



ТЕХНОСФЕРА

Лекция Языковые модели

Владимир Гулин

7 ноября 2019 г.

Языковая модель - это вероятностное распределение на множестве словарных последовательностей

Языковые модели

Языковая модель приписывает вероятность фрагменту текста (высказыванию, предложению...)

В хорошей модели вероятности языковых фрагментов соответствуют их относительной частотности в текстах

Иными словами

- ▶ максимизирует вероятность реальных текстов
- ▶ минимизирует вероятность нереальных текстов

Где это требуется?

- ▶ Предсказание следующей лингвистической единицы (буквы, слова)
- ▶ Определение языка
- ▶ Исправление опечаток
- ▶ Снятие неоднозначностей разбора
- ▶ Машинный перевод
- ▶ Распознавание речи
- ▶ Генерация текстов
- ▶ Ранжирование результатов поиска
- ▶ и т.д.

Вероятности предложений: Интуиция

Probability of a sentence = насколько вероятно встретить его
в естественном языке

$P(\text{"Иван Грозный википедия"}) > P(\text{"Иван Грозный фото"})$
 $P(\text{"Анекдот про порутчика Ржевского"}) > P(\text{"Телефон порутчика Ржевского"})$

Language models in NLP

- ▶ В реальности крайне сложно узнать истинную вероятность заданной последовательности слов
- ▶ Однако мы можем использовать языковые модели, которые дают нам неплохую аппроксимацию
- ▶ Как и любые модели, языковые модели будут хороши в одних задачах и неприменимы в других

N-gram Language Models

Вероятность языковых событий

- ▶ Вероятность основана на подсчете частотности событий
- ▶ Обычно считаем по заданному корпусу
- ▶ вероятность = относительная частотность

Пример расчета

Всего слов в корпусе = 411165

sunday = 17

$$P(sunday) = \frac{17}{411165} = 0.00004$$

Maximum Likelihood Estimation, MLE

$$p(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{\sum_w C(w_{n-1} w)}$$

$$p(w_n | w_{n-1}) = \frac{C(w_{n-1} w_n)}{C(w_{n-1})}$$

Вопрос

Что делать с предложениями, которые не встречались никогда в заданном корпусе?

Предложениями, которые не встречались никогда

the Archaeopteryx soared jaggedly amidst foliage

vs

jaggedly trees the on flew

- ▶ Первое осмысленное, второе нет
- ▶ $C(S) = 0$ в обоих случаях

Sparse data problem

- ▶ Не достаточно данных для корректной оценки вероятностей (слишком большое признаковое пространство)
- ▶ Большинство предложений можно встретить либо слишком редко, либо не встретить вообще, поэтому надо что-то думать :)

Как победить проблему?

Идея:

Будем оценивать вероятности предложений $P(S)$ комбинируя вероятности меньших частей предложения, которые встречаются чаще

Таким образом, получили:

N-граммные языковые модели

Вывод N-граммной модели

- ▶ Хотим оценить $P(s = w_1 \dots w_n)$

Пример: $P(s = \text{иван грозный фото в душе})$



- ▶ По факту это совместное распределение слов в s :

$$P(w_1 = \text{иван}, w_2 = \text{грозный}, w_3 = \text{фото}, w_4 = \text{в}, w_5 = \text{душе})$$

- ▶ Вспоминаем, что для совместного распределения верно $P(X, Y) = P(Y|X)P(X)$, тогда:

$$\begin{aligned} P(\text{иван грозный фото в душе}) &= P(\text{душе}|\text{иван грозный фото в}) \cdot P(\text{иван грозный фото в}) = \\ &= P(\text{душе}|\text{иван грозный фото в}) \cdot P(\text{в}|\text{иван грозный фото}) \cdot P(\text{фото}|\text{иван грозный}) \cdot \\ &\quad \cdot P(\text{грозный}|\text{иван}) \cdot P(\text{иван}) \end{aligned}$$

Вывод N-граммной модели

- ▶ Chain rule дает нам:

$$P(w_1, \dots, w_n) = \prod_{i=1}^n P(w_i | w_1 \dots w_{i-1})$$

Вопрос:

В чем тут подвох?

Вывод N-граммной модели

- ▶ Chain rule дает нам:

$$P(w_1, \dots, w_n) = \prod_{i=1}^n P(w_i | w_1 \dots w_{i-1})$$

- ▶ Многие из этих условных вероятностей попрежнему sparse!

Если мы пытаемся оценить $P(\text{иван грозный фото в душе})$,
то нам нужно знать $P(\text{душе} | \text{иван грозный фото в})$

Вывод N-граммной модели

- ▶ Сделаем предположение, что вероятность слова зависит только от конечного количества предыдущих слов

Марковское свойство:

$$P(w_i | w_1, w_2, \dots, w_{i-1}) = P(w_i | w_{i-n+1}, \dots, w_{i-1})$$

- ▶ trigram model: $P(w_i | w_0, w_1, \dots, w_{i-1}) \approx P(w_i | w_{i-2}, w_{i-1})$
- ▶ bigram model: $P(w_i | w_0, w_1, \dots, w_{i-1}) \approx P(w_i | w_{i-1})$
- ▶ unigram model: $P(w_i | w_0, w_1, \dots, w_{i-1}) \approx P(w_i)$

Вывод N-граммной модели

$$\begin{aligned} P(w_i | w_1, w_2, \dots, w_{i-1}) &= P(w_i | w_{i-n+1}, \dots, w_{i-1}) = \\ &= \frac{P(w_i, w_{i-1}, \dots, w_{i-n+1})}{P(w_{i-1}, \dots, w_{i-n+1})} \approx \\ &\approx \frac{\text{Count}(w_i, w_{i-1}, \dots, w_{i-n+1})}{\text{Count}(w_{i-1}, \dots, w_{i-n+1})} \end{aligned}$$

Проблемы N-граммных моделей

$$\begin{aligned} P(w_i | w_1, w_2, \dots, w_{i-1}) &= P(w_i | w_{i-n+1}, \dots, w_{i-1}) = \\ &= \frac{P(w_i, w_{i-1}, \dots, w_{i-n+1})}{P(w_{i-1}, \dots, w_{i-n+1})} \approx \\ &\approx \frac{\text{Count}(w_i, w_{i-1}, \dots, w_{i-n+1})}{\text{Count}(w_{i-1}, \dots, w_{i-n+1})} \end{aligned}$$

Какие в этой формуле проблемы?

Backoff (aka "stupid backoff")

- ▶ Иногда помогает использование меньшего контекста

"иван грозный фото в" не встречалось → пробуем "иван грозный фото"

$$P(\text{душе}|\text{иван грозный фото в}) \approx P(\text{душе}|\text{иван грозный фото})$$

"иван грозный фото" не встречалось → пробуем "иван грозный"

$$P(\text{душе}|\text{иван грозный фото}) \approx P(\text{душе}|\text{иван грозный})$$

"иван грозный" не встречалось → пробуем "иван"

$$P(\text{душе}|\text{иван грозный}) \approx P(\text{душе}|\text{иван})$$

Backoff

- ▶ Если есть достаточно статистики, то используем триграммы
- ▶ Иначе биграммы
- ▶ Иначе униграммы

Более интеллектуальный подход: Линейная интерполяция

Сделаем линейную комбинацию униграмм, биграмм, триграмм и т.д.

$$P(w_i | w_1, w_2, \dots, w_{i-1}) \approx \lambda_3 P(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P(w_i | w_{i-1}) + \lambda_1 P(w_i)$$

$$\sum_{k=0}^{n-1} \lambda_k = 1$$

Более интеллектуальный подход: Линейная интерполяция

Сделаем линейную комбинацию униграмм, биграмм, триграмм и т.д.

$$P(w_i | w_1, w_2, \dots, w_{i-1}) \approx \lambda_3 P(w_i | w_{i-2}, w_{i-1}) + \lambda_2 P(w_i | w_{i-1}) + \lambda_1 P(w_i)$$

$$\sum_{k=0}^{n-1} \lambda_k = 1$$

Вопрос:

Как выбрать λ_k ?

Как оценивать языковые модели

Оценивание на промежуточных подзадачах

- ▶ Perplexity
- ▶ Cross entropy

Оценивание на реальных задачах

- ▶ Встраиваем языковую модель в реальную боевую систему
- ▶ Учим модель системы с разными языковыми моделями
- ▶ Если итоговое качество лучше, то успех!

Cross-entropy and Perplexity

- ▶ Для $(w_1 w_2 \dots w_n)$ cross entropy определяется как

$$H_M(w_1 w_2 \dots w_n) = \frac{1}{n} \cdot \log P_M(w_1 w_2 \dots w_n)$$

- ▶ Чем меньше cross entropy, тем лучше модель предсказывает следующее слово
- ▶ Perplexity (часто можно встретить в статьях):

$$\textit{Perplexity} = 2^{\textit{cross-entropy}}$$

Проблемы N-граммных моделей

$$\begin{aligned} P(w_i | w_1, w_2, \dots, w_{i-1}) &= P(w_i | w_{i-n+1}, \dots, w_{i-1}) = \\ &= \frac{P(w_i, w_{i-1}, \dots, w_{i-n+1})}{P(w_{i-1}, \dots, w_{i-n+1})} \approx \\ &\approx \frac{\text{Count}(w_i, w_{i-1}, \dots, w_{i-n+1})}{\text{Count}(w_{i-1}, \dots, w_{i-n+1})} \end{aligned}$$

А что если числитель равен 0?

Laplace smoothing

- ▶ Сделаем вид, что мы видели каждое слово на один раз больше, чем на самом деле
- ▶ Добавим ко всем статистикам по единице

Если 1 слишком грубое приближение, то можно взять небольшую константу δ для каждого слова $w_i \in V$.

$$P(w_i | w_{i-n+1}, \dots, w_{i-1}) = \frac{\delta + P(w_i, w_{i-1}, \dots, w_{i-n+1})}{\delta \cdot |V| + P(w_{i-1}, \dots, w_{i-n+1})}$$

Good-Turing

Пусть n_r - число n -грамм, повторяющихся ровно r раз.

Сдвинем частоты:

$$r^* = (r + 1) \frac{n_{r+1}}{n_r}$$

и положим

$$p_{GT}(w_{i-n+1}^i) = \frac{r^*}{N},$$

где $N = \sum_{r=0}^{\infty} n_r r^*$

Kneser-Ney smoothing

Идея:

Модель более низкого порядка имеет смысл пользоваться только когда статистика по модели высокого порядка равна 0 или незначительна

- ▶ Пример: Рассмотрим "San Francisco" и положим, что "Francisco" встречается только после "San"
- ▶ "Francisco" будет иметь высокую униграммную вероятность, после каждой новой биграммы
- ▶ Лучше дать "Francisco" низкую униграммную вероятность, потому что он встречается только после "San", тогда биграммная модель будет адекватной

Kneser-Ney smoothing

- ▶ Пускай счетчик для каждой униграммы определяется как число различных слов, которые идут после него:

$$N_{1+}(\bullet w_i) = \|\mathbf{w}_{i-1} : c(\mathbf{w}_{i-1} w_i) > 0\|$$

$$N_{1+}(\bullet\bullet) = \sum_{w_i} N_{1+}(\bullet w_i)$$

- ▶ Распределение меньшего порядка:

$$p_{KN}(w_i) = \frac{N_{1+}(\bullet w_i)}{N_{1+}(\bullet\bullet)}$$

- ▶ Объединяем вместе:

$$p_{KN}(w_i | w_{i-n+1}^{i-1}) = \frac{\max\{c(w_{i-n+1}^{i-1}) - \delta, 0\}}{\sum_{w_i} c(w_{i-n+1}^i)} + \\ + \frac{\delta}{\sum_{w_i} c(w_{i-n+1}^i)} N_{1+}(\bullet w_{i-n+1}^{i-1}) p_{KN}(w_i | w_{i-n+2}^{i-1})$$

Вопросы

