Natural Language Processing

Зинина Анастасия

Bag-of-words

- Рассмотрим пример. Есть два текста.
  (1) Buy a ticket and win special prizes! The number of prizes is limited.
  (2) Ann will buy a ticket to the theatre for her sister.
- Составим словарь из всех имеющихся слов.
  {'a','and', 'ann', 'buy', 'for', 'her','is','limited', 'number','of','prizes', 'sister', 'special', 'ticket', 'theatre', 'the', 'to', 'will' }
- Каждый текст представим в виде вектора, размерность которого равна количеству слов в словаре.
  (1) [1,1,0,1,0,0,1,1,1,1,2,0,1,1,0,1,0,0]
  (2) [1,0,1,1,1,1,0,0,0,0,0,1,0,1,1,1,1,1]
- Можем удалить из словаря так называемые стоп-слова: 'a','the','to'
- Теперь применяем алгоритмы классификации к полученным векторам признаков.

N-gram model

- В BOW не учитывается порядок слов
- n-gram позволяет учесть грамматические особенности
- Bigram:
  {'Buy ticket', 'ticket win', 'win special','special prizes' etc.}

TF-IDF

- Term Frequency - частота слова в документе

$$TF = \frac{\text{Количество раз, когда слово встретилсось в тексте}}{\text{количество всех слов в тексте}}$$

- Inverse Document Frequency. Некоторые слова встречаются в текстах любой тематики, а термины лишь в специальных текстах. Учтём распространённость слова.

$$IDF = \ln \frac{\text{Общее количество документов}}{\text{Количество документов, в которых встречается слово}}$$

- Модификация IDF

$$IDF = \ln(1 + \frac{\text{Общее количество документов}}{\text{Количество документов, в которых встречается слово}})$$

- $TF - IDF = (TF) * (IDF)$

Нейронные сети как универсальная модель аппроксимации

Нейронные сети принято считать универсальным методом решения задач регрессии и классификации. Такое восприятие связано со следующим утверждением.

Колмогоров,1957 Каждая непрерывная функция $a(x)$, заданная на единичном кубе n-мерного пространства,представима в виде

$$a(x) = \sum_{i=1}^{2d+1} \sigma_i (\sum_{j=1}^{d} f_{ij}(x_{ij})),$$

где $x = [x_1, ..., x_{x\,d}]^T$-вектор описания объекта, функции σ_i и $f_{ij}(\cdot)$ являются непрерывными функциями.

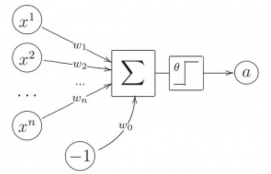
Однослойная нейронная сеть

Модель линейного порогового классификатора МакКаллока-Питтса

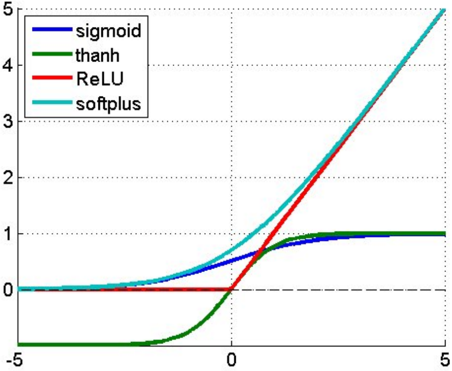
- Пусть все признаки $f_i(x)$ бинарные.
- Значения признаков-величины импульсов, поступающих на вход нейрона через n синапсов.
- Поступающие импульсы складываются с весами w_i
- Если суммарный импульс превышает порог активации, то нейрон выдаёт на выходе 1, иначе 0:

$$a(x) = \varphi(\sum_{j=1}^{n} w_j x^j - w_0),$$

где $\varphi(z) = I_{\{z \geqslant 0\}}$-функция активации.

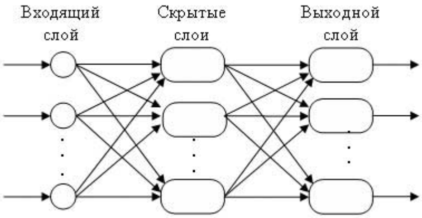


Позже модель была обобщена на случай произвольных вещественных входов и выходов, а также произвольных функций активации.



Двухслойная нейронная сеть определяется как линейная комбинация D нейронов:

$$a(x, w) = \sigma^{(2)}(\sum_{i=1}^{D} w_i^{(2)} \sigma^{(1)}(\sum_{j=1}^{d} w_{ji}^{(1)} x_j^{(1)} + w_{0i}^{(1)}) + w_0^{(2)})$$



Аналогично определяются сети с большим числом слоёв.

Оптимизация параметров нейронной сети

Оптимальные значения параметров определяются как:

$$w^* = \underset{w}{argmin} Q(w),$$

где Q-функция ошибки.

Методы оптимизации:

- стохастическая оптимизация(генетические алгоритмы, метод отжига, метод Нелдера-Мида);
- градиентные методы

Метод обратного распространения ошибок

Рассматриваем полносвязную сеть, $X = R^n, Y = R^M$.

- Выходной слой: M нейронов, функции активации σ_m, выходы a^m, m=1,...,M.
- Скрытый слой: H нейронов, функции активации σ_h, выходы u^h, h=1,...,H.
- Синаптические связи между h-м нейроном скрытого слоя и m-м нейроном выходного слоя обозначим w_{hm}
- Перед эти скрытым слоем находится либо распределительный, либо ещё один скрытый слой с выходами $v_j, j = 1, ..., J$ и синаптическими весами w_{jh}.

Выходные значения сети на объекте x_i определяются как суперпозиция:

$$a^m(x_i) = \sigma_m(\sum_{h=0}^{H} w_{hm} u^h(x_i)),$$

$$u^h(x_i) = \sigma_h(\sum_{j=0}^{J} w_{jh} v^j(x_i)).$$

- Функционал среднеквадратичной ошибки для объекта x_i:

$$Q(w) = 0.5 \sum_{m=1}^{M} (a^m(x_i) - y_i^m)^2.$$

- В дальнейшем для вычисления градиента нам понадобятся частные производные:

$$\frac{\partial Q(w)}{\partial a^m} = a^m(x_i) - y_i^m = \varepsilon_i^m - \text{ошибка на выходном слое},$$

$$\frac{\partial Q(w)}{\partial u^h} = \sum_{m=1}^{M} (a^m(x_i) - y_i^m) \sigma_m^{'} w_{hm} = \sum_{m=1}^{M} \varepsilon_i^m \sigma_m^{'} w_{hm} = \varepsilon_i^h - \text{ошибка на скрытом сло}$$

- ε_i^h вычисляются по ε_i^m, если запустить сеть "задом наперёд".
- Можем записать градиент Q:

$$\frac{\partial Q(w)}{\partial w_{hm}} = \frac{\partial Q(w)}{\partial a^m} \frac{\partial a^m}{\partial w_{hm}} = \varepsilon_i^m \sigma_m^{'} u^h,$$

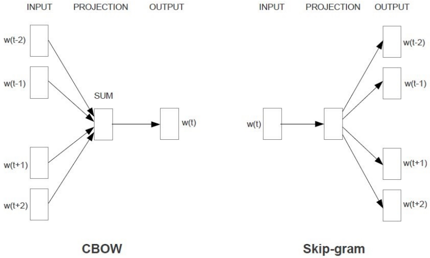
$$\frac{\partial Q(w)}{\partial w_{jh}} = \frac{\partial Q(w)}{\partial u^h} \frac{\partial u^h}{\partial w_{jh}} = \varepsilon_i^h \sigma_h^{'} v^j,$$

word2vec

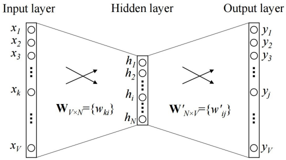
WORD2VEC — инструмент для расчета векторных представлений слов. Задачи:

- кластеризация слов;
- выявление семантической близости слов;
- анализ тональности.

word2vec использует 2 архитектуры: CBOW и Skip-gram



CBOW



- Предположим, что у нас есть одно слово контеста, по которому мы предсказываем целевое слово
- Размер входного слоя V, скрытого - N. Сеть полносвязная.
- На вход подаётся one-hot encoded вектор.
- Матрица весов между входным и скрытым слоями $W_{V\times N}$. i-ая строка v_w^T-векторное представление слова w, поступившего на вход. Полагая $x_k = 1$ и $x_l = 0$ для $l \neq k$, запишем

$$h = W^T x = W^T_{(k,\cdot)} := v^T_{w_I} \quad (1)$$

- Функция активации скрытого слоя - линейная.

- Матрица весов между входным и скрытым слоями $W'_{N\times V}$. i-ая строка v_w^T-векторное представление слова w, поступившего на вход. Вычислим значение u_j для каждого слова в словаре

$$u_j = {v'_{w_j}}^T h, \quad (2)$$

где ${v'_{w_j}}^T$-j-ый столбец W'

- Используем softmax для получения апостериорной вероятности слов:

$$p(w_j \mid w_I) = y_j = \frac{exp(u_j)}{\sum_{j'=1}^{V} exp(u_{j'})} \quad (3)$$

- Подставим (1) и (2) в (3):

$$p(w_j \mid w_I) = y_j = \frac{exp({v'}_{w_j}^T v_{w_I})}{\sum_{j'=1}^{V} exp({v'}_{w_{j'}}^T v_{w_I})} \quad (4)$$

Обновление весов между скрытым и выходным слоями

- Задача - максимизировать (4) - условную вероятность истинного выходного слова w_O (его индекс в выходном слове обозначим j^*) при данном контекстном слове w_I. Определим функцию потерь:

$$\log p(w_O \mid w_I) = \log y_{j^*} = u_{j^*} - \log \sum_{j'=1}^{V} exp(u_{j'}) := -E \to \min$$

- Нам понадобятся производные E по u_j:

$$\frac{\partial E}{\partial u_j} = y_j - t_j := e_j,$$

где $t_j = 1$ только при $j = j^*$.

- Теперь запишем производную по весам:

$$\frac{\partial E}{\partial w'_{ij}} = \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial w'_{ij}} = e_j h_i$$

- Шаг SGD:

$$w'^{(new)}_{ij} = w'^{(old)}_{ij} - \eta e_j h_i,$$

$$v'^{(new)}_{w_j} = v'^{(old)}_{w_j} - \eta e_j h \text{ для } j = 1, ..., V$$

Смысл градиентного шага

Посмотрим на смысл градиентного шага.

- v_w - векторное представление данного слова w на входе, $v_w^{'}$ -векторное представление данного слова w на выходе.
- Если $y_j > t_j$ ("переоценивание"), то мы вычитаем из $v_{w_j}^{'}$ величину, пропорциональную h $(= v_{w_I})$. Т.о. $v_{w_j}^{'}$ становится дальше от v_{w_I} (дальше в смысле скалярного произведения)
- Если $y_j < t_j$ (это возможно тогда и только тогда, когда мы нашли истинное слово, которое предсказываем $w_j = w_O$), то мы добавляем к $v_{w_O}^{'}$ величину, пропорциональную h. Т.о. $v_{w_O}^{'}$ становится ближе к v_{w_I}.
- Чем ближе y_j к t_j, тем меньше изменяются веса.
- Тогда оправдывается идея word2vec: если слова появляются в похожих контекстах (т.е. векторные представления слов контекста на входе близки), то эти слова семантически близки (векторные представления этих слов на выходе близки).

Обновление весов между входным и скрытым слоями

- Нам понадобятся производные E по h_i:

$$\frac{\partial E}{\partial h_i} = \sum_{j=1}^{V} \frac{\partial E}{\partial u_j} \frac{\partial u_j}{\partial h_i} = \sum_{j=1}^{V} e_j w'_{ij} := EH_i$$

где $t_j = 1$ только при $j = j^*$.

- Теперь запишем производную по W:

$$h_i = \sum_{k=1}^{V} x_k w_{ki}$$

$$\frac{\partial E}{\partial w_{ki}} = \frac{\partial E}{\partial h_i} \frac{\partial h_i}{\partial w_{ki}} = EH_i x_k$$

- Шаг SGD:

$$v'^{(new)}_{w_I} = v^{(old)}_{w_I} - \eta EH^T \quad (14)$$

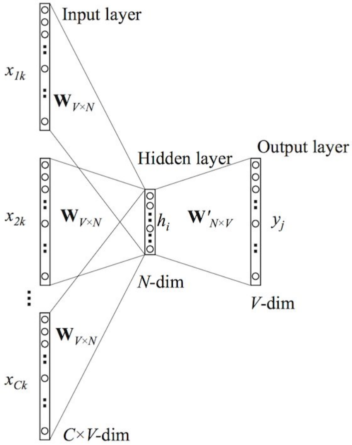
- Только одна компонента x не равна 0, следовательно есть только одна строка W, производная которой не равна 0.

Смысл градиентного шага

Посмотрим на смысл градиентного шага.

- v_w - векторное представление данного слова w на входе,
  v'_w -векторное представление данного слова w на выходе.
- Если $y_j > t_j$, то входной вектор слова контекста w_I отходит от выходного вектора w_j
- Если $y_j < t_j$, то входной вектор слова контекста w_I подходит ближе к выходному вектору w_j
- Чем ближе y_j к t_j, тем меньше изменяются веса.
- В процессе обновления параметров модели при рассмотрении пар "контекстное-целевое слово"из нашего текста эффект будет накапливаться. Можно представить, что входные вектора слов контекста "толкают"выходной вектор слова w. И аналогично выходные вектора толкают входной.

Несколько слов в контексте



- Пусть у нас C слов контекста. Усредним векторы слов контекста:

$$h = \frac{1}{C}W^T(x_1 + x_2 + ... + x_C) = \frac{1}{C}(v_{w_1} + v_{w_2} + ... + v_{w_C})^T$$

- Функция потерь:

$$E = -\log p(w_O \mid w_{I,1}, ..., w_{I,C}) = -u_{j^*} + \log \sum_{j'=1}^{V} exp(u_{j'}) = -v_{w_O}^{'T} h + \log \sum_{j'=1}^{V} exp(v_{u}^{'}$$

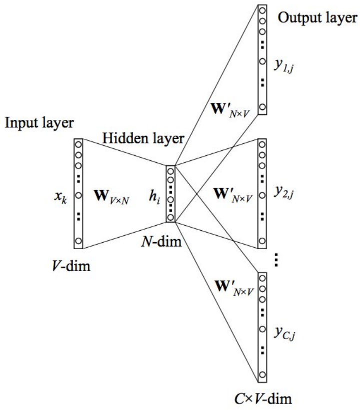
- Шаг градиентного спуска для весов между скрытым и выходным слоями тот же:

$$v_{w_j}^{'(new)} = v_{w_j}^{'(old)} - \eta e_j h_{\text{для}} j = 1, ..., V$$

- Шаг градиентного спуска для весов между скрытым и входным слоями:

$$v_{w_{I,c}}^{'(new)} = v_{w_{I,c}}^{(old)} - \frac{1}{C} \eta E H^T \quad (14)$$

Skip-Gram



- Предположим, что у нас есть одно слово, по которому мы предсказываем C слов контекста
- Размер входного слоя V, скрытого - N. Сеть полносвязная.
- На вход подаётся one-hot encoded вектор.

- Матрица весов между входным и скрытым слоями $W_{V \times N}$. i-ая строка v_w^T-векторное представление слова w, поступившего на вход. Полагая $x_k = 1$ и $x_l = 0$ для $l \neq k$, запишем

$$h = W^T x = W^T_{(k,\cdot)} := v^T_{w_I} \quad (1)$$

- На выходном слое получаем не одно мультиномиальное распределение, а C распределений. Каждое вычисляется с помощью одной и той же матрицы весов между скрытым и выходным слоем:

$$p(w_{c,j} = w_{O,c} \mid w_I) = y_{c,j} = \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{c,j'})}$$

- Матрица весов между входным и скрытым слоями $W'_{N \times V}$. i-ая строка v_w^T-векторное представление слова w, поступившего на вход. Вычислим значение u_j для каждого слова в словаре

$$u_{c,j} = {v'}_{w_j}^{T} h,$$

где v'_{w_j}-j-ый столбец W'

- Функция потерь:

$$E = -\log p(w_{O,1}, ..., w_{O,C} \mid w_I) = -\log \prod \frac{exp(u_{c,j})}{\sum_{j'=1}^{V} exp(u_{c,j'})} = \log y_{j^*}$$

$$= -\sum u_{j^*} + C \log \sum_{j'}^{V} exp(u_{j'})$$

- Нам понадобятся производные E по u_j:

$$\frac{\partial E}{\partial u_{c,j}} = y_{c,j} - t_{c,j} := e_{c,j},$$

где $t_j = 1$ только при $j = j^*$.

- Для удобства определим вектор $EI = \{EI_1, ..., EI_V\}$ как сумму ошибок выходного слоя каждого предсказываемого слова контекста:

$$EI_j = \sum_{c=1}^{C} e_{c,j}$$

- Теперь запишем производную по весам:

$$\frac{\partial E}{\partial w'_{ij}} = \frac{\partial E}{\partial u_{c,j}} \frac{\partial u_{c,j}}{\partial w'_{ij}} = EI_j h_i$$

- Шаг SGD:

$$w'^{(new)}_{ij} = w'^{(old)}_{ij} - \eta EI_j h_i,$$

$$v'^{(new)}_{w_j} = v'^{(old)}_{w_j} - \eta EI_j h_{\text{для}} j = 1, ..., V$$

- Шаг SGD для обновления весов между входным и скрытым слоями:

$$v'^{(new)}_{w_I} = v^{(old)}_{w_I} - \eta EH^T,$$

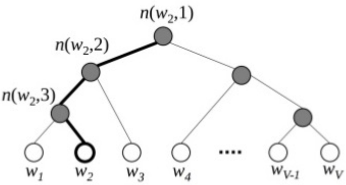
$$EH_i = \sum_{j=1}^{V} EI_j w'_{ij}$$

KASPERSKY

Оптимизация эффективности

- Для обновления выходных векторов $v_w^{'}$ нужно пройтись по каждому слову из словаря w_j, вычислить значение u_j, предсказание вероятности y_j и вероятность ошибки e_j.
- Таким образом, вычислительная сложность высоока. интуитивно очевидно, что для повышения эффективности нужно ограничить количество выходных векторов, которые обновляются для каждого примера обучающей выборки.
- Рассмотрим далее два подхода: иерархический софтмакс и негативное семплирование.
- Нас будут интересовать три значения, необходимые для обновления выходных векторов: E, $\frac{\partial E}{\partial v_w^{'}}$ и $\frac{\partial E}{\partial h}$.

Hierarchical Softmax



- Не используется векторное представление выходных слов. Вместо этого каждый из $V-1$ внутренних вершин имеет выходной вектор $v'_{n(w,j)}$. И вероятность того,что слово является целевым:

$$p(w = w_O) = \prod_{j=1}^{L(w)-1} \sigma([[n(w, j+1) = ch(n(w,j))]] {v'}^{T}_{n(w,j)} h)$$

- Выведем формулы для обновления весов в случае, когда на вход подаётся одно контекстное слово
  Обозначим $v'_j = v'_{n_{w,j}}$
- Для одного примера функция потерь:

$$E = -\log p(w = w_O \mid w_I) = -\sum_{j=1}^{L(w)-1} \log \sigma([[\cdot]] {v'_j}^T h)$$

- Производная по ${v'_j}^T h$:

$$\frac{\partial E}{\partial {v'_j}^T h} = (\sigma([[\cdot]]{v'_j}^T h) - 1)[[\cdot]] = \sigma({v'_j}^T h) - t_j,$$

где $t_j = 1$ если $[[\cdot]] = 1$ и $t_j = 0$ если $[[\cdot]] = -1$.

- Теперь производная по векторному представлению внутренней вершины

$$\frac{\partial E}{\partial v'_j} = \frac{\partial E}{\partial {v'_j}^T h} \frac{\partial {v'_j}^T h}{\partial v'_j} = (\sigma({v'_j}^T h) - t_j)h$$

- Шаг SGD:

$${v'_{w_I}}^{(new)} = v_{w_I}^{(old)} - \eta(\sigma({v'_j}^T h) - t_j)h, j = 1, ..., L(w) - 1$$

- Производная по h:

$$\frac{\partial E}{\partial h} = \sum_{\substack{L(w)-1 \\ j=1}} \frac{\partial E}{\partial {v'_j}^T h} \frac{\partial {v'_j}^T h}{\partial h} = \sum_{\substack{L(w)-1 \\ j=1}} (\sigma({v'_j}^T h) - t_j)v'_j := EH$$

- Т.о. сложность сократилась с $O(V)$ до $O(\log V)$ на контекстное слово при практически таком же количестве параметров (V-1 значение для внутренних вершин и V выходных векторов).

Negative Sampling

- Ограничим количество выходных векторов иным способом.
- Истинное выходное слово - позитивный пример - должно содержаться в нашем примере и обновляться, также нам нужно сэмплировать несколько слов как негативные примеры. Для сэмплирования нужно вероятностное распределение, которое выбирается произвольно. Такое распределение называется шумовым и обозначается $P_n(w)$.
- Используем упрощённую функцию (здесь $\sigma(u) = \frac{1}{1+exp(-u)}$)

$$E = -\log \sigma({v'_{w_O}}^T h) - \sum_{w_j \in W_{neg}} \log \sigma(-{v'_{w_j}}^T h)$$

- Производная по ${v'_{w_j}}^T h$:

$$\frac{\partial E}{\partial {v'_{w_j}}^T h} = \sigma({v'_{w_j}}^T h) - t_j,$$

  где $t_j = I_{w_j = w_O}$
- Шаг SGD:

$${v'_{w_j}}^{(new)} = v_{w_j}^{(old)} - \eta(\sigma({v'_{w_j}}^T h) - t_j)h, \quad w_j \in w_O \bigcup W_{neg}$$

- Т.о. мы применяем обновления только к $w_j \in w_O \bigcup W_{neg}$, а не к каждому слову из словаря.

- Производная по h:

$$\frac{\partial E}{\partial h} = \sum_{w_j \in w_O \bigcup W_{neg}} \frac{\partial E}{\partial {v'_{w_j}}^T h} \frac{\partial {v'_{w_j}}^T h}{\partial h} = \sum_{w_O \bigcup W_{neg}} (\sigma({v'_{w_j}}^T h) - t_j) v'_{w_j} := EH$$

Список литературы

- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean "Distributed Representations of Words and Phrases and their Compositionality"
- Xin Rong "word2vec Parameter Learning Explained"
- Yoav Goldberg, Omer Levy "word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method"
- Kaggle tutorial