# Генеративный подход к обучению на последовательностях

И. Куралёнок

СПб, 2017

# Чему мы учимся на последовательностях

$$s = \{w_i\}_1^{T_s}, w_i \in \mathcal{A}$$

**Классификации:** есть примеры $s$ и правильные классы на них $y_i \in \{1, \ldots, k\}$

**Сегментации:** в строке $s$ в момент времени $t_s$ что-то идет не так, хотим как можно лучше предсказать $t_s$

**Генерация:** предсказываем следующий символ

**Маркировка:** каждому элементу ставим в соответствие маркер $y_i \in \{1, \ldots, k\}$

$\Rightarrow$ Все можно свести к маркировке!

# Преобразования алфавитов

- $w \in \mathbb{R}$

- $w \in \mathbb{R}^n$

- Алфавиты можно уменьшать

- Алфавиты можно увеличивать

# Преобразования алфавитов

- $w \in \mathbb{R} \Rightarrow$ "порежем" область значений $w$ (например медианами), получим обычный алфавит.
- $w \in \mathbb{R}^n$

- Алфавиты можно уменьшать

- Алфавиты можно увеличивать

# Преобразования алфавитов

- $w \in \mathbb{R} \Rightarrow$ "порежем" область значений $w$ (например медианами), получим обычный алфавит.
- $w \in \mathbb{R}^n \Rightarrow$ проделаем предыдущий фокус и запишем в алфавит все комбинации букв компонент.
- Алфавиты можно уменьшать
- Алфавиты можно увеличивать

# Преобразования алфавитов

- $w \in \mathbb{R} \Rightarrow$ "порежем" область значений $w$ (например медианами), получим обычный алфавит.
- $w \in \mathbb{R}^n \Rightarrow$ проделаем предыдущий фокус и запишем в алфавит все комбинации букв компонент.
- Алфавиты можно уменьшать $\Rightarrow$ закодируем символы в $\mathcal{A}_0 = \{0, 1\}$.
- Алфавиты можно увеличивать

# Преобразования алфавитов

- $w \in \mathbb{R} \Rightarrow$ "порежем" область значений $w$ (например медианами), получим обычный алфавит.
- $w \in \mathbb{R}^n \Rightarrow$ проделаем предыдущий фокус и запишем в алфавит все комбинации букв компонент.
- Алфавиты можно уменьшать $\Rightarrow$ закодируем символы в $\mathcal{A}_0 = \{0, 1\}$.
- Алфавиты можно увеличивать $\Rightarrow$ выделим подпоследовательности, скажем что это и есть новый алфавит.

# Общая генеративная схема

Сегодня мы будем подходить к проблеме маркировки следующим образом:

1. Пусть $\Omega = \{1, \ldots, k\}$
2. $X_t : \Omega \to \mathbb{R}^+$
3. Будем искать такие последовательности $X_t$, которые будут доставлять в максимум:

$$\sum_{t=1}^{T} \log \frac{X_t(y_t)}{\sum_y X_t(y)}$$

# Наивный подход I

Будем считать, что $X_t$ определяется только текущим символом:

$$X_t(y) = X_{w_t}(y)$$

# Наивный подход I

n-gram

Будем считать, что $X_t$ определяется только текущим символом:

$$X_t(y) = X_{w_t}(y)$$

# Наивный подход II

Будем рассматривать всю предыдущую историю как множество:

$$X_t = X : \Omega \times \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}$$

При этом, задав категории $y$ целевой развесовкой в $\mathbb{R}^{|\mathcal{A}|}$ $X$ можно строить так:

$$X_t = y^T TF(s_t)$$

# Наивный подход II

## Bag of words

Будем рассматривать всю предыдущую историю как множество:

$$X_t = X : \Omega \times \mathbb{R}^{|\mathcal{A}|} \to \mathbb{R}$$

При этом, задав категории $y$ целевой развесовкой в $\mathbb{R}^{|\mathcal{A}|}$ $X$ можно строить так:

$$X_t = y^T TF(s_t)$$

# Hidden Markov Model

## Definition (HMM)

Пускай $x_t \in \{\xi_i\}_1^p$, где $\xi_i$—простая случайная величина над $\Omega$, заданная строкой матрицы $A$. Переходы от $x_t$ к $x_{t+1}$ случайны и заданы матрицей переходов $B$. Тогда получившийся процесс называется скрытой марковской моделью с параметрами $A$ и $B$.

$\Rightarrow$ кажется, такую штуку можно использовать для нашей задачи, вопрос только в том откуда начать и как пойти, так как это почти электрон :)

# Viterbi

HMM можно учить исходя из $X_t = x_t$:

1. С помощью Байеса можно получить наиболее вероятный маршрут $r(s)$, генерирующий данный пример:
$$p_{ti} = b_{iw_t} \sum_k p_{t-1k} a_{ik}$$
$$r(s)_t = \arg\max_i p_{ti}$$
2. Будем считать, что этот маршрут и наблюдался (expectation step)
3. Подвинем веса $A$ и $B$ в соответствии с этим знанием (maximization step)

# Как получить ответ от HMM

Как у настоящего электрона, у HMM можно посчитать ожидание вместо одного генерирующего маршрута:

$$X_t = \sum_{\xi} \xi \sum_{r} P(r|s, A, B) I\{x_{rt} = \xi\}$$

Тут есть только одна проблема, $X_t$ считается довольно долго из-за того что маршрутов – трилиарды.

# Взад-назад алгоритм

## Forward-Backward algorithm

С помощью формулы Байеса легко показать, что:

$$P(X_t = \xi | s, A, B) = \sum_r P(r | s, A, B) I\{x_{rt} = \xi\} \sim f_{\xi t} b_{\xi t}$$
$$f_t = f_{t-1} A \; diag(b_{w_t})$$
$$b_t = b_{t+1} A \; diag(b_{w_{t+1}})$$

# Baum-Welch

Попробуем напрямую максимизировать:

$$(\hat{A}, \hat{B}) = \arg\max_{A,B} \sum_{t=1}^{T} \log \frac{X_t(y_t)}{\sum_y X_t(y)}$$
$$X_t = \sum_\xi \xi P(X_t = \xi | s, A, B) =$$
$$P(X_t = \xi | s, A, B) =\sim f_{\xi t} b_{\xi t}$$

# Baum-Welch

Попробуем напрямую максимизировать:

$$(\hat{A}, \hat{B}) = \arg\max_{A,B} \sum_{t=1}^{T} \log \frac{X_t(y_t)}{\sum_y X_t(y)}$$
$$X_t = \sum_\xi \xi P(X_t = \xi | s, A, B) =$$
$$P(X_t = \xi | s, A, B) =\sim f_{\xi t} b_{\xi t}$$

Это не кажется простым занятием :)
Оказывается можно максимизировать не это, а делать такие шаги:

$$(A_t, B_t) = \arg\max_{A,B} \sum_t P(w_t, y_t | A, B) \log P(w_t, y_t | A_{t-1}, B_{t-1})$$

# Критика HMM

- Конечное количество состояний
- Каждое состояние приносит $|\Omega|$ параметров
- Непонятно как делать классификацию

# Случайные поля

Это такое обобщение случайных процессов, которое позволяет индексировать случайные переменные любыми множествами. Например графами :).

# Conditional Random Fields

1. Хотим ограничить не количество состояний, а ввести зависимость состояний:

$$X_t = f(X_{t-1})$$

   В этом случае обобщение будет строится не вокруг малого количества состояний, а на степени зависимости соседних состояний.

2. Так как мы строим зависимость, можно использовать знание о $y_{t-1}$.

# Преобразование состояния в CRF

Введем функции $f$ и $g$, зависящие от соседей по графу. $f$—от перехода, $g$—от нода.

$$X_t(\lambda, \mu) = \exp\left(\sum_{k,e} \lambda_k f_k(e, y|_e, w_t) + \sum_{k,v} \mu_k g_k(v, y|_v, w_t)\right)$$

Теперь количество параметров определяется количеством функций, а не состояний.

# Подбор параметров CRF

Можно напрямую оптимизировать:

$$(\hat{\lambda}, \hat{\mu}) = \arg\max_{\lambda,\mu} \sum_s \sum_{t=1}^{T_s} \log \frac{X_t(y_t, x, \lambda, \mu)}{\sum_y X_t(y, x, \lambda, \mu)}$$

- Целевая функция **выпукла**
- Оригинально оптимизируется с помощью IIS[1] или аналогом Baum-Welch
- Кажется, что можно простым SGD, или даже GD

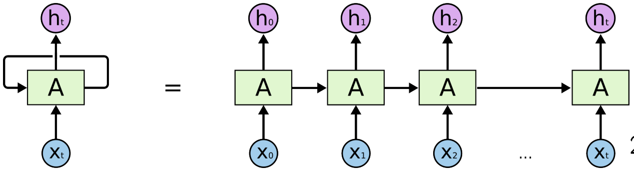
[1] Improved Iterative Scaling, Della Pietra et. al. 1997

# Критика CRF

- Надо подбирать функции исходя из доменной области
- Эффективность работы существенно меняется от набора функций

# Рекурентные нейронные сети

Когда думать не хочется, можно воспользоваться сетями :).



[2]

В нашем случае рекурентными. Назначим $X_t$ выходным вектором $h$.

---

[2]Красивые картинки взяты тут: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

# Особенности обучения RNN

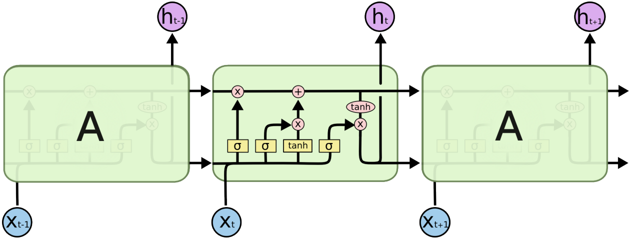
Можно использовать back propagation!

$$\frac{\partial T}{\partial \beta} = \sum_l \frac{\partial T}{\partial s_l} \frac{\partial s_l}{\partial \beta}$$

К сожалению, такая сумма очень шумная (из-за наличия дополнительных входных сигналов) + сигнал быстро затухает.

# Long-Short Term Memory

Hochreiter & Schmidhuber в 1997 г. придумали механизм сохранения сигнала:



# Что мы сегодня узнали

- Есть такая задача, работать с последовательностями
- Ее можно решать генеративными моделями на случайных процессах
- Механизм моделирования усложнялся со временем
- Из-за подхода сверху-вниз рассмотренные методики до сих пор эффективны во многих доменных областях