[illegible]

① Word2Vec - CBOW и Skip-gram

Continuous bag of words

Хотим представить слово вектором фикс. длины, например, 256. И чтобы у похожих слов были похожие вектора.

Мы не можем использовать граф, т.к. ~~[illegible]~~ тогда вектор будет слишком огромным, а нам надо длина 256. И one-hot encoding тоже плохо - огромная размерность, не учитывает близость слов.

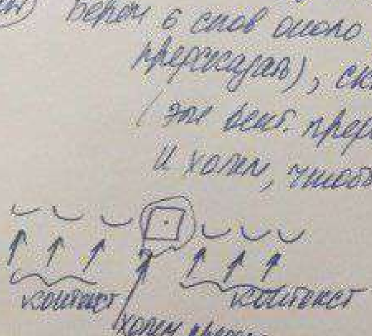
Поэтому правильный способ делать embedding слова - это word2vec.

Есть 2 варианта: CBOW и Skip-gram.

Раньше считалось, что CBOW работает хорошо только для маленьких текстов, а skip-gram, хоть и ~~[illegible]~~ дольше учится - но работает лучше - но потом нашли в библиотеке свои маленькие баги, исправили, и теперь они одинаково хорошо работают.

Матрицу E - эмбеддинга будем обучать с помощью нейросети.

CBOW Берем k слов около интересующей нас дырки (слова, которое хотим предсказать), складываем вект. представления этих k слов-соседей (эти вект. представления сначала случайно инициализированы) и хотим, чтобы получилось примерно среднее слово.



Т.е. у k слов считаем сумму, получаем вектор, считаем его скал. произведение со всеми синими векторами (тоже сначала случайными), ~~и хотим, чтобы пол~~ применяем softmax, и хотим, чтобы полученное распределение было похоже на 0...010...0 - т.е. с этим считаем лосс: $L = -\log p(w_i \mid context(w_i))$

(где $p(w_i \mid context(w_i)) = softmax(W'h)[w_i]$);

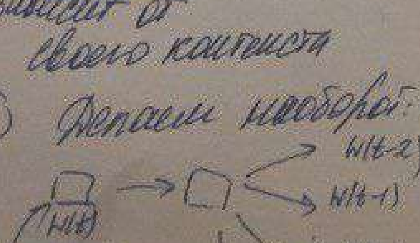
$$h_t = W \sum_{w \in context(w_t)} one\text{-}hot(w)$$

И синие, и красные вектора - обучаем градиентным спуском.

синие - как слово ~~[illegible]~~ зависит от своего контекста

красные - как слово участвует в контексте

Skip-grams Делаем наоборот: пытаемся по слову предсказать его контекст.

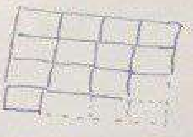


← хотим предсказать распределение красных векторов-элементов контекста.

② Можно ли использовать ReLU в ~~[illegible]~~ качестве 1-го слоя декодера состязательного автокодировщика?

Ответ: лучше не надо, т.к. ReLU просто занулит отрицательные компоненты эмбеддинга, и нечего учиться не будет. Лучше использовать LeakyReLU, т.к. она сохраняет отрицательные значения:

④

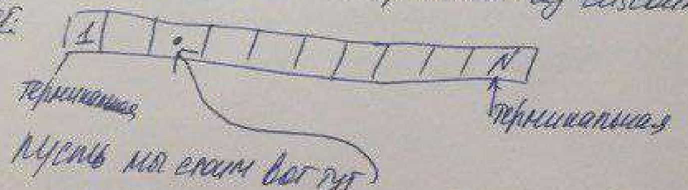


Ответ:
Без pack-padded-sequence LSTM-ячейка будет применена 4·4=16 раз (т.е. ко всем ячейкам, даже к ненужным)

А с pack-padded-sequence LSTM-ячейка будет применена 4+3+3+2=12 раз (т.к. к ненужным ячейкам мы её не применяем, для этого и нужна упаковка)

⑥ Придумал MDP, в котором опт. стратегия с discount=0.9 отличается от оптимальной стратегии без discount.

Ответ:



Если discount factor = 0 ⇒ идем влево
Если discount factor = 0.9 ⇒ идем вправо } – разные оптимальные стратегии.

③ Оценка для $\frac{\partial}{\partial\theta}\int_{\mathbb{R}} f(x,\theta)p(x|\theta)dx$; $p(x|\theta)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\mu(\theta))^2}{2}} \sim N(\mu(\theta);1)$

Ответ:

$$\frac{\partial}{\partial\theta}\int_{\mathbb{R}} f(x,\theta)p(x|\theta)dx=\int_{\mathbb{R}}\underbrace{\frac{\partial f}{\partial\theta}}_{\text{это то, что дает backprop}}(x,\theta)p(x|\theta)dx+\int_{\mathbb{R}}\underbrace{f(x,\theta)}_{\text{нейросеть}}\underbrace{(\mu(\theta)-x)\mu'(\theta)}_{\text{вычисляется напрямую}}p(x|\theta)dx$$

А по плотности $\int F(x)p(x)dx$ просто приближаем так: $\frac{1}{n}\sum_{i=1}^{n}F(x_i)$, где $x_i\sim p(x)$ (т.е. сэмплируем n точек из $N(\mu(\theta);1)$)

Плюсы оценки: Она быстро считается, т.к. чтобы получить $\frac{\partial f}{\partial\theta}$, достаточно одного прохода по графу вычислений.

Минусы оценки: предполагаем дифференцируемость μ и применимость теоремы о занесении производной под интеграл.

5 Для каких значений c не изменится опт. стратегия при замене ф-ции вознаграждения R на а) $c\cdot R$; б) $c+R$; в) R^c

ответ:

~~[illegible]~~ опт. стратегия: $\pi^*(s) = \arg\max_a Q^*(s,a)$, где $Q^*(s,a) = \max_\pi Q^\pi(s,a)$.

Ур-я Беллмана: $v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s]$

$q_\pi(s,a) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$.

а) Если $R \rightsquigarrow c\cdot R$, то $\forall c>0$: π^* – не изменится, т.к. знак сохранится, а c прост вынесется за знак матожидания.

(а $c<0$ нам не подходят, т.к. стратегия обратится: самой большой R станет самым маленьким)

б) Если $R \rightsquigarrow c+R$: подходит только $c=0$, т.к. (пусть discount factor $=1$), если $c>0$ – то мы будем предпочитать более длинные пути, а если $c<0$ – то более короткие)

в) Если $R \rightsquigarrow R^c$ ~~если c нечетное натур. число, то $\max R^c$ дает лучшую π^* что и $\max R$, если $R<0$. (а если $R>0$, то max~~

подходит $c=1$, ~~А если $R>0$, то $\max_a R^c$ дает лучшую π^*, что и $\max R$ при $c>1$.~~

т.к. при $c<0$ – получим некорректную стратегию

при $c>1$ получим строго выпуклую ф-цию (т.е. предпочитаем более длинные пути)

при $c \in (0;1)$ получим строго вогнутую ф-цию (т.е. предпочитаем более короткие пути)

при $c=0$: получится константа 1, что изменит стратегию