# DEEP LEARNING X: WORD2VEC, GLOVE И ПРОЧ.

Сергей Николенко
СПбГУ — Санкт-Петербург
24 ноября 2018 г.

*Random facts*:

- 24 ноября в странах Арктики — День моржа, а в Турции — День учителя
- 24 ноября 1835 г. правительство Техаса учредило особый вид конной полиции — техасских рейнджеров
- 24 ноября 1859 г. вышла книга Чарльза Дарвина «Происхождение видов путём естественного отбора или сохранение благоприятствуемых пород в борьбе за жизнь»; первое издание книги разошлось за один день
- 24 ноября 1964 г. бельгийские десантники взяли Стэнливиль, а 24 ноября 1965 г. на встрече армейского руководства в Киншасе было решено передать власть Мобуту Сесе Секо Куку Нгбенду ва за Банга, урождённому Жозефу-Дезире Мобуту; в тот же день он объявил об отставке президента Касавубу и премьера Кимбы, и началось...
- 24 ноября 1973 г. из-за нефтяного кризиса 1973 г. на немецких автобанах было введено ограничение скорости; впрочем, продержалось оно буквально четыре месяца

# Распределённые представления слов и других объектов

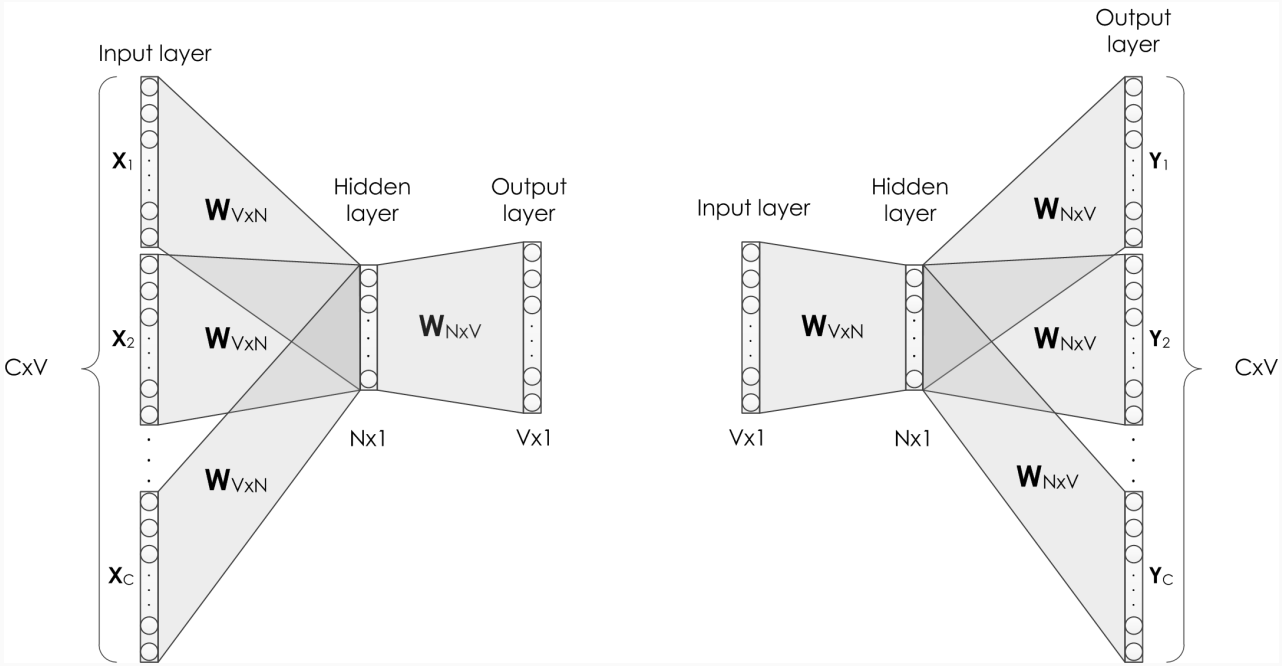
# Word embeddings

- Distributional hypothesis в лингвистике: слова с похожими значениями будут встречаться в похожих контекстах.
- *Распределённые представления слов* (distributed word representations, word embeddings) отображают слова в евклидово пространство $\mathbb{R}^d$, обычно $d$ порядка нескольких сотен:
  - началось в (Bengio et al. 2003; 2006), хотя подобные идеи были и раньше;
  - *word2vec* (Mikolov et al. 2013): обучаем веса, которые лучше всего решают простую задачу предсказания слова по его контексту: continuous bag-of-words (CBOW) и skip-gram;
  - *Glove* (Pennington et al. 2014): обучаем веса слов, раскладывая матрицу совместной встречаемости.

# Word embeddings

- CBOW предсказывает слово из локального контекста:
  - входы – one-hot представления слов размерности $V$;
  - скрытый слой – матрица представлений слов $W$;
  - выход скрытого слоя – среднее векторов слов контекста;
  - на выходе получаем оценку $u_j$ для каждого слова и берём $\mathrm{softmax}$:

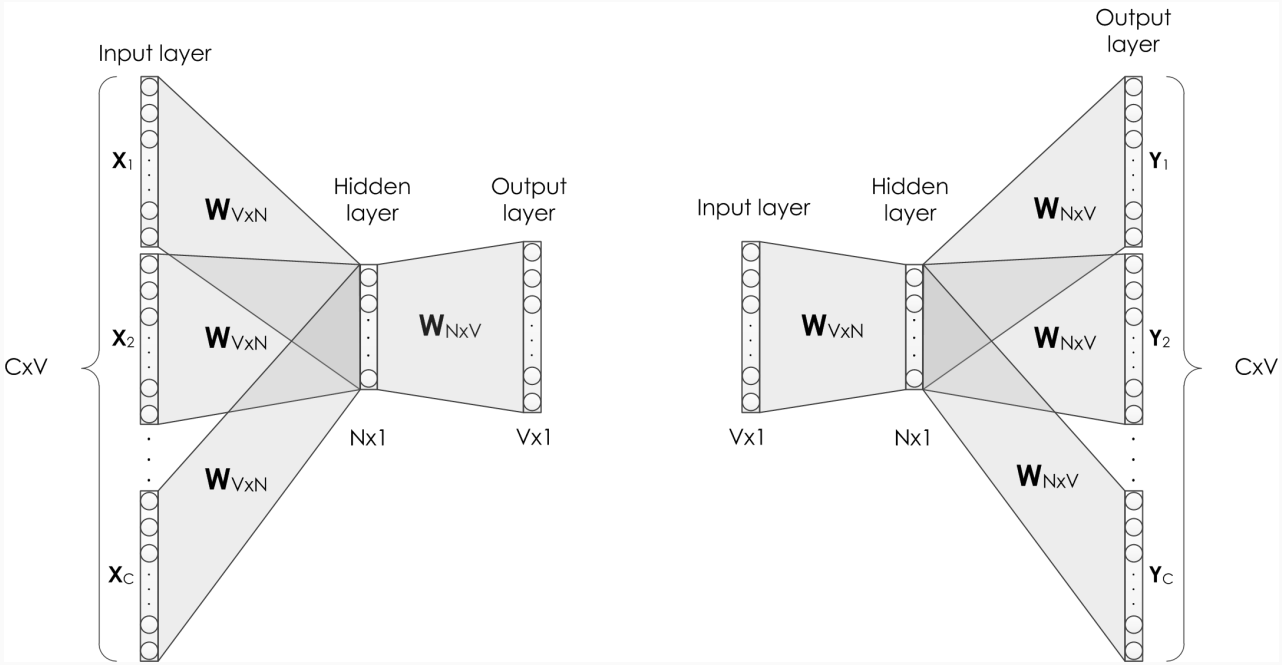
$$\hat{p}(i|c_1, \dots, c_n) = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u_{j'})}.$$



# Word embeddings

- Skip-gram предсказывает слова контекста из текущего слова:
  - предсказываем каждое слово контекста из центрального;
  - теперь несколько мультиномиальных распределений и по $\mathrm{softmax}$ для каждого слова контекста:

$$\hat{p}(c_k|i) = \frac{\exp(u_{kc_k})}{\sum_{j'=1}^{V} \exp(u_{j'})}.$$



# Представления слов

- Как обучить такую модель?
- Например, в skip-gram выбираем $\theta$ так, чтобы максимизировать

$$L(\theta) = \prod_{i \in D} \left( \prod_{c \in C(i)} p(c \mid i; \theta) \right) = \prod_{(i,c) \in D} p(c \mid i; \theta),$$

параметризуем

$$p(c \mid i; \theta) = \frac{\exp(\tilde{w}_c^\top w_i)}{\sum_{c'} \exp(\tilde{w}_{c'}^\top w_i)}.$$

# Представления слов

- Теперь общее правдоподобие равно

$$\arg\max_\theta \prod_{(i,c)\in D} p(c \mid i;\theta) = \arg\max_\theta \sum_{(i,c)\in D} p(c \mid i;\theta) =$$

$$= \arg\max_\theta \sum_{(i,c)\in D} \left( \exp(\tilde{w}_c^\top w_i) - \log \sum_{c'} \exp(\tilde{w}_{c'}^\top w_i) \right),$$

  и его можно максимизировать отрицательным сэмплированием (negative sampling).

- Вопрос: зачем нужно разделять векторы $\tilde{w}$ и $w$?

# Представления слов

- GloVe – пытаемся приблизить матрицу встречаемости $X \in \mathbb{R}^{V \times V}$:

$$p_{ij} = p(j \mid i) = \frac{X_{ij}}{X_i} = \frac{X_{ij}}{\sum_k X_{ik}}.$$

- Точнее, их отношения $\frac{p_{ij}}{p_{kj}}$.
- Пример на русской википедии:

| Слово $k$ | Число вхождений | | | Вероятности $p(k \mid ...)$, $\times 10^{-4}$ | | Отношение $\frac{p(k\mid\text{клуб})}{p(k\mid\text{команда})}$ |
|---|---|---|---|---|---|---|
| | Всего | Вместе с: | | | | |
| | | клуб | команда | клуб | команда | |
| футбол | 29988 | 54 | 34 | 18.0 | 11.3 | 1.588 |
| хоккей | 10957 | 16 | 7 | 6.39 | 14.6 | 2.286 |
| гольф | 2721 | 11 | 1 | 40.4 | 3.68 | 11.0 |
| корабль | 100127 | 0 | 30 | 0.0 | 3.00 | 0.0 |

# Представления слов

- Обучаем функцию $F(w_i, w_j; \tilde{w}_k) = \frac{p_{ij}}{p_{kj}}$.
- Ещё проще – обучаем

$$F((w_i - w_j)^\top \tilde{w}_k) = \frac{F\left(w_i^\top \tilde{w}_k\right)}{F\left(w_j^\top \tilde{w}_k\right)} = \frac{p_{ij}}{p_{kj}}.$$

- Это фактически должна быть экспонента:

$$w_i^\top \tilde{w}_k = \log(p_{ik}) = \log(X_{ik}) - \log(X_i).$$

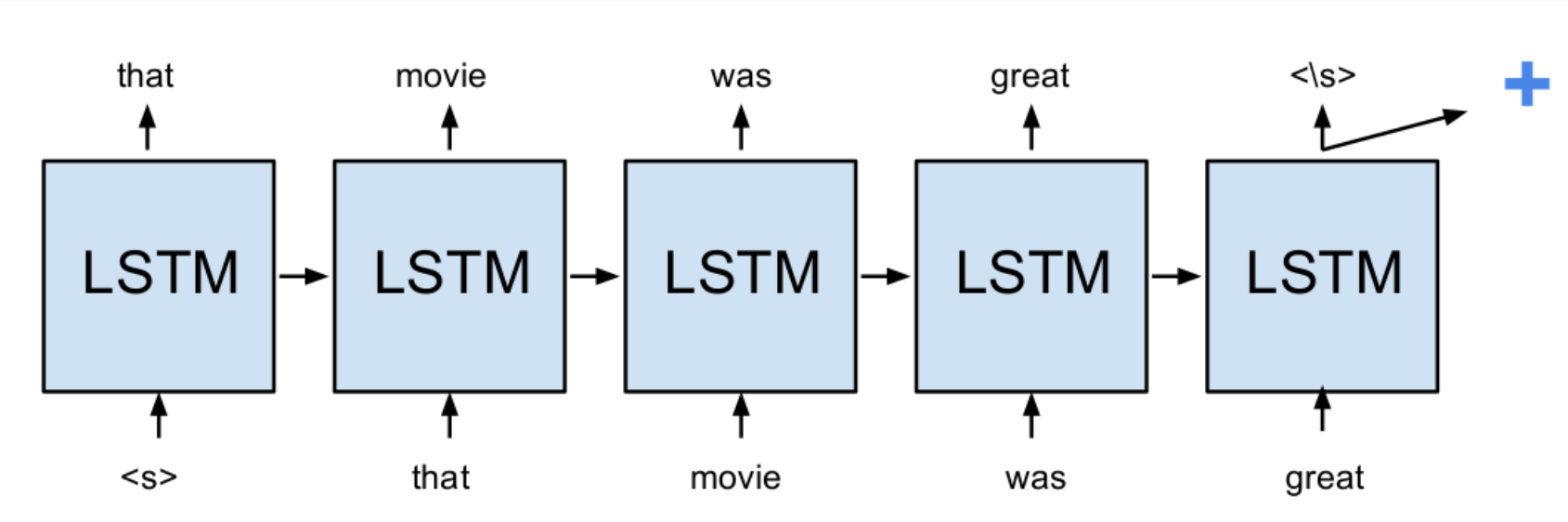
- Можно спрятать $\log(X_i)$ как $w_i^\top \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik})$.
- И целевая функция у GloVe будет

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij}\right)^2.$$

- Демо: ближайшие соседи, геометрические соотношения.

# Как использовать векторы слов

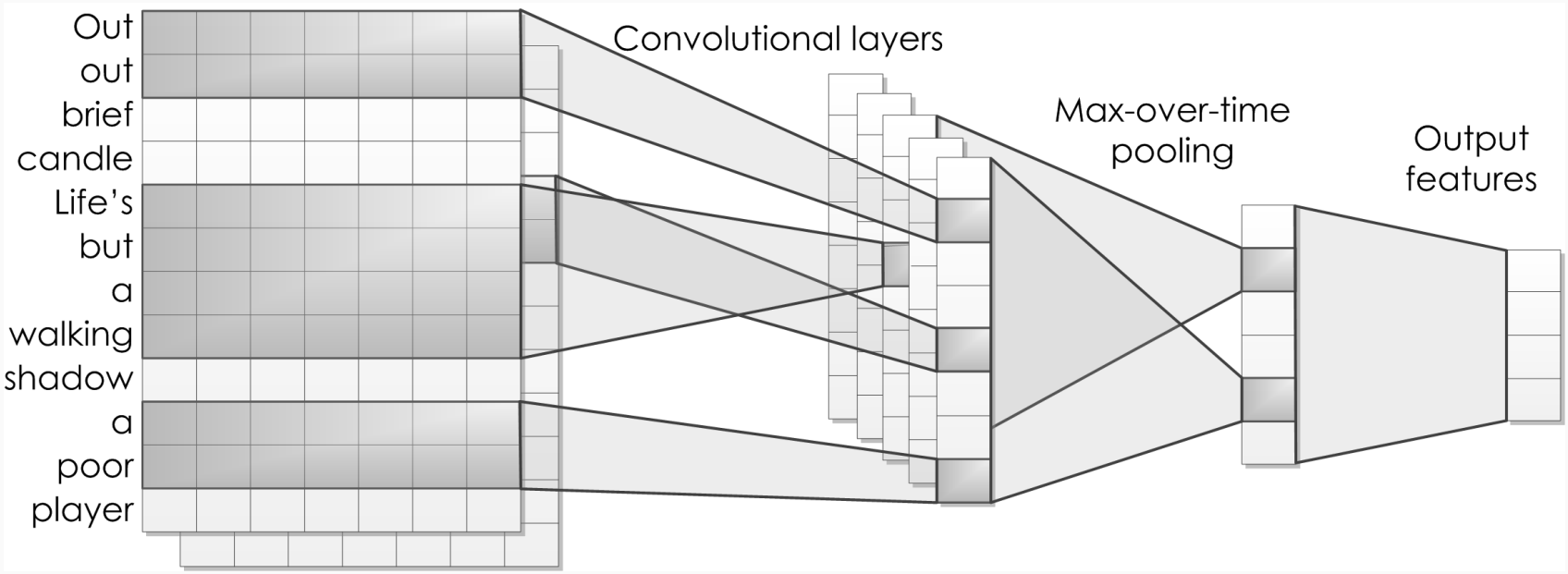
- Теперь можно просто строить нейронные сети поверх вложений.
- Например, стандартные LSTM для анализа тональности:




- Обучаем seq2seq для моделирования языка, затем используем последний выход или средний выход для тональности.

# Как использовать векторы слов

- Или CNN с одномерными свёртками:



# Примеры представлений слов

- Примеры по-русски:
  - ближайшие соседи слова **конференция**:

| | |
|---|---|
| пресс-конференция | 0.6919 |
| программа | 0.6827 |
| выставка | 0.6692 |
| ассоциация | 0.6638 |
| кампания | 0.6406 |
| ярмарка | 0.6372 |
| экспедиция | 0.6305 |
| презентация | 0.6243 |
| сходка | 0.6162 |
| встреча | 0.6100 |

# Примеры представлений слов

- Антонимы тоже подходят:
  - ближайшие соседи слова **любовь**:

| | |
|---|---|
| жизнь | 0.5978 |
| нелюбовь | 0.5957 |
| приязнь | 0.5735 |
| боль | 0.5547 |
| страсть | 0.5520 |

  - ближайшие соседи слова **синоним**:

| | |
|---|---|
| антоним | 0.5459 |
| эвфемизм | 0.4642 |
| анаграмма | 0.4145 |
| омоним | 0.4048 |
| оксюморон | 0.3930 |

# Примеры представлений слов

- Русский сексизм:
  - ближайшие соседи слова **программист**:

| | |
|---|---|
| компьютерщик | 0.5618 |
| программер | 0.4682 |
| электронщик | 0.4613 |
| автомеханик | 0.4441 |
| криптограф | 0.4316 |

  - ближайшие соседи слова **программистка**:

| | |
|---|---|
| стажерка | 0.4755 |
| инопланетянка | 0.4500 |
| американочка | 0.4481 |
| предпринимательница | 0.4442 |
| студенточка | 0.4368 |

# Примеры представлений слов

- Какими будут
  - ближайшие соседи слова **комендантский**?

# Примеры представлений слов

- Какими будут
  - ближайшие соседи слова **комендантский**:

| | |
|---|---|
| неурочный | 0.7276 |
| неровен | 0.7076 |
| урочный | 0.6849 |
| ровен | 0.6756 |
| предрассветный | 0.5867 |
| условленный | 0.5597 |

# Вверх и вниз от представлений слов

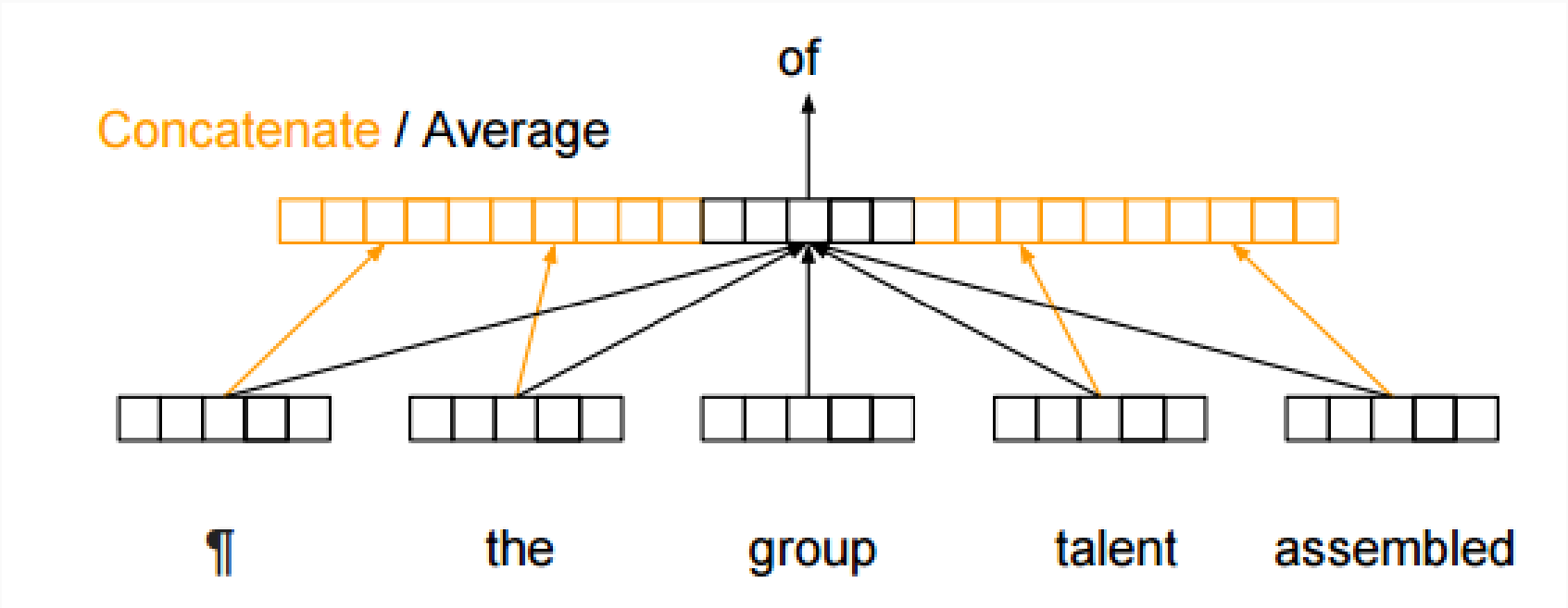
- Представления слов – первый шаг многих нейросетевых моделей в NLP.
- Но от представлений слов можно двигаться в обоих направлениях.
- Во-первых, предложение – не обязательно сумма своих слов.
- Во-вторых, слово не так уж атомарно, как модели хотелось бы думать.

# Представления предложений

- Как комбинировать векторы слов в векторы «кусков текста»?
- Простейшая идея: берём сумму и/или среднее представлений слов как представление предложения/абзаца:
  - это baseline в (Le and Mikolov 2014) и многих других работах;
  - разумный метод для коротких фраз в (Mikolov et al. 2013);
  - и довольно эффективно работает для реферирования в (Kageback et al. 2014).
- Совсем не так плохо, скорее всего как раз из-за геометрических соотношений.

# Представления предложений

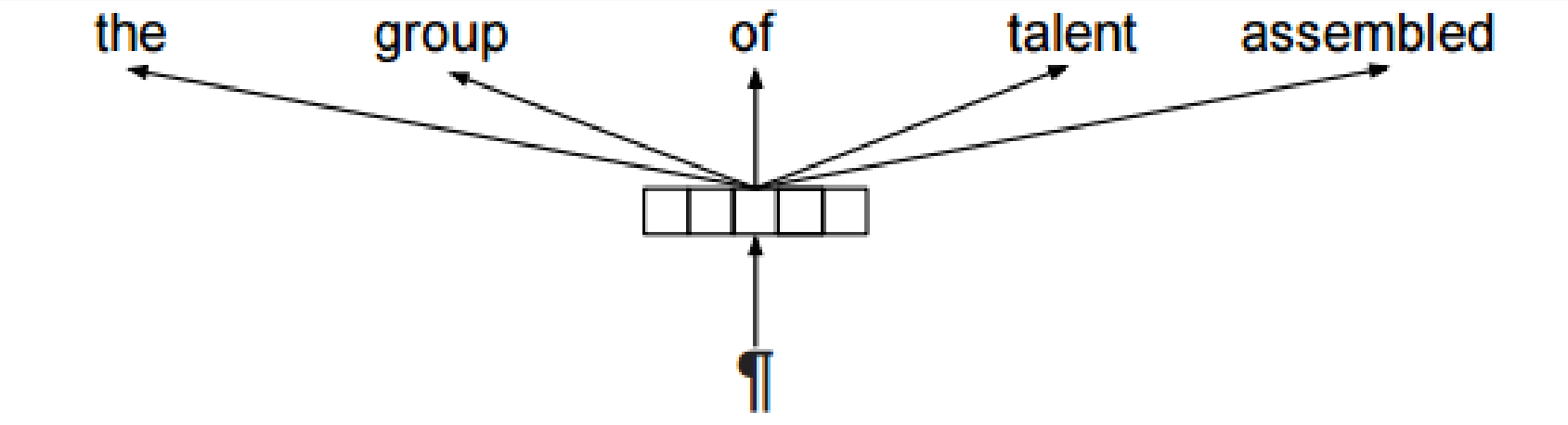
- Как комбинировать векторы слов в векторы «кусков текста»?
- Distributed Memory Model of Paragraph Vectors (PV-DM) (Le and Mikolov 2014):
  - вектор абзаца – дополнительный вектор для каждого абзаца;
  - служит «памятью» для более глобального контекста.



# Представления предложений

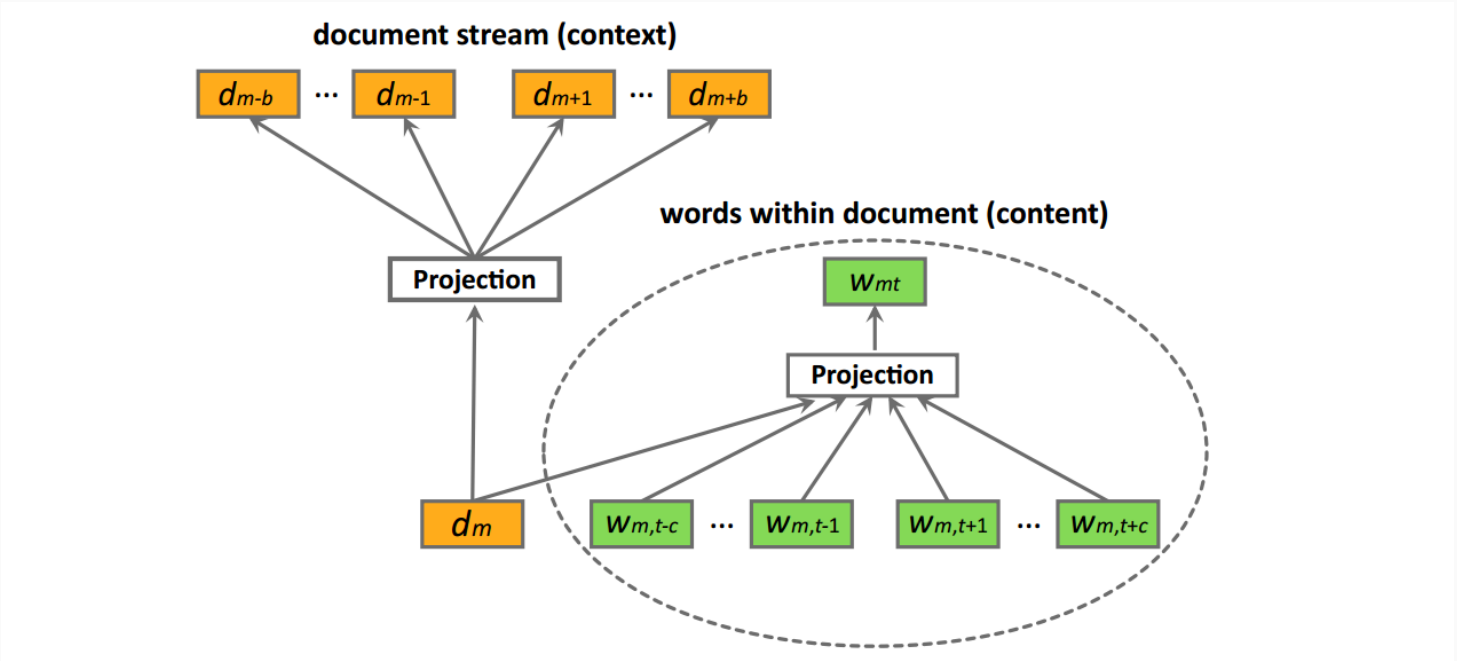
- Как комбинировать векторы слов в векторы «кусков текста»?

- Distributed Bag of Words Model of Paragraph Vectors (PV-DBOW) (Le and Mikolov 2014):
  - модель предсказывает слова, случайно выбранные из данного абзаца;
  - и вектор абзаца помогает предсказывать слова из этого абзаца во всех локальных контекстах.



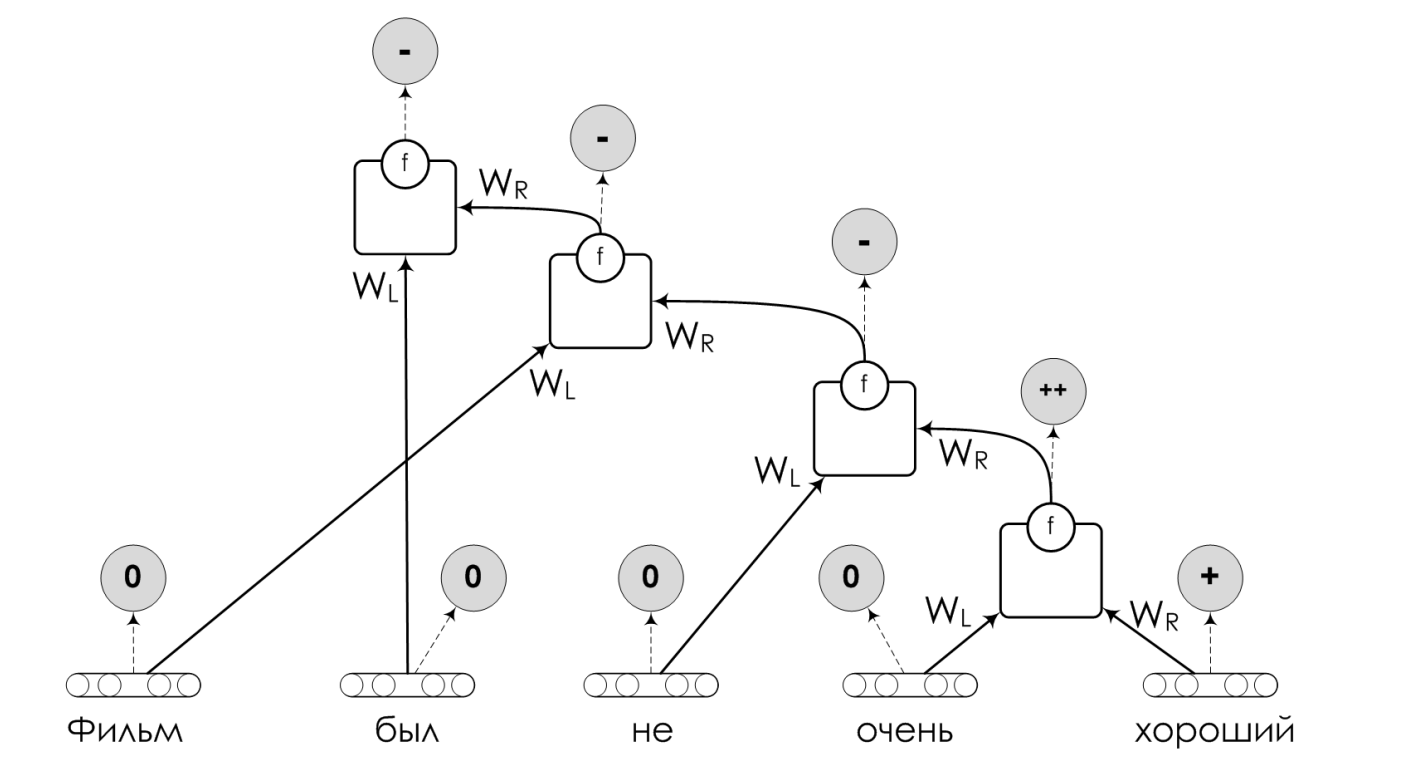
# Представления предложений

- Как комбинировать векторы слов в векторы «кусков текста»?
  - Свёрточные архитектуры (Ma et al., 2015; Kalchbrenner et al., 2014).
  - (Kiros et al. 2015): skip-thought векторы, обучаются из skip-gram векторов на предложениях.
  - (Djuric et al. 2015): моделирует большие потоки текстов иерархическими нейросетевыми моделями.



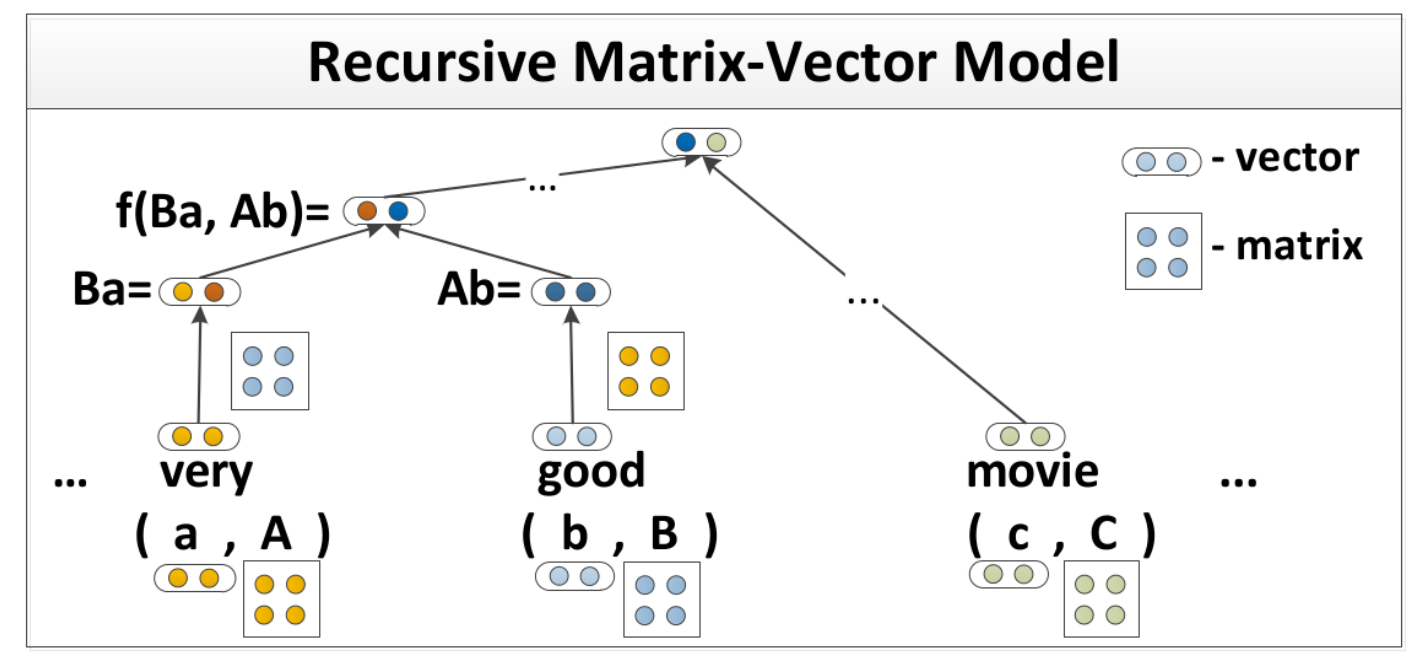
# Глубокие рекурсивные сети

- *Рекурсивные нейронные сети* (recursive neural networks; Socher et al., 2012):
  - нейронная сеть сворачивает представление куска текста в двух поддеревьях, проходя от слов до корня дерева синтаксического разбора.



# Глубокие рекурсивные сети

- *Рекурсивные нейронные сети* (recursive neural networks; Socher et al., 2012):
  - можно представлять узел матрицей и вектором;
  - в целом очень эффективный подход к анализу тональности (Socher et al. 2013).



# Глубокие рекурсивные сети

- Глубокие рекурсивные сети для анализа тональности (Irsoy, Cardie, 2014).
- Первая идея: отделим листья от внутренних узлов; вместо применения одних и тех же весов

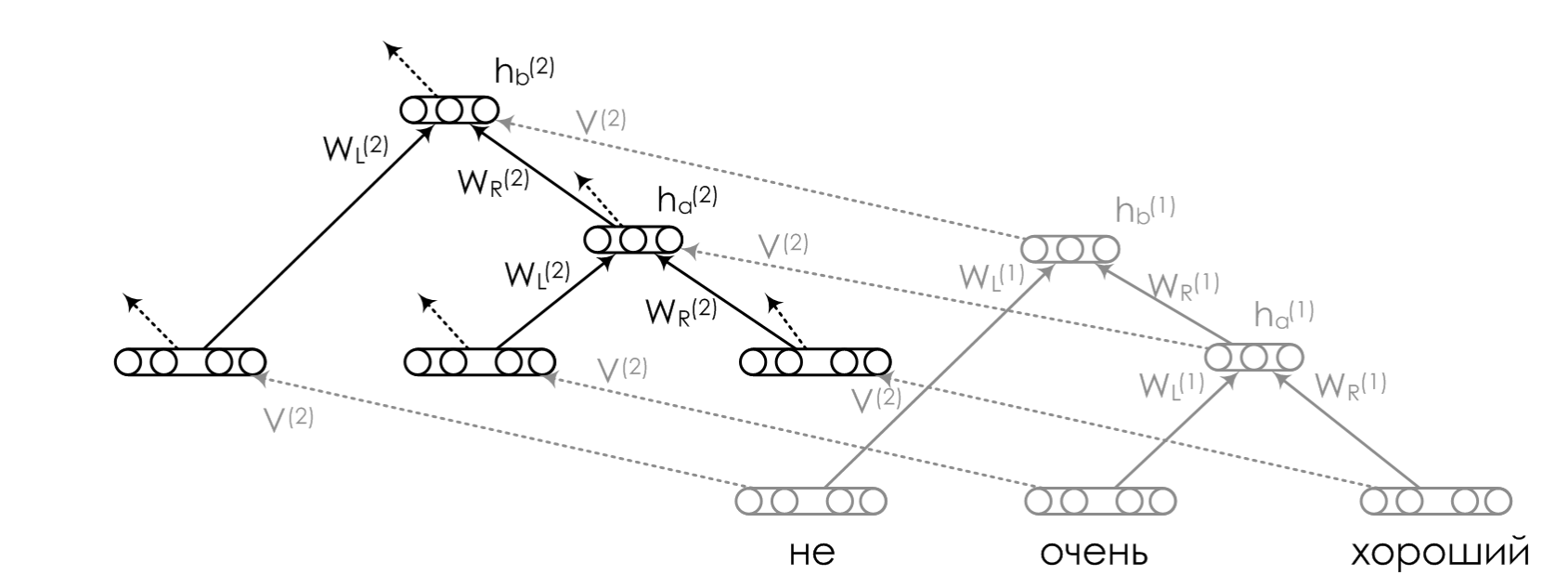
$$x_v = f(W_L x_{l(v)} + W_R x_{r(v)} + b)$$

теперь будут разные матрицы для листьев (слов) и внутренних узлов:

  - теперь внутренняя размерность может быть меньше;
  - и можно использовать ReLU, т.к. теперь нет проблемы с разреженными входами и плотными внутренними узлами.

# Глубокие рекурсивные сети

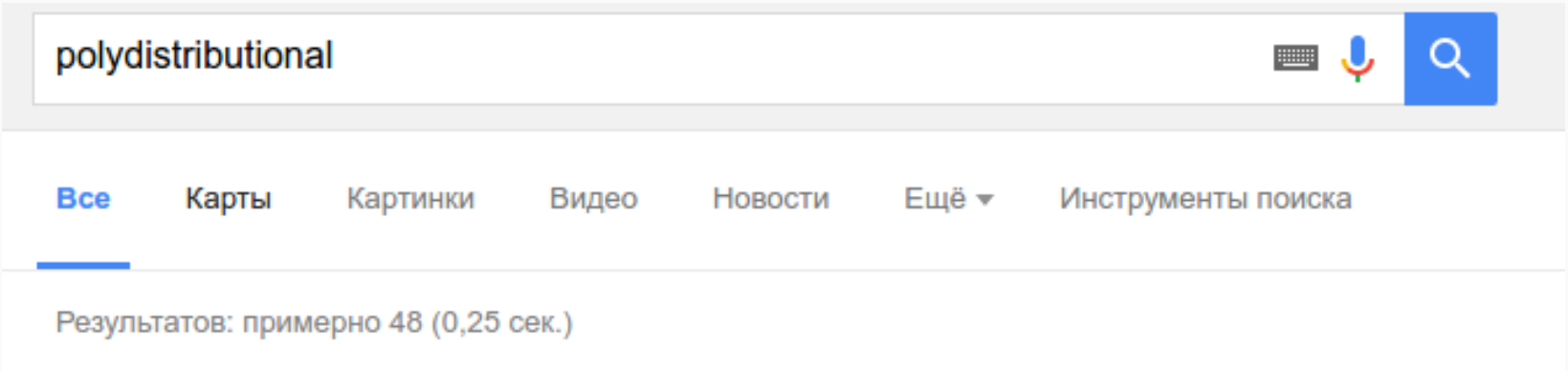
- Вторая идея – добавим в иерархическое представление глубины: $h_v^{(i)} = f(W_L^{(i)} h_{l(v)}^{(i)} + W_R^{(i)} h_{r(v)}^{(i)} + V^{(i)} h_v^{(i-1)} + b^{(i)})$.




- Прекрасная архитектура для анализа тональности… если, конечно, есть деревья разбора.
- Stanford Sentiment TreeBank есть; а по-русски непонятно…

# Character-level models

- Важные недостатки векторов слов:
  - векторы независимы, а слова нет (морфология);
  - то же относится к словам не из словаря (новым, очень редким);
  - модель очень большая получается, если все слова обучать.
- Например, слово “polydistributional” даёт 48 результатов в Google, так что вы его скорее всего никогда не видели, и обучаться не на чем:



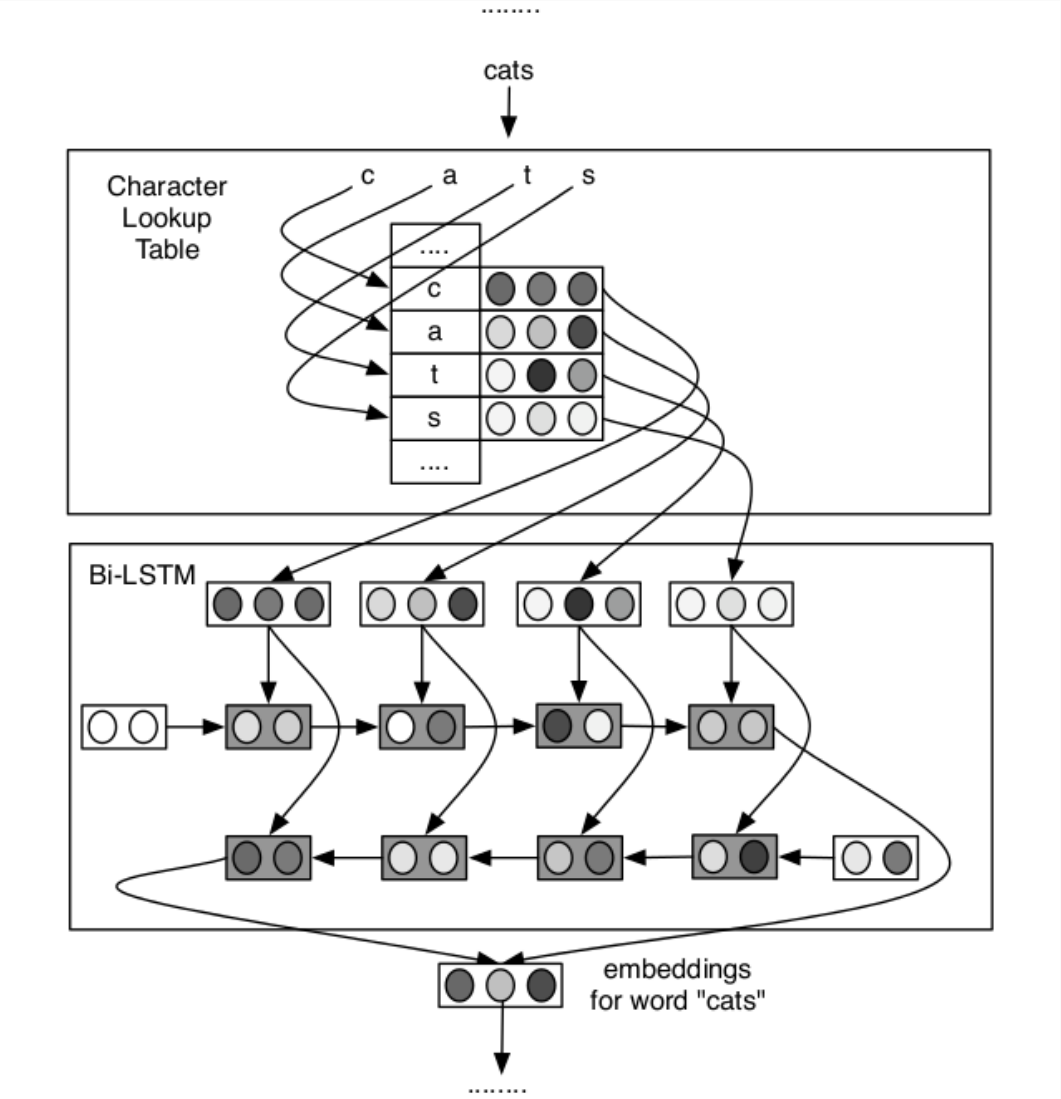

- Понимаете, что оно значит? Я тоже понимаю.

# Character-level models

- Отсюда идея *character-level representations*:
  - сначала раскладывали слово на морфемы (Luong et al. 2013; Botha and Blunsom 2014; Soricut and Och 2015);
  - но тогда надо делать морфологический анализ, тоже непросто;
  - два естественных подхода на буквах: LSTM/GRU и CNN;
  - в любом случае, модель медленная, но к каждому слову применять не обязательно, можно частично закешировать заранее.

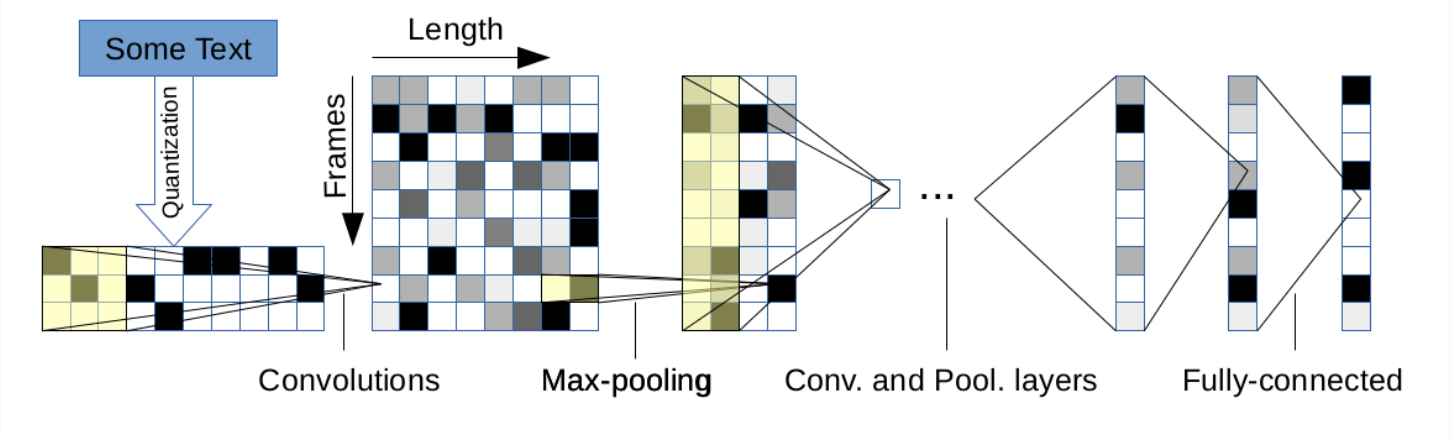
# Character-level models

- C2W (Ling et al. 2015) основано на двунаправленных LSTM:



# Character-level models

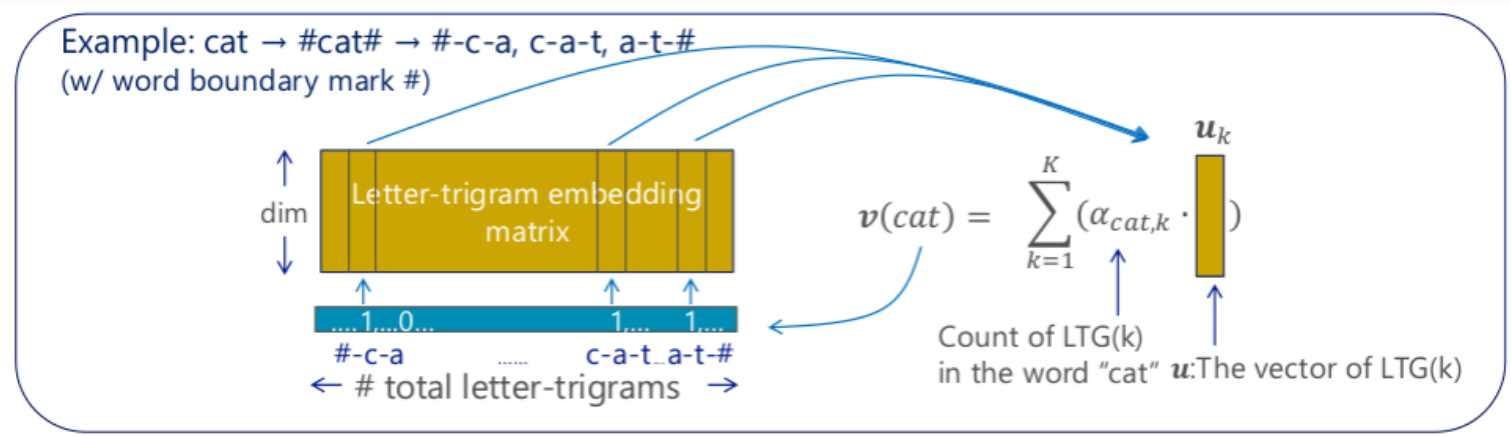
- ConvNet (Zhang et al. 2015): понимание текста с нуля, с букв, целиком на CNN.




- Character-level модели становятся всё важнее и нужнее, особенно для (1) морфологически богатых языков и (2) порождённых пользователями текстов.

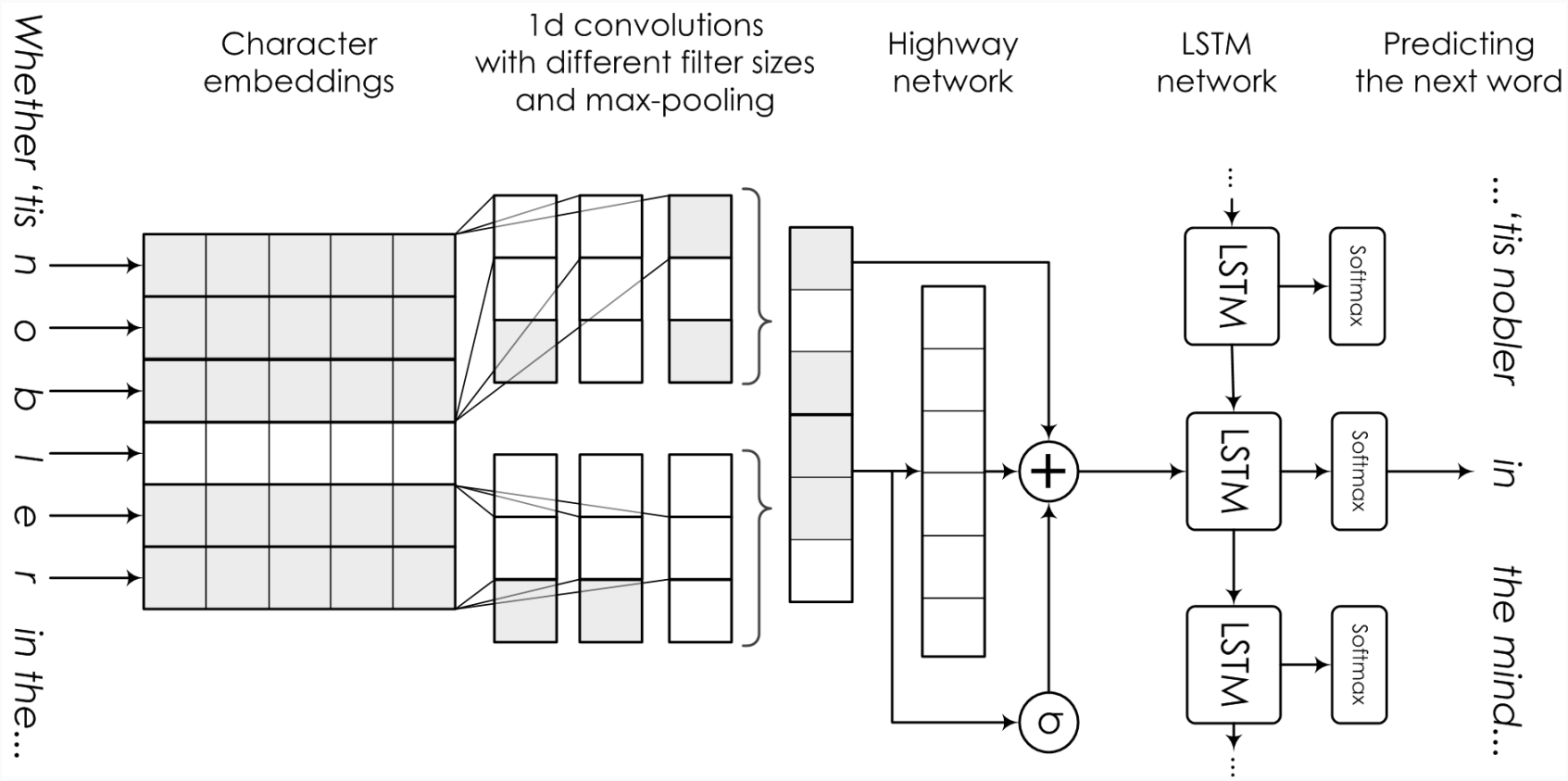
# Character-level models

- Подход *Deep Structured Semantic Model* (DSSM) (Huang et al., 2013; Gao et al., 2014a; 2014b):
  - sub-word embeddings: представим слово как множество триграмм букв;
  - словарь теперь $|V|^3$ (десятки тысяч вместо миллионов), но коллизии очень редки;
  - представление устойчиво к опечаткам и т.п. (очень важно для порождённых пользователями текстов).



# Пример модели: Kim et al., 2015

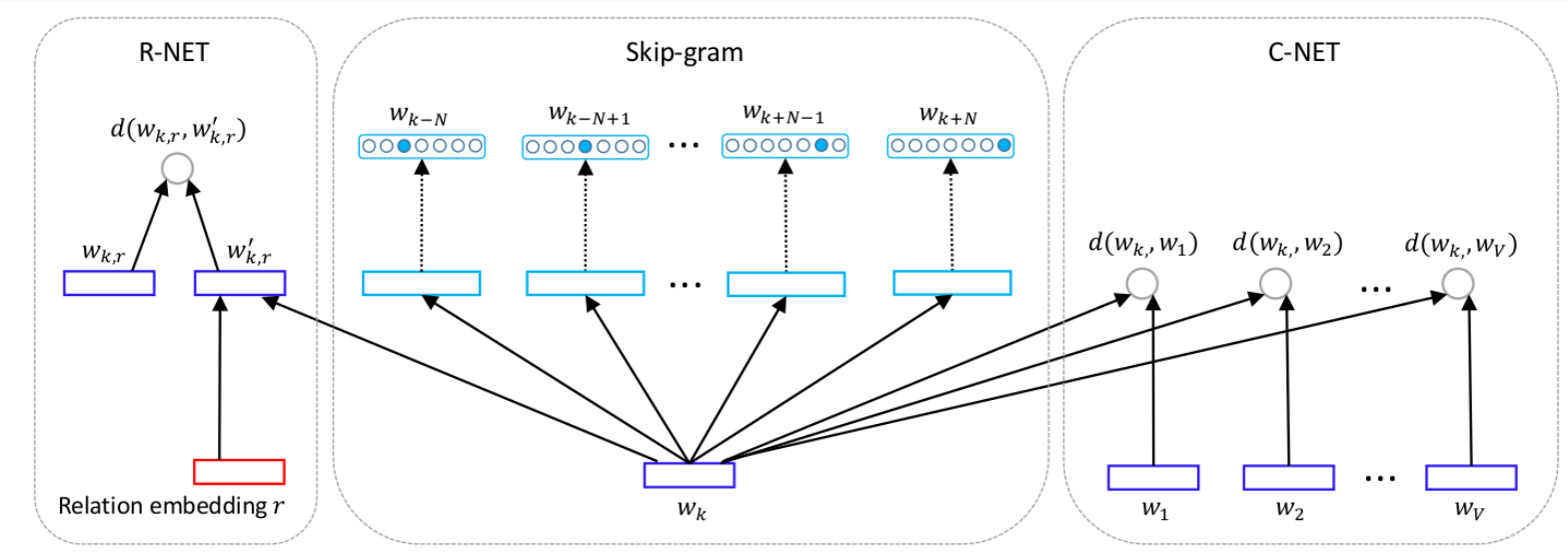
- Пример современной character-based языковой модели (Kim et al., 2015):




- Объединяет CNN, RNN, highway networks, embeddings...

# Вектора слов с внешней информацией

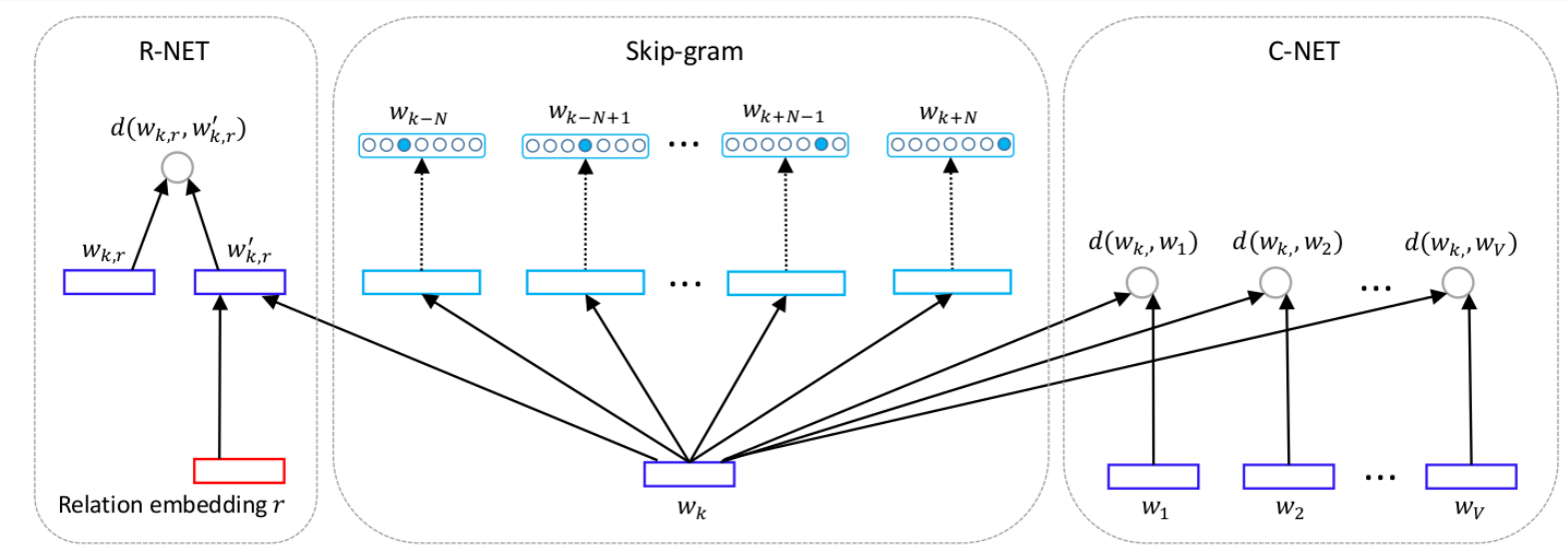
- Другие модификации представлений слов добавляют внешнюю информацию.
- Например, модель RC-NET (Xu et al. 2014) расширяет skip-gram семантическими и синтаксическими отношениями и знаниями.



# Вектора слов с внешней информацией

- И базовый *word2vec* получает регуляризатор для каждого отношения в базе, пытаясь выразить его линейным соотношением:

$$w_{\text{Hinton}} - w_{\text{Wimbledon}} \approx r_{\text{born at}} \approx w_{\text{Euler}} - w_{\text{Basel}}$$



# Разрешение смысла слов

- Другая важная проблема для всех моделей выше – омонимы.
- Как различить разные смыслы слова? Беда:
  - в модели обычно обучится только один смысл;
  - давайте проверим ближайших соседей слова **коса** и других омонимов.
- Надо добавить *скрытые* переменные для разных возможных смыслов и потом вывести их из контекста.
- Чтобы обучить смыслы со скрытыми переменными — стохастический вариационный вывод (Bartunov et al., 2015).

# Спасибо!

Спасибо за внимание!