Анализ неструктурированных данных

Семинар 2
SENNA

Национальныи Исследовательский Университет
Высшая Школа Экономики

12 сентября 2018

SENNA (Semantic Extraction using a Neural Network Architecture)

SENNA – архитектура, позволяющая достигнуть state-of-the-art результатов в нескольких задачах обработки текстов:

- POS (part-of-speech tagging)
- CHK (chunking)
- NER (named entity recognition)
- SRL (semantic role labeling)

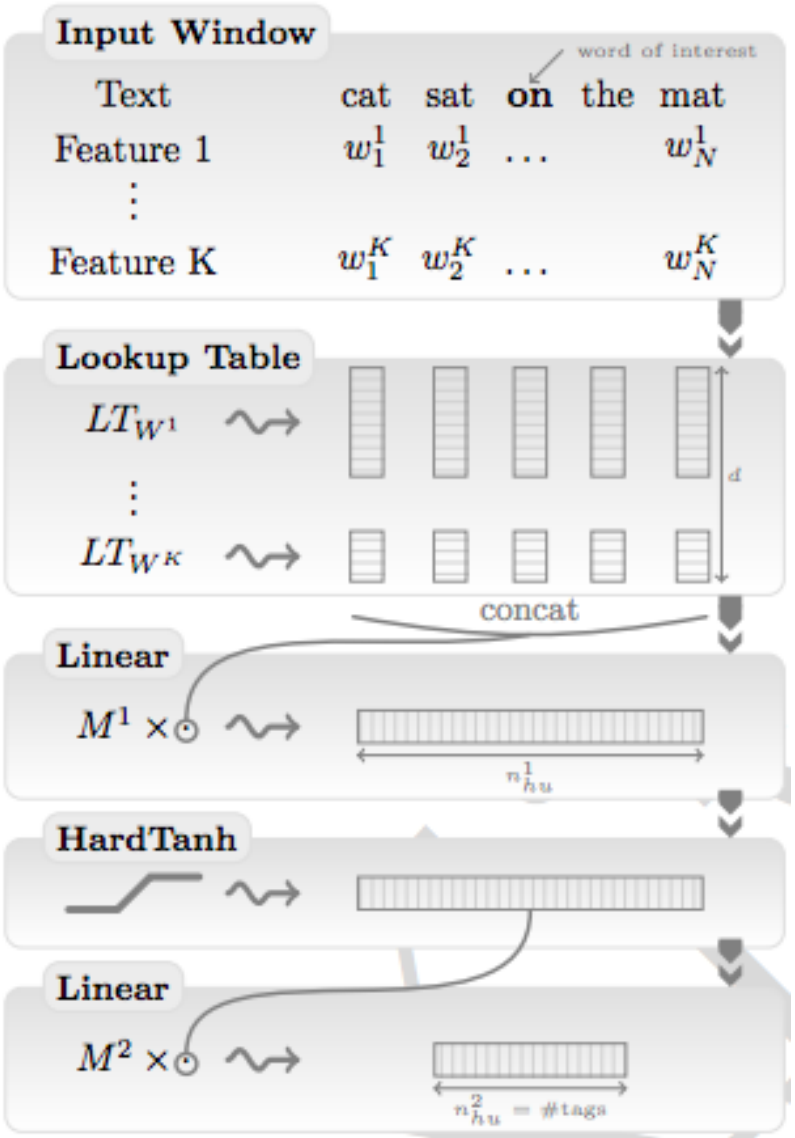
Основное преимущество подхода ("almost from scratch"): не надо генерировать фичи под каждую из задач (NN выучивает внутренние представления под каждую из задач).

Window vs Sentence approach

- Window-Approach: информация содержится в контексте слов
- Sentence-Approach: рассматриваются предложения целиком (разбор предложений)

Window approach

- Контекст каждого слова и эмбеддинги
- Конкатенируются слова из контекста
- Hidden layer
- Tanh layer
- Softmax layer
- **Можно добавить др. наборы фичей e.g., стеммированную форму слова**

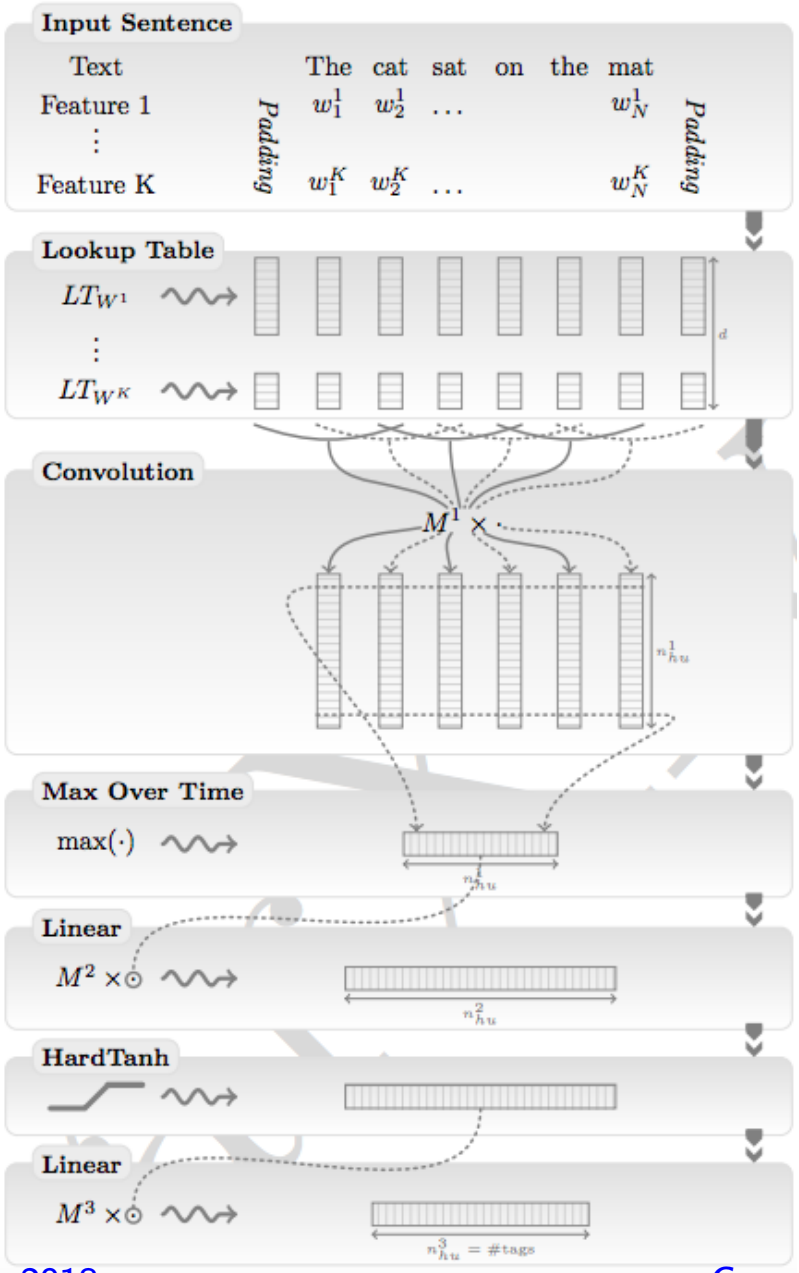


Недостатки window approach

В задаче SRL тег слова зависит от глагола/предиката, который детектируется в предложении в первую очередь. Если получится, что в окно слова предикат не попадет, тег этого слова точно не определится корректно. Поэтому хотим принимать во внимание все предложение.

Sentence approach

- Сверточный слой извлекает локальный контекст каждого слова
- **Max** over все предложение (размерность выхода сверточного слова зависит от размера предложения)
- Дальше как раньше



Word-level log-likelihood

Обозначения: x - input, θ - параметры модели, i - номер тега, $f_\theta(x)_i$ - скор, который выдала модель для i-того тега

$$p(i|x,\theta) = \frac{exp^{f_\theta(x)_i}}{\sum_j exp^{f_\theta(x)_j}}$$

$$p(y|x,\theta) = f_\theta(x)_y - logadd_j f_\theta(x)_j$$

Недостаток: не принимаем во внимание зависимость слова (для которого делаем предсказание) и тегов соседних слов.
Хотим: выучивать валидные последовательности тегов, используя предсказания тегов для всех слов предложения.

Sentence-level log-likelihood

Идея: вводим дополнительный параметр для обучения $(A)_{ij}$ - transition score от i до j тега. Итого $\tilde{\theta} = \theta \cup \{(A)_{ij}\}_{i,j}$
Тогда скор для всего предложения вдоль последовательности тегов:

$$s([x]_1^T, [i]_1^T, \tilde{\theta}) = \sum_{t=1}^{T} \left([A]_{[i]_{t-1}, [i]_t} + [f_\theta]_{[i]_t, t} \right)$$

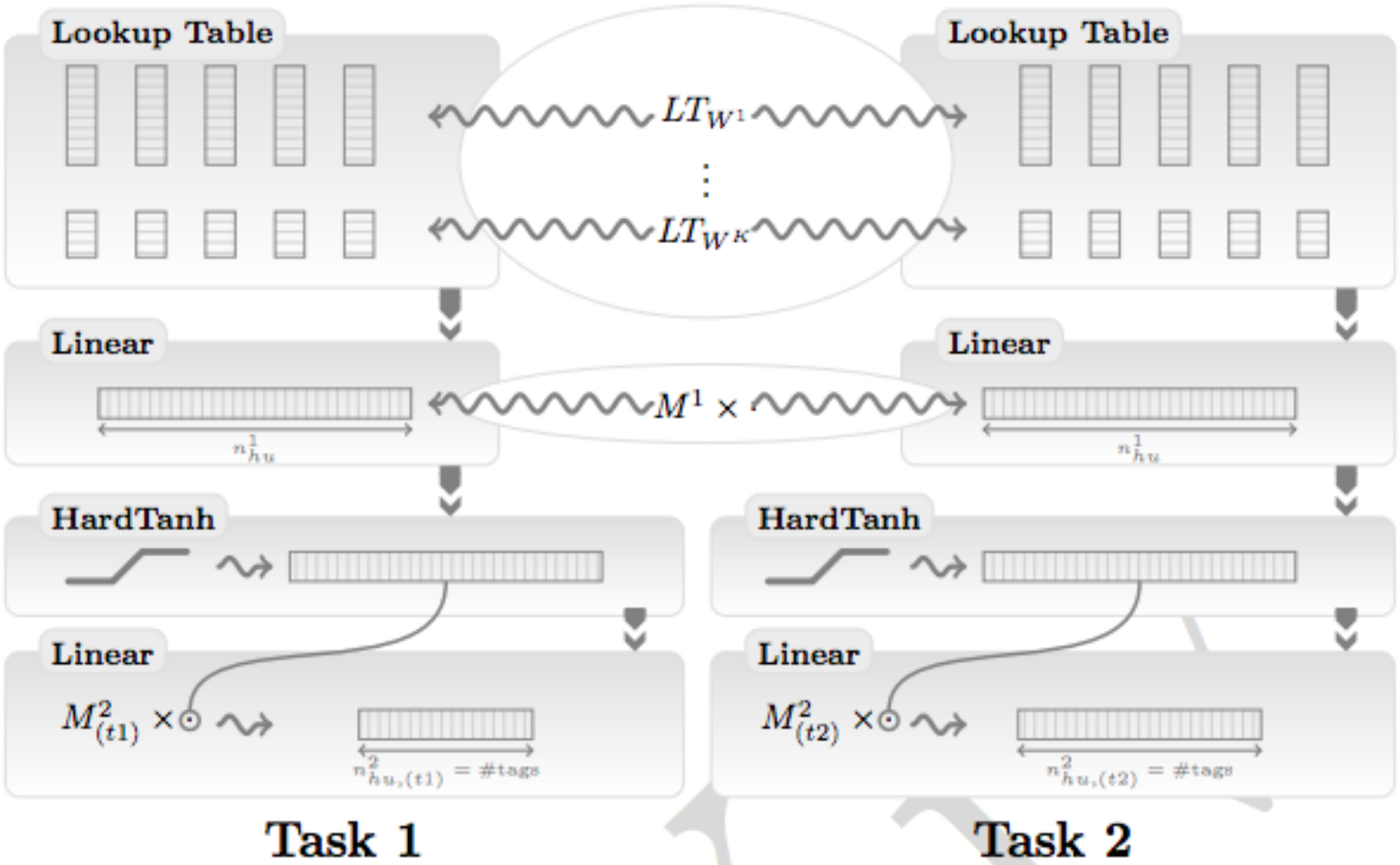
Далее берем софтмакс вдоль всех путей аналогично тому, как делали для слова:

$$\log p([y]_1^T \mid [x]_1^T, \tilde{\theta}) = s([x]_1^T, [y]_1^T, \tilde{\theta}) - \operatorname*{logadd}_{\forall [j]_1^T} s([x]_1^T, [j]_1^T, \tilde{\theta})$$

Сколько всего путей, вдоль которых нужно взять софтмакс?

Multitask learning

Идея: информация из выученных представлений для одной задачи могут быть полезны для решения другой задачи



Задание семинара

Не забудьте, пожалуйста, **сделать копию** ноутбука!
Семинар в google colab тут

Материалы

- NLP (almost) from scratch
- реализация SENNA