Анализ неструктурированных данных

Семинар 5

Языковые модели.

Мурат Апишев (great-mel@yandex.ru)

НИУ ВШЭ (ГУ)

4 октября, 2018

Содержание занятия

- Классические языковые модели
  - Определение
  - Приложения
  - Виды сглаживания
- Нейросетевые модели
  - Архитектуры RNN/LSTM (повторение)
  - Нейросетевая языковая модель
  - Генерация текстов

Языковая модель

- Хотим присвоить вероятности последовательностям слов
- *Языковой моделью* называется модель, умеющая вычислять хоть одну из этих вероятностей:
  1. $P(W) = P(w_1, \ldots, w_n)$
  2. $P(w_n \mid w_1, \ldots, w_{n-1})$
- Цепное правило:

$$P(X_1, \ldots, X_n) = P(X_1)P(X_2 \mid X_1) \ldots P(X_n \mid X_1, \ldots, X_{n-1})$$

- Формула условной вероятности:

$$P(X_n \mid X_1, \ldots, X_{n-1}) = \frac{P(X_1, \ldots, X_n)}{P(X_1, \ldots, X_{n-1})},$$

Языковая модель

- Для оценивания $P(X_n \mid X_1, \ldots, X_{n-1})$ нужно посчитать $P(X_1, \ldots, X_n)$ и $P(X_1, \ldots, X_{n-1})$
- С ростом n число всевозможных последовательностей растёт экспоненциально $\Rightarrow$

  Проблема: для большого n эти вероятности близки к нулю
- Для упрощения применим *марковское предположение*:

$$P(X_n \mid X_1, \ldots, X_{n-1}) \approx P(X_n \mid X_{n-k+1}, \ldots, X_{n-1}), \quad k \ll n$$

- Окончательная формула ($w_{i-k}^{i} := w_{i-k}, \ldots, w_i$)

$$P(w_1, \ldots, w_n) = \prod_i P(w_i \mid w_{i-k+1}^{i-1})$$

Приложения

- Генерация текста
- Распознавание речи/текста
- Машинный перевод
- Исправление опечаток
- Определения языка
- Определение части речи (POS)
- ...

Открытые вопросы

1. Как обучать вероятности?
2. Как использовать их для генерации текста?
3. Какие проблемы могут возникнуть при обучении?
4. Как эти проблемы решать?

Ответы

1. Статистически по обучающему корпусу:

$$\hat{P}_S(w_N \mid w_1^{N-1}) = \frac{c(w_1^N)}{c(w_1^{N-1})},$$

   $c(w_1^N)$ — это число последовательностей $w_1, \ldots, w_N$ в корпусе

2. Сэмплируем из полученных эмпирических распределений.
3. Для многих вероятностей даже при небольших n значения могут быть нулевыми.
4. Увеличение обучающего корпуса, сглаживание частот, откат.

Борьба с нулями

- *Add-one smoothing* (сглаживание Лапласа):

$$\hat{P}_{AOS}(w_N \mid w_1^{N-1}) = \frac{c(w_1^N) + \delta}{c(w_1^{N-1}) + \delta V},$$

V — это размер словаря, а δ — некоторая фиксированная константа.

Чем плох такой подход?

- *Katz smoothing* (простой откат): если не получается применить модель высокого порядка, пробуем для данного слова модель меньшего порядка с понижающим множителем.

Получим не вероятностное распределение!

Борьба с нулями

- *Jelinek-Mercer smoothing* (интерполяционное сглаживание): заведем вектор $\bar{\lambda} = (\lambda_1, \ldots, \lambda_N)$, такой, что $\sum_i \lambda_i = 1$ и $\lambda_i \geqslant 0$. Тогда

$$\hat{P}_{IS}(w_N \mid w_1^{N-1}) = \sum_{i=1}^{N} \lambda_i \hat{P}_S(w_N \mid w_{N-i+1}^{N-1}).$$

- Другие виды сглаживаний:
  - *Good-Turing estimate*
  - *Witten-Bell smoothing*
  - *Absolute discounting*
  - *Kneser-Ney smoothing*

Рекуррентные нейронные сети

- Обычные нейронные сети плохо подходят для обработки последовательностей, поскольку наблюдают только текущий элемент
- Для учёта контекста используются *рекуррентные нейронные сети* (RNN)

Примеры задач:

- Распознавание речи/музыки
- Распознавание рукописного текста
- Распознавание/генерация печатного текста
- Анализ временных рядов
- Машинный перевод

Ничего не напоминает?

Посмотрим на примере задачи генерации стихотворений (источник)

Задача генерации текста

- В узкой тематике бот может обмануть обычного человека, но не специалиста.

- В широкой и простой тематике выявить хорошего бота можно только по шаблонам в предложениях и явному комбинированию слов:

  Все ваши посты – типичное клише лживой инсинуации, которая стремится дискредитировать и осмеять всякого, кто начинает прозревать и открыто говорить о преступлениях преступного режима. Колет глаза держимордам кровавого кремлёвского упыря правда об их бесчеловечии и о фашистской сути кровавого кремлёвского режима! Интересной особенностью данного форума является то, что путинисты в основном занимаются флудом или обсуждением личностей, а топиков по существу проблем России, вроде этого, боятся как черт ладана.

Генерация текстов Шекспира

Художественную прозу можно генерировать и HMM:

- **Униграммная модель:**

  To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have. Every enter now severally so, let. Hill he late speaks; or! a more to leg less first you enter.

- **Биграммная модель:**

  What means, sir. I confess she? then all sorts, he is trim, captain. Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

Генерация текстов Шекспира

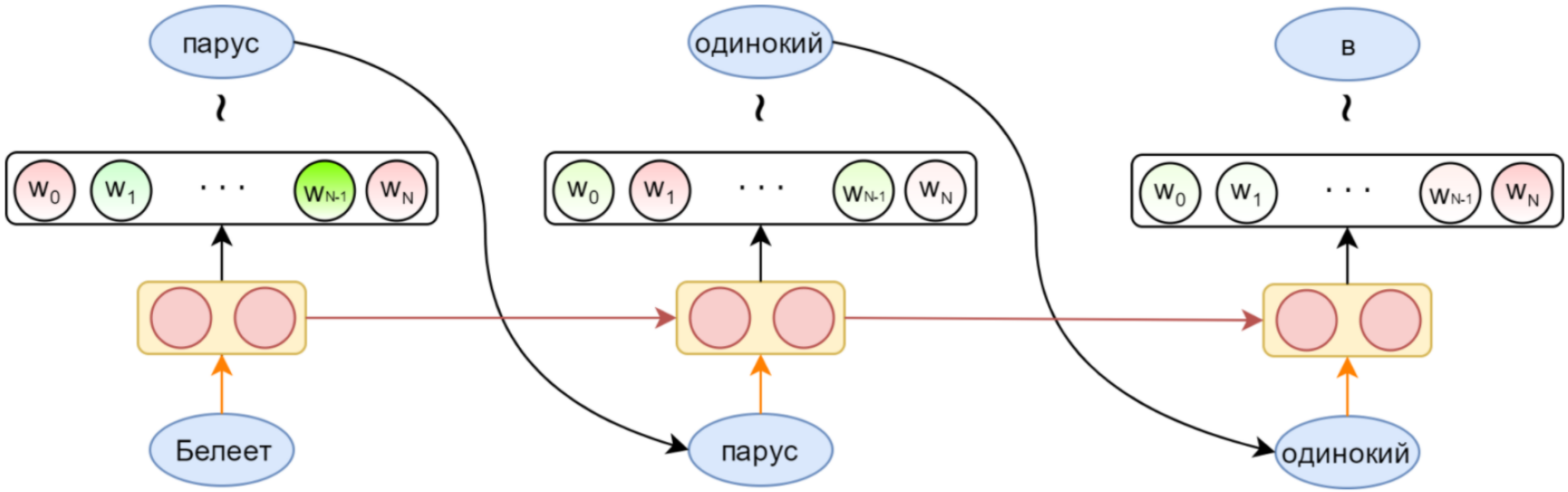
- **3-граммная модель:**

  Sweet prince, Falstaff shall die. Harry of Monmouth's grave. This shall forbid it should be branded, if renown made it empty. What is't that cried? Indeed the duke; and had a very good friend. Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

- **4-граммная модель:**

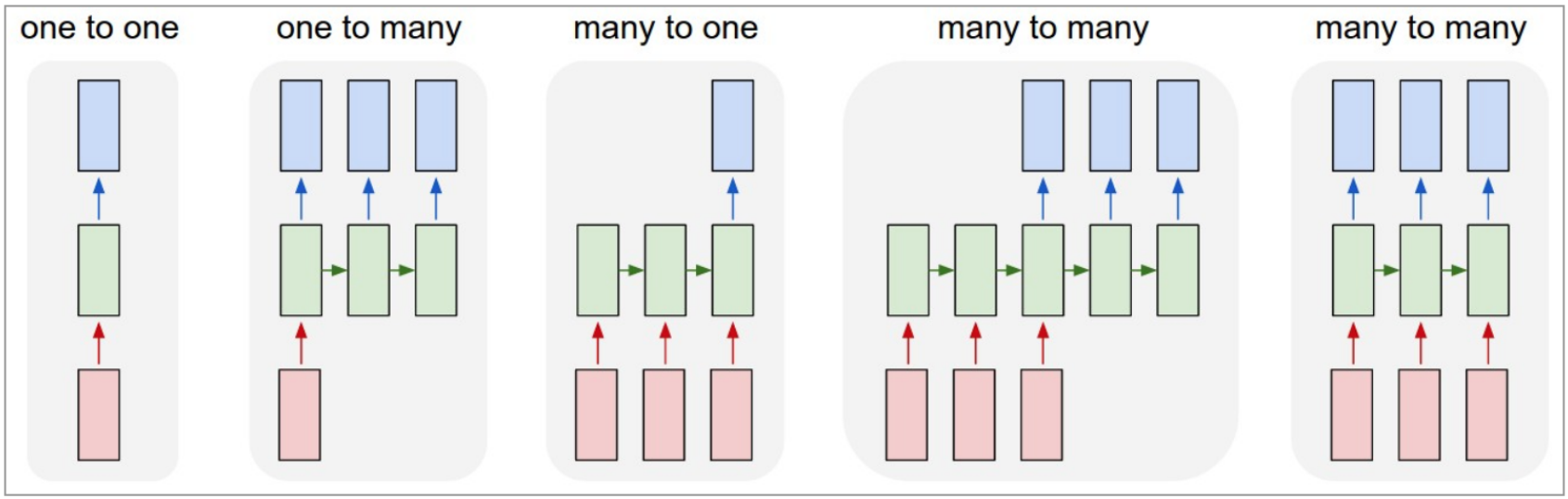
  King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; Will you not tell me who I am? It cannot be but so. Indeed the short and the long. Marry, 'tis a noble Lepidus. They say all lovers swear more performance than they are wont to keep obliged faith unforfeited.

Рекуррентные нейронные сети



- На входе – embeding слова (рыжая стрелка)
- На выходе обычно полносвязный слой + softmax (берём argmax или сэмплируем)
- Результаты предыдущих итераций и информация с прошлого прохода передаются дальше.

Виды RNN

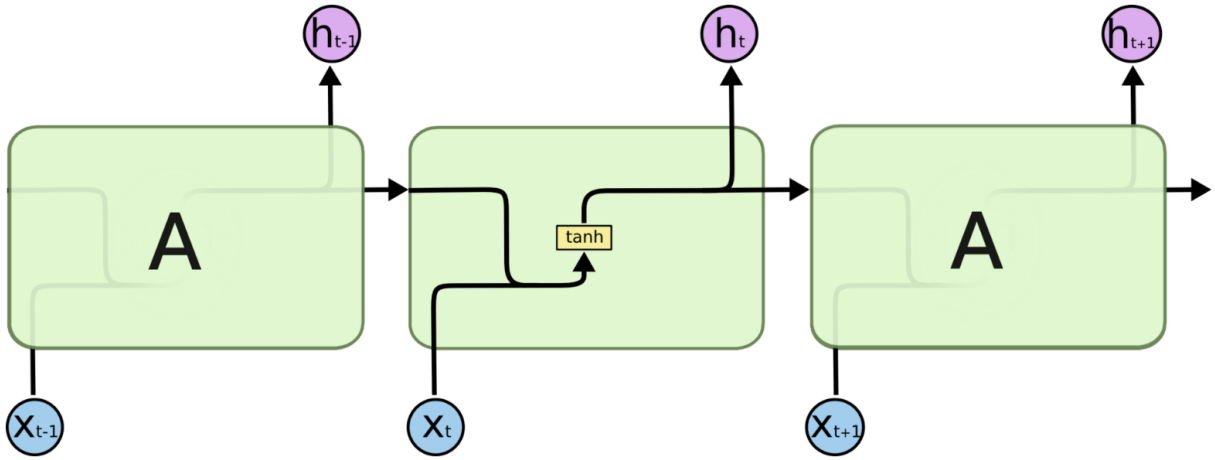


Ссылка на источник картинки

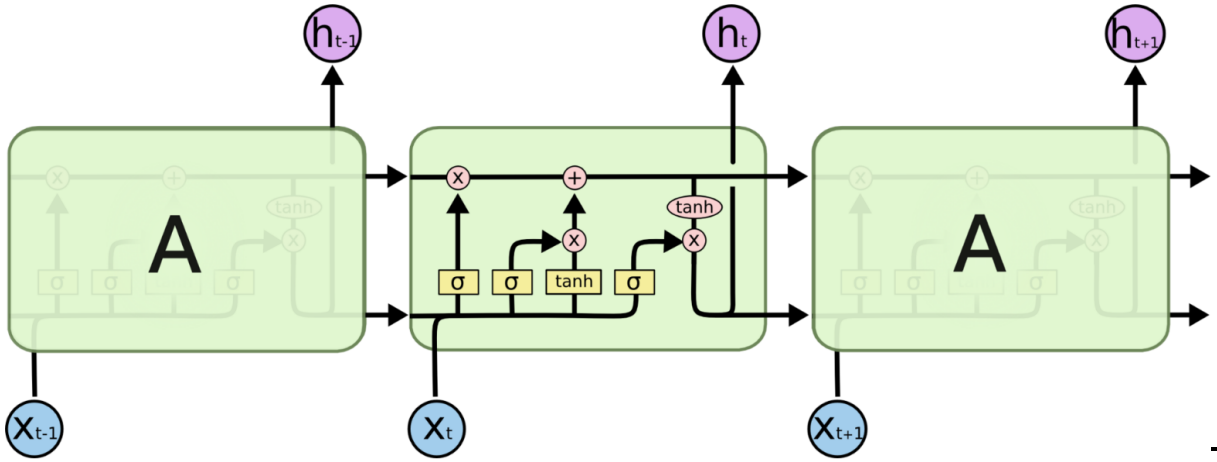
Архитектура LSTM

- В реальности обычная RNN хранит информацию только о коротком контексте (затухание градиентов)
- Такого недостатка лишена LSTM – нейросетевой рекуррентный блок, состоящий из пяти элементов:
  - Основной слой (как и в обычной RNN)
  - Три сигмоидальных слоя-фильтра
  - Ячейка памяти (вектор)
- Каждый слой имеет свои обучаемые веса
- Каждый фрейм LSTM передаёт не только свои выходы, но и состояние ячейки памяти

Архитектура LSTM



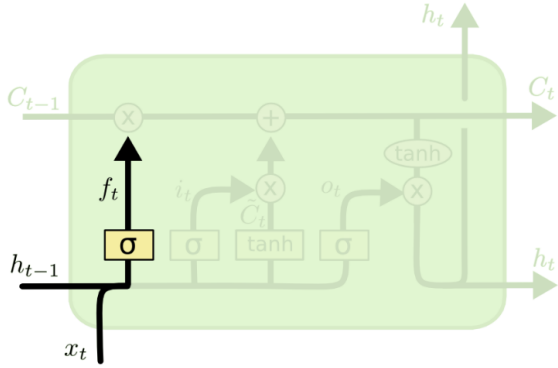
– RNN: 1 слой



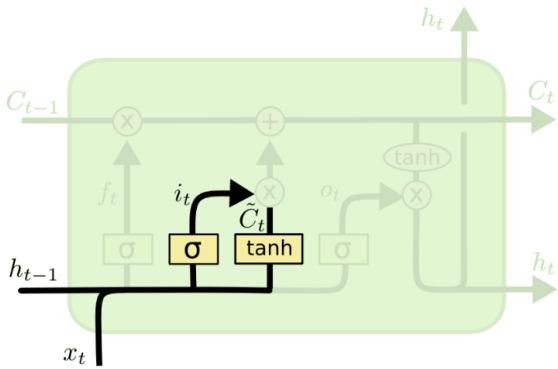
– LSTM: 4 слоя

Ссылка на источник картинок

Архитектура LSTM

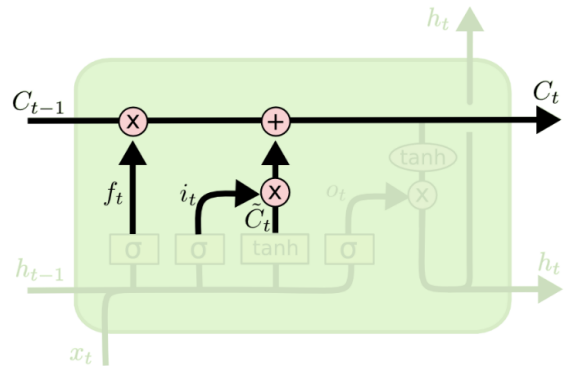


1) $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$

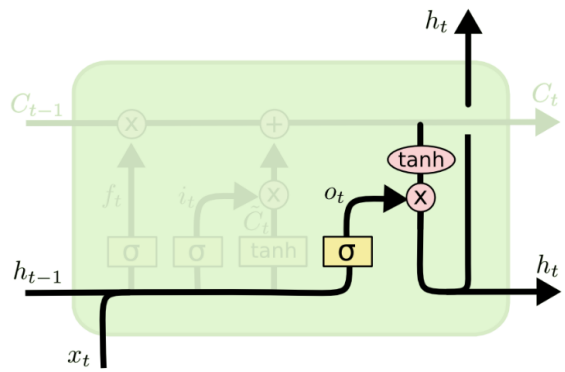


2) $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$

$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$



3) $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

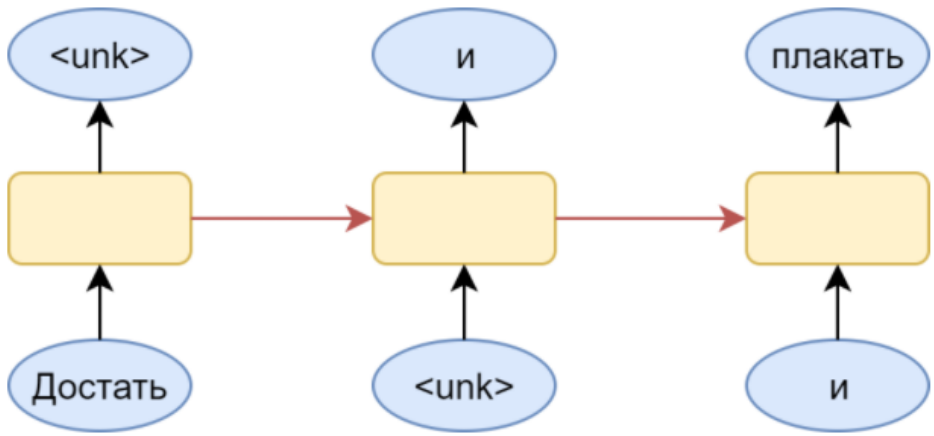


4) $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$

$h_t = o_t * \tanh(C_t)$

Генерация стихов: обработка отсутствующих слов

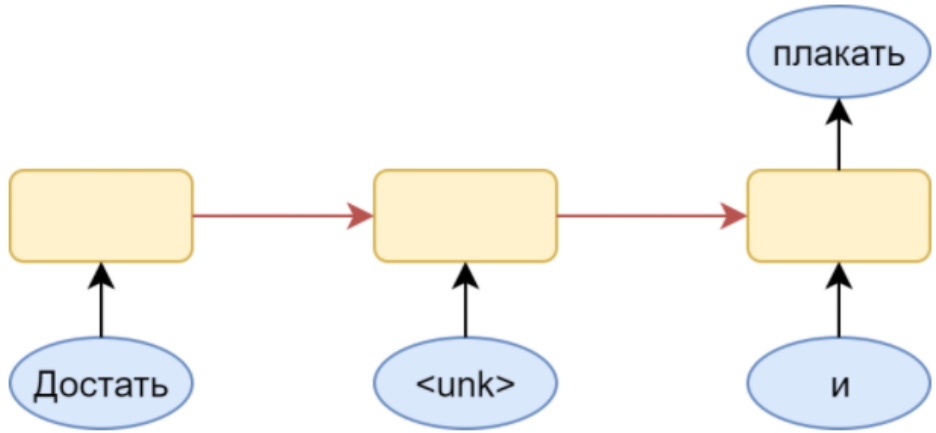
- Словарь может состоять из миллионов слов, но часто его приходится сильно фильтровать
- Вместо отсутствующего слова берём <unk>:



- В модели предсказания слова по предыдущему выдаваемое распределение на словах сместится в пользу <unk>
- **Выход:** можно сэмплировать без него, но получается криво

Обработка отсутствующих слов

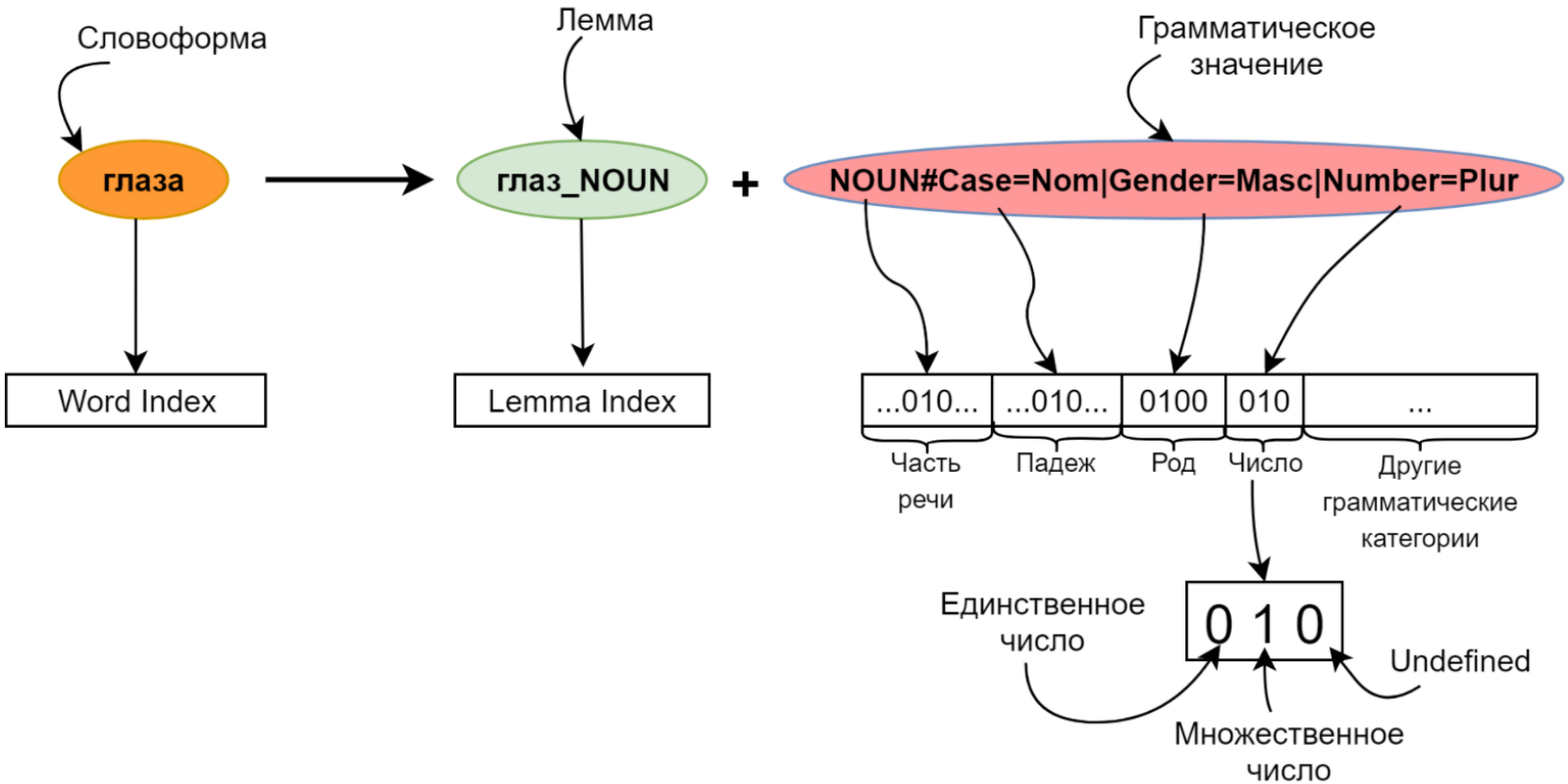
- Альтернатива – предсказывать слова по цепочке предыдущих:



- Из обучающей выборки придётся нарезать всевозможные цепочки, что приведёт к её существенному увеличению
- Зато можно выкинуть все цепочки, заканчивающиеся неизвестным словом.

Доработка входного слоя

Необходимо сократить размерность выходного слоя

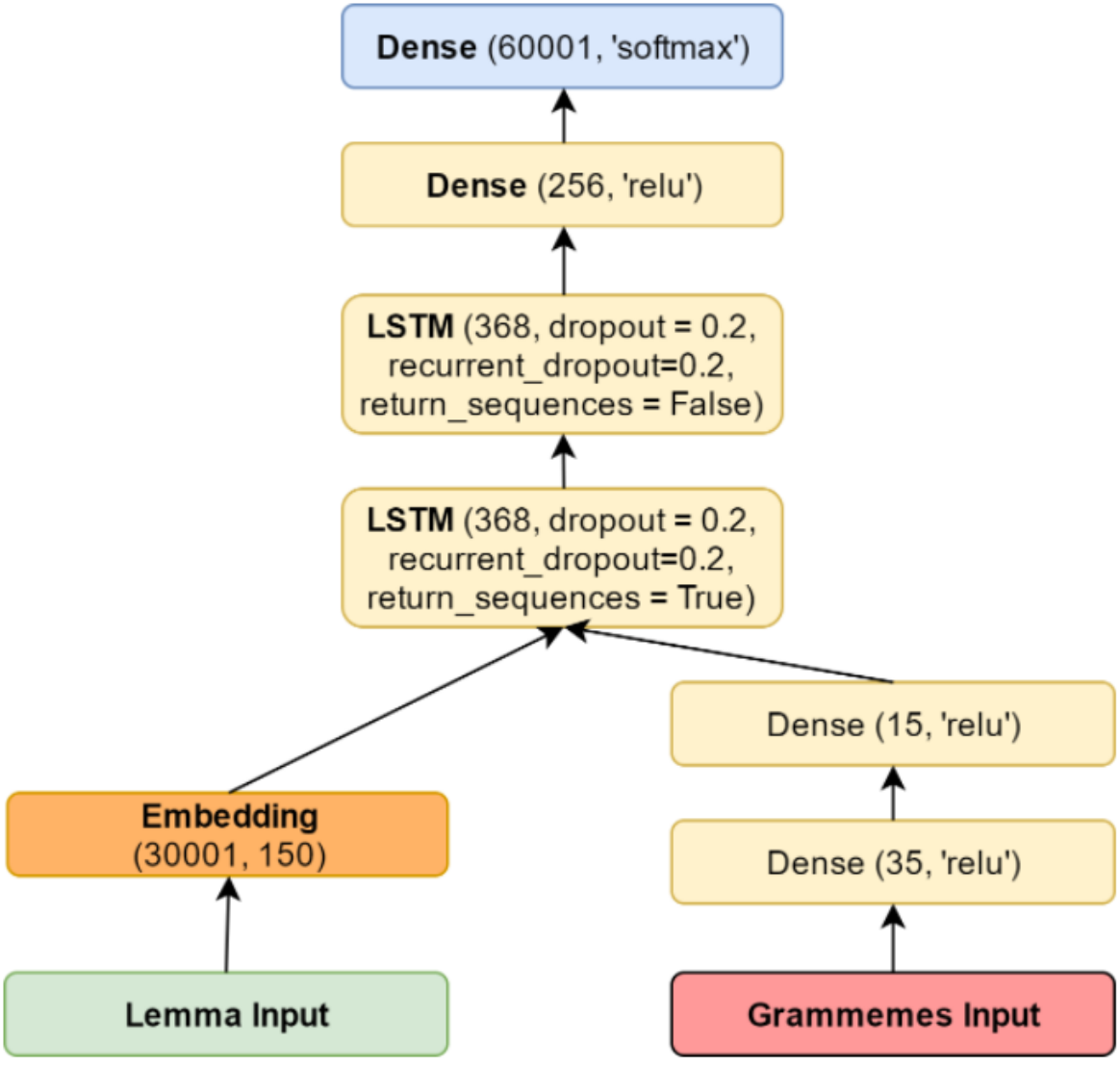


Можно использовать уже предобученные эмбеддинги для лемм (например, от RusVectores).

Доработка выходного слоя

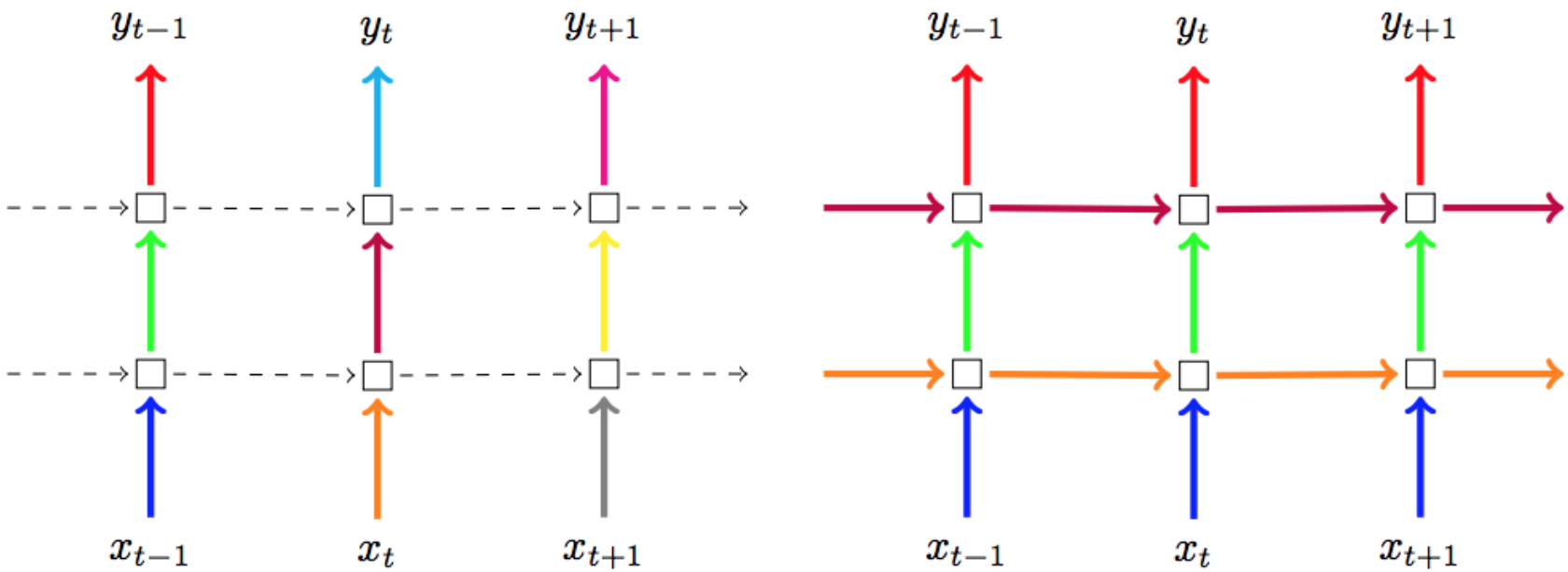
- Вместо индекса слова можно предсказывать по-отдельности лемму и грамматическое значение
- **Проблема:** у сэмплированной леммы может не оказаться нужного грамматического значения
- **Варианты решения:**
  1. выбирать наиболее вероятную пару «лемма + грамматическое значение» из существующих
  2. выбирать наиболее вероятное грамматическое значение среди возможных для сэмплированной леммы

Итоговая архитектура сети (keras)



Рекуррентный dropout

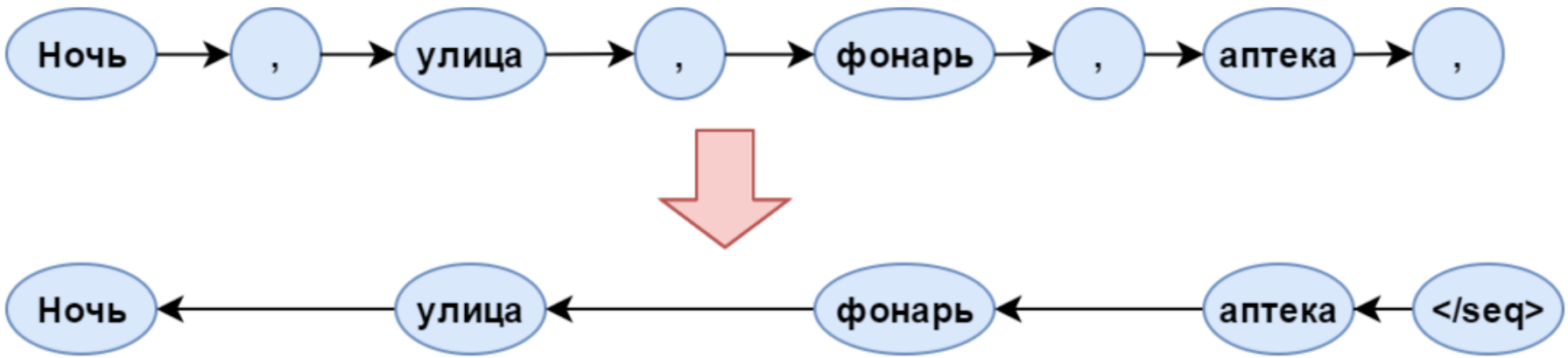
- Обычный дропаут применяется к весам, связанным с входными данными
- Можно применять дропаут и к весам, связанным с выходами предыдущего фрейма h_{t-1}



Ссылка на источник картинки

Данные

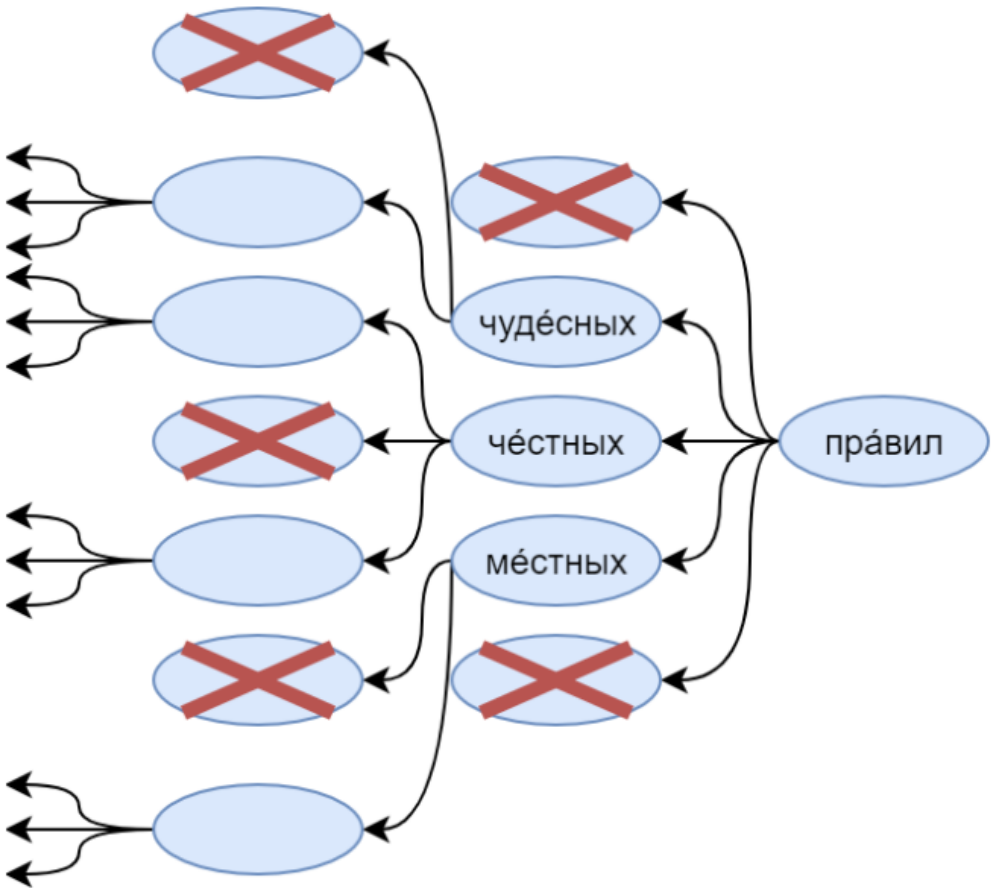
- http://stihi.ru/ + морфологическая разметка.
- Объект выборки – строка стихотворения.
- В конец каждой строки добавлялся завершающий символ.
- Строки инвертировались для упрощения рифмовки при генерации.
- Из выборки удалены знаки препинания (сеть сильно обучается на запятых и многоточиях).



Правила фильтрации

- У нас есть модель-генератор, нужно фильтровать слова так, чтобы получались именно стихотворения
- Метрические правила определяют последовательность ударных и безударных слогов в строке
- Правила рифмы допускают только словоформы, которые корректно рифмуются (слова с одной леммой рифмовать запрещено)
- Ударения были получены путём обучения классификатора на словаре, рифмы – эвристическими правилами

Лучевой поиск (beam search)



- В результате работы фильтров могло не остаться ни одного слова
- Для борьбы с этим на каждом этапе применения фильтров берём не лучшего, а несколько лучших кандидатов, так, чтобы на каждом шаге их было N штук

Пример результата

Так толку мне теперь грустить

Что будет это прожито

Не суждено кружить в пути

Почувствовав боль бомжика