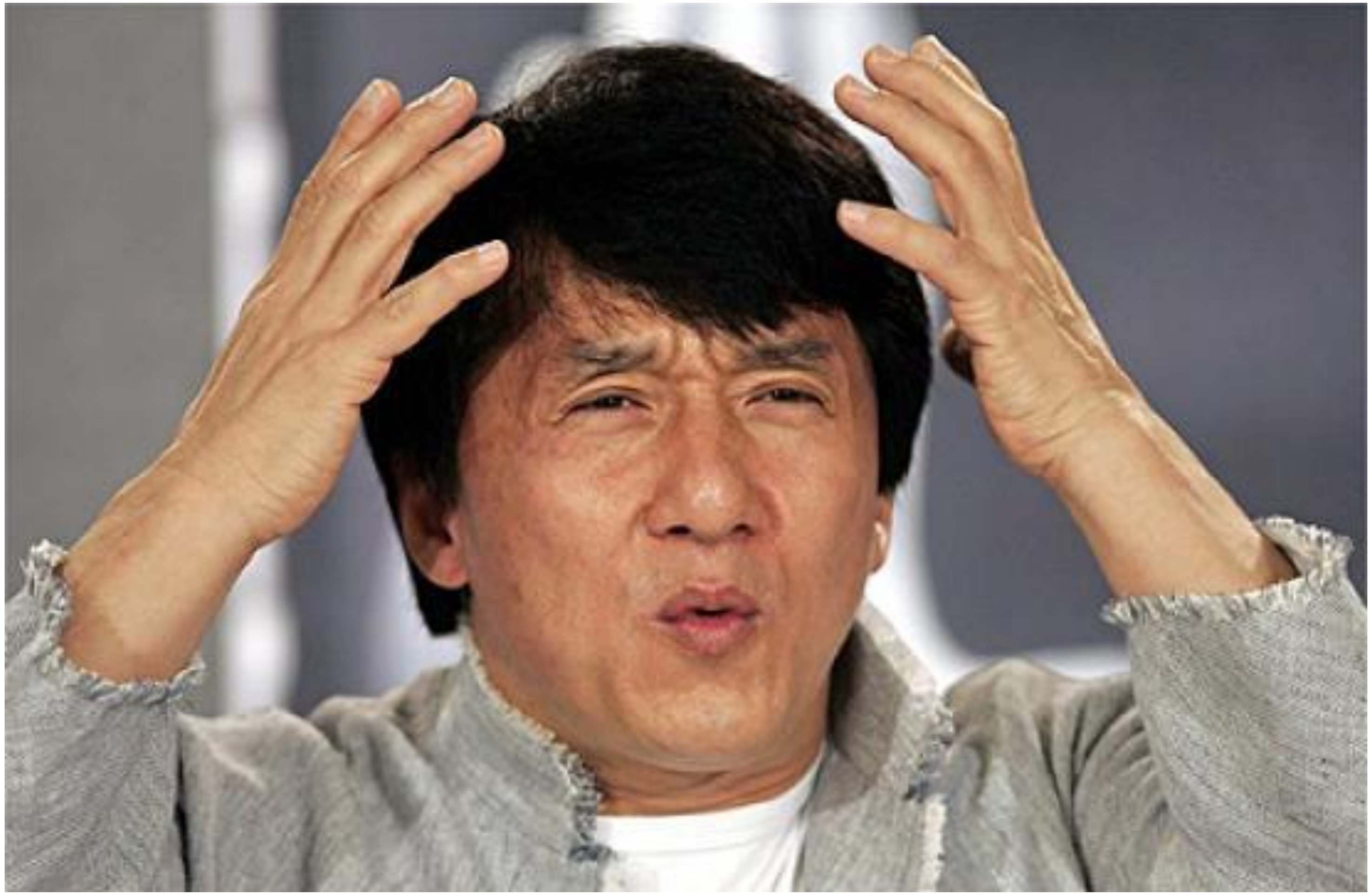
Пересчёт “назад” векторов

$$\beta(n+1, u) = \mathbb{I}[u = < END >]$$

$$\beta(k, u) = \sum_{v \in Y} \beta(k+1, v) \exp(\Phi_{k+1}(v, u))$$

В конце и начале последовательности стоят специальные метки.

Осталось собрать всё вместе…



Алгоритм вперёд-назад (forward-backward)

$$\frac{\partial}{\partial \theta_j} \log p_\theta (y|x) = F_j(x, y) - \sum_{i=1}^{n} \sum_{y_i \in Y} \sum_{y_{i-1} \in Y} p_\theta(y_{i-1}, y_i|x) f_j(y_i, y_{i-1}, x, i)$$

$$p_\theta(y_{i-1}, y_i|x) = \frac{\exp(\Phi_k(y_k, y_{k-1}))}{Z(\theta)} \times \alpha(k-1, y_{k-1}) \times \beta(k, y_k)$$

$$\mathrm{Z}(\theta) = \sum_{v \in Y} \alpha(n, v)$$

Какая вычислительная сложность алгоритма? $O(n|Y|^2\tilde{J})$

Резюме по linear-CRF

- Linear-CRF – аналог мультиномиальной регрессии для последовательности
- Коэффициенты Linear-CRF можно обучить при помощи SGD
- Градиент модели вычисляется при помощи алгоритма вперёд-назад за $O(n|Y|^2\tilde{J})$
- Процедура получения оптимальной разметки готовой моделью производится при помощи алгоритма Витерби за $O\left(n|Y|^2\tilde{J}\right)$
- В настоящий момент редко используется сама по себе, но часто работает в комбинации с нейросетевыми моделями

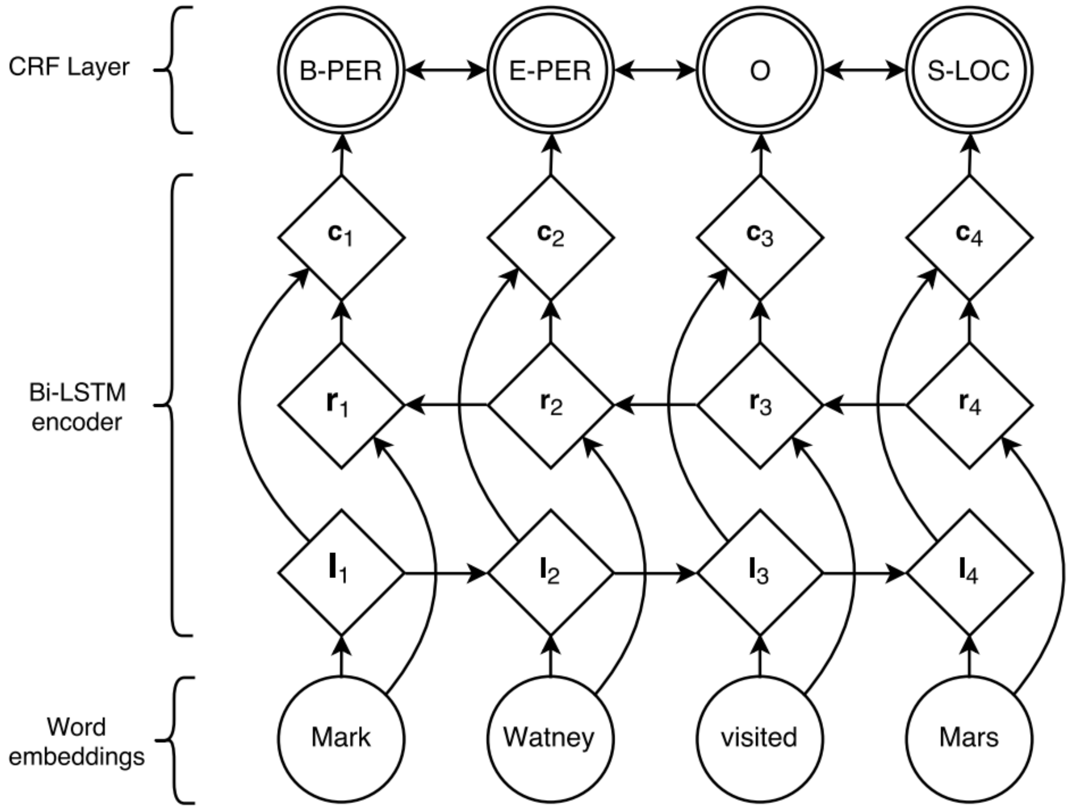
Linear-CRF как слой

Linear-CRF – дифференцируемая функция вычисления вероятности последовательности и может использоваться как последний слой в нейросети.

Зачем это может быть надо?

- Согласование предсказаний для соседних меток
- Улучшение качества модели (если мало обучающих данных)

Linear-CRF как слой



- CRF используется в конце сети вместо softmax
- на вход CRF поступает последовательность логитов $(o_1, \ldots, o_n), o_i \in \mathbb{R}^{|Y|}$
- можно интерпретировать CRF как обучаемый постпроцессинг модели

Признаки CRF для постобработки выходов сети

Выход нейросети, соответствующий метке v, на i-ом элементе последовательности будем обозначать $o_i(v)$

$|Y| \times |Y|$ признаков для связи с нейросетевой моделью

$$f_{uv}(y_i, y_{i-1}, x, i) = \mathbb{I}[y_i = u]\mathbb{I}[y_{i-1} = v]o_i(u)$$

$|Y| \times |Y|$ бинарных признаков для связи меток друг с другом

$$\phi_{uv}(y_i, y_{i-1}, x, i) = \mathbb{I}[y_i = u]\mathbb{I}[y_{i-1} = v]q(u, v),$$

$q(u, v)$ – любая попарная статистика меток u и v или 1

Преобразование выражения для признаков

Обозначим линейные коэффициенты $W(u,v)$ и $\Theta(u,v)$.

$$a(x,y) = \sum_{i=1}^{n} \Phi_{\mathrm{x,i}}(y_i, y_{i-1})$$

$$\Phi_{\mathrm{x,i}}(y_i, y_{i-1}) = \left(\sum_{u \in Y} \sum_{v \in Y} W(u,v) f_{uv}(\dots) + \Theta(u,v)\phi_{uv}(\dots) \right) =$$

$$= \sum_{u \in Y} \sum_{v \in Y} \mathbb{I}[y_i = u]\mathbb{I}[y_{i-1} = v](W(u,v)o_i(u) + \Theta(u,v)q(u,v)) =$$

$$= W(y_i, y_{i-1})o_i(y_i) + \Theta(y_i, y_{i-1})q(y_i, y_{i-1}) = \{W \text{ положим равным } 1\} =$$

$$= o_i(y_i) + \Theta(y_i, y_{i-1})q(y_i, y_{i-1})$$

Улучшение от CRF в biLSTM

Добавление CRF слоя улучшает результаты:

Table 2: Comparison of tagging performance on POS, chunking and NER tasks for various models.

		POS	CoNLL2000	CoNLL2003
Random	Conv-CRF (Collobert et al., 2011)	96.37	90.33	81.47
	LSTM	97.10	92.88	79.82
	BI-LSTM	97.30	93.64	81.11
	CRF	97.30	93.69	83.02
	LSTM-CRF	**97.45**	93.80	84.10
	BI-LSTM-CRF	97.43	**94.13**	**84.26**
Senna	Conv-CRF (Collobert et al., 2011)	97.29	94.32	88.67 (89.59)
	LSTM	97.29	92.99	83.74
	BI-LSTM	97.40	93.92	85.17
	CRF	97.45	93.83	86.13
	LSTM-CRF	97.54	94.27	88.36
	BI-LSTM-CRF	**97.55**	**94.46**	**88.83** (**90.10**)

Huang et al (2015); Bidirectional LSTM-CRF Models for Sequence Tagging

Добавление CRF слоя на практике

- Если у вас мало обучающих данных, может сильно помочь
- Есть реализация в Pytorch (ссылка)
- Сильно замедляет модель при большом количестве уникальных меток
- Можно не обучать коэффициенты модели
- Можно использовать более простые эвристики
- Можно использовать более сложные методы структурированного обучения

И ещё немного про практику…

Архитектура NER в Spacy

Tok2Vec слой – конкатенация нескольких представлений:

- Два хэша слова ($emb_1(hash1(w))\ +\ emb_2(hash2(w))$)
- Форма слова (shape)
  word -> xxx, SotA -> XxxX, Y2k -> Xdx
- Префикс и суффикс слова
- Лемма слова

К последовательности таких эмбеддингов применяется CNN или трансформер.

Как обучался теггер Natasha

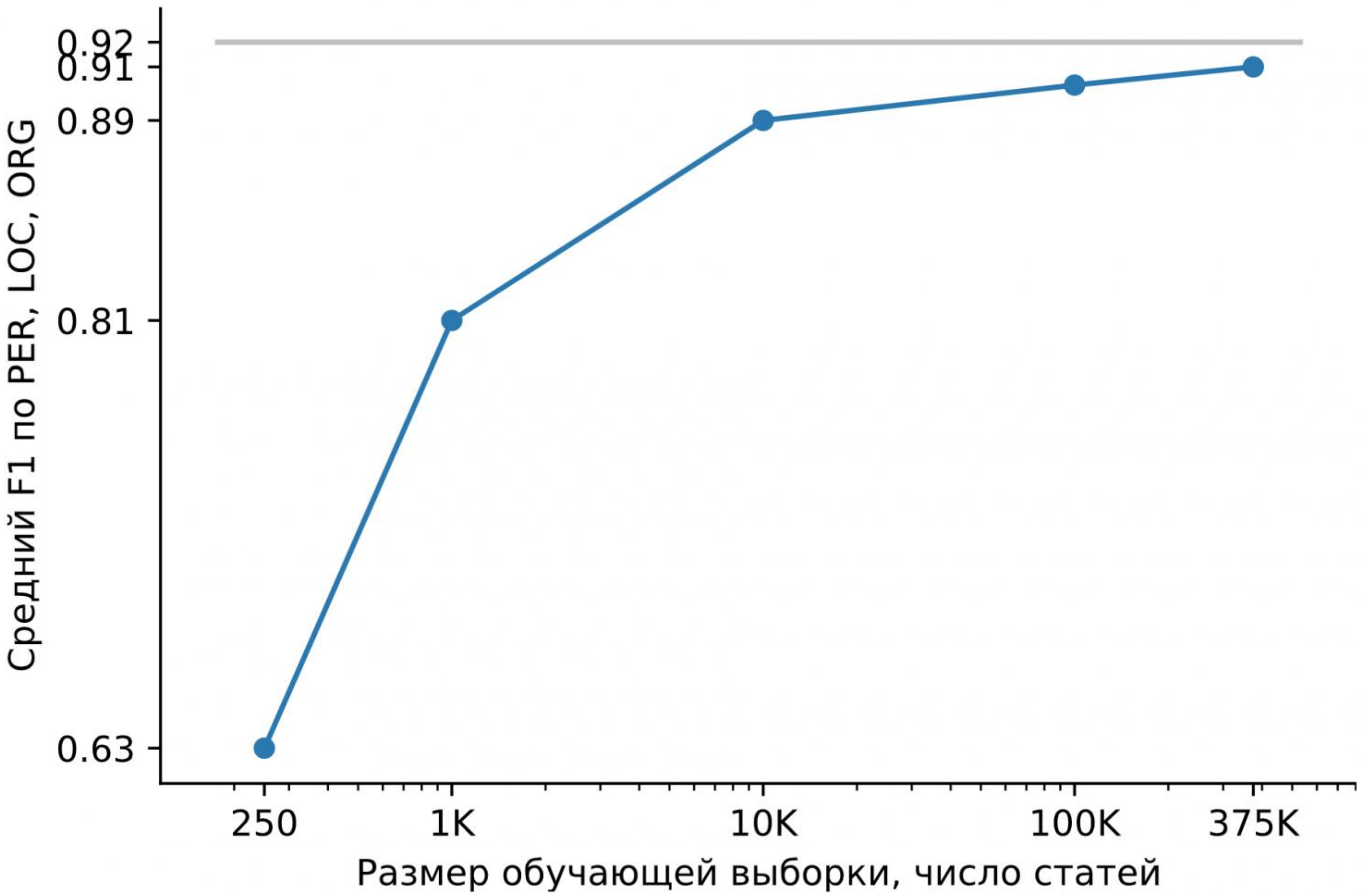
Хотим обучить лёгкую модель, но нет корпуса большого размера.

Решение:

1. Обучим очень тяжёлую модель на основе BERT на небольшом NER датасете (≈ 1k объектов)
2. Полученным теггером разметим большой датасет (≈ 700k объектов)
3. Обучим лёгкую модель (wordCNN-CRF) на синтетическом датасете

После обучения производится квантизация эмбеддингов.

Natasha: зависимость качество от размера синтетического датасета



Библиотеки для NER/POS

Библиотеки с моделями NER/POS:

- Spacy – пайплайны для разных задач (Tok2Vec + CNN / transformer), есть частичная поддержка русского языка
- UDPipe – пайплайны для разных задач (LSTM-CRF), есть частичная поддержка русского языка
- pymorphy2 – неконтекстный rule-based для русского языка
- nltk – rule-based и n-граммные модели для POS
- rnnmorph – POS для русского языка (biLSTM-CRF)
- deepavlov – NER для русского языка (transformer, BERT-based)
- natasha – NER для русского языка (CNN-CRF)