Методы машинного обучения. Обучение без учителя: векторизация данных

Воронцов Константин Вячеславович
www.MachineLearning.ru/wiki?title=User:Vokov
вопросы к лектору: voron@forecsys.ru

материалы курса:
github.com/MSU-ML-COURSE/ML-COURSE-22-23
орг.вопросы по курсу: ml.cmc@mail.ru

МФТИ • 28 февраля 2023

Методы обучения без учителя (unsupervised learning)

Выявление структуры данных на основе сходства:

- кластеризация (clustering) и квантизация (quantization)
- оценивание плотности распределения (density estimation)
- одноклассовая классификация (anomaly detection)

Преобразование признакового пространства:

- метод главных компонент (principal components analysis)
- автокодировщики (autoencoders)
- многомерное шкалирование (multidimensional scaling)
- матричные разложения (matrix factorization)

Поиск взаимосвязей в данных или синтез учителя:

- поиск ассоциативных правил (association rule learning)
- частичное обучение (semi-supervised learning)
- самостоятельное обучение (self-supervised learning)

Содержание

Постановка задачи кластеризации и квантизации

Дано:
$X^\ell = \{x_i\}_{i=1}^{\ell}$ — обучающая выборка объектов, $x_i \in \mathbb{R}^n$
$\rho^2(x, w) = \|x - w\|^2$ — евклидова метрика в $\mathbb{R}^n$

Найти:
центры кластеров $w_y \in \mathbb{R}^n$, $y \in Y$; алгоритм кластеризации «правило жёсткой конкуренции» (WTA, Winner Takes All):

$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

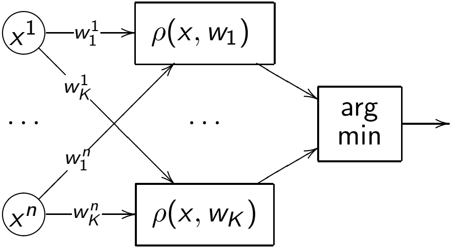
Критерий: среднее внутрикластерное расстояние

$$Q(w; X^\ell) = \sum_{i=1}^{\ell} \rho^2(x_i, w_{a(x_i)}) \to \min_{w_y \colon y \in Y}$$

Квантизация данных — замена x_i на ближайший центр $w_{a(x_i)}$

Сеть Кохонена (сеть с конкурентным обучением)

Структура алгоритма похожа на двухслойную нейросеть:



$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

Градиентный шаг в методе SG: для выбранного $x_i \in X^\ell$

$$w_y := w_y + \eta(x_i - w_y)\big[a(x_i) = y\big]$$

Если x_i относится к кластеру y, то w_y сдвигается в сторону x_i

T.Kohonen. Self-organized formation of topologically correct feature maps. 1982.

Алгоритм SG (Stochastic Gradient)

Вход: выборка X^ℓ; темп обучения η; параметр λ;
Выход: центры кластеров $w_1, \ldots, w_K \in \mathbb{R}^n$;

инициализировать центры $w_y, \;\; y \in Y$;
инициализировать текущую оценку функционала:
$Q := \sum\limits_{i=1}^{\ell} \rho^2(x_i, w_{a(x_i)})$;
повторять
- выбрать объект x_i из X^ℓ (например, случайно);
- определить ближайший центр: $y := \arg\min\limits_{y \in Y} \rho(x_i, w_y)$;
- градиентный шаг: $w_y := w_y + \eta(x_i - w_y)$;
- оценить значение функционала:
  $Q := (1 - \lambda)Q + \lambda\rho^2(x_i, w_y)$;

пока значение Q и/или веса w не стабилизируются;

Жёсткая и мягкая конкуренция

Правило жёсткой конкуренции WTA (winner takes all):

$$w_y := w_y + \eta(x_i - w_y)\big[a(x_i) = y\big], \quad y \in Y$$

Недостатки правила WTA:

- медленная скорость сходимости
- некоторые w_y могут никогда не выбираться

Правило мягкой конкуренции WTM (winner takes most):

$$w_y := w_y + \eta(x_i - w_y)\, K\big(\rho(x_i, w_y)\big), \quad y \in Y$$

где ядро $K(\rho)$ — неотрицательная невозрастающая функция

Теперь центры *всех* кластеров смещаются в сторону x_i,
но чем дальше от x_i, тем меньше величина смещения

Карта Кохонена (Self Organizing Map, SOM)

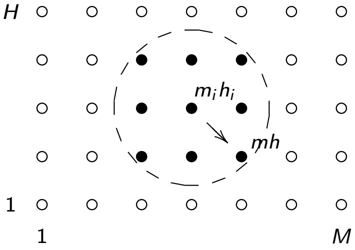
$Y = \{1, \ldots, M\} \times \{1, \ldots, H\}$ — прямоугольная сетка кластеров

Каждому узлу (m, h) приписан нейрон Кохонена $w_{mh} \in \mathbb{R}^n$

Наряду с метрикой $\rho(x_i, x)$ на X вводится метрика на сетке Y:

$$r\big((m_i, h_i), (m, h)\big) = \sqrt{(m - m_i)^2 + (h - h_i)^2}$$

Окрестность(m_i, h_i):



Teuvo Kohonen. Self-Organizing Maps. 2001.

Обучение карты Кохонена

Вход: X^ℓ — обучающая выборка; η — темп обучения;
Выход: $w_{mh} \in \mathbb{R}^n$ — векторы весов, $m = 1..M$, $h = 1..H$;

$w_{mh} := \text{random}\left(-\frac{1}{2MH}, \frac{1}{2MH}\right)$ — инициализация весов;

повторять

- выбрать объект x_i из X^ℓ случайным образом;
- WTA: вычислить координаты кластера:
  $(m_i, h_i) := a(x_i) \equiv \arg\min\limits_{(m,h)\in Y} \rho(x_i, w_{mh})$;
- **для всех** $(m, h) \in \text{Окрестность}(m_i, h_i)$
  - WTM: сделать шаг градиентного спуска:
    $w_{mh} := w_{mh} + \eta(x_i - w_{mh})\, K\big(r((m_i, h_i), (m, h))\big)$;

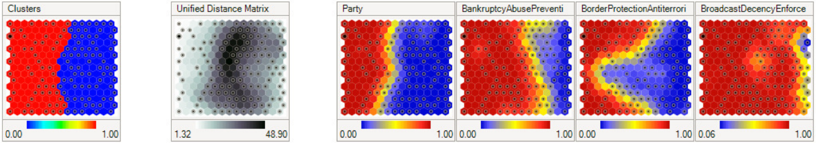
пока кластеризация не стабилизируется;

Интерпретация карт Кохонена

Два типа графиков — цветных карт $M \times H$:

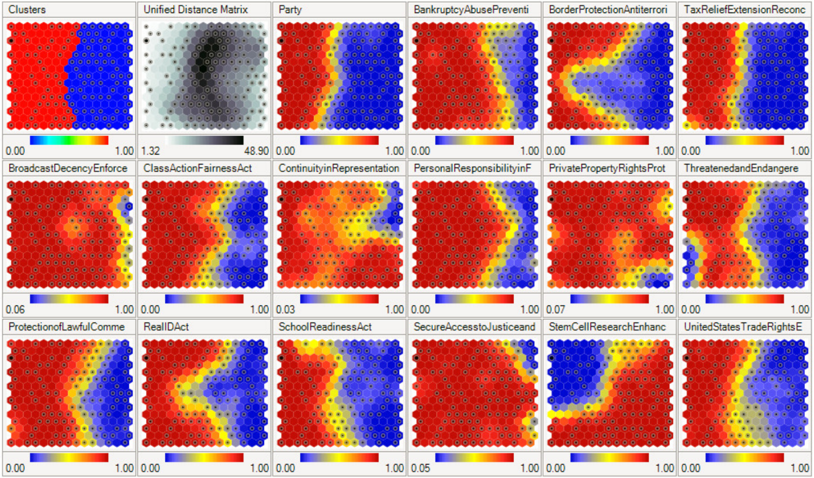
- Цвет узла (m, h) — локальная плотность в точке (m, h) — среднее расстояние до k ближайших точек выборки
- По одной карте на каждый признак:
  цвет узла (m, h) — значение j-й компоненты вектора $w_{m,h}$

Пример: задача UCI house-votes (US Congress voting patterns)
Объекты — конгрессмены
Признаки — результаты голосования по различным вопросам
Есть целевой признак «партия» $\in$ {демократ, республиканец}



Интерпретация карт Кохонена (продолжение примера)

Пример: задача UCI house-votes (US Congress voting patterns)



Достоинства и недостатки карт Кохонена

Достоинства:

- Возможность визуального анализа многомерных данных

Недостатки:

- **Субъективность.** Карта зависит не только от кластерной структуры данных, но и от...
  — свойств сглаживающего ядра;
  — (случайной) инициализации;
  — (случайного) выбора x_i в ходе итераций.
- **Искажения.** Близкие объекты исходного пространства могут переходить в далёкие точки на карте, и наоборот.

Рекомендуется только для разведочного анализа данных.

Построение автокодировщика — задача обучения без учителя

$X^\ell = \{x_1, \dots, x_\ell\}$ — обучающая выборка
$f\colon X \to Z$ — кодировщик (encoder), кодовый вектор $z = f(x, \alpha)$
$g\colon Z \to X$ — декодировщик (decoder), реконструкция $\hat{x} = g(z, \beta)$

Суперпозиция $\hat{x} = g(f(x))$ должна восстанавливать исходные x_i:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) = \sum_{i=1}^{\ell} \mathscr{L}\bigl(g(f(x_i, \alpha), \beta), x_i\bigr) \to \min_{\alpha, \beta}$$

Квадратичная функция потерь: $\mathscr{L}(\hat{x}, x) = \|\hat{x} - x\|^2$

Пример 1. Линейный автокодировщик: $x \in \mathbb{R}^n$, $z \in \mathbb{R}^m$

$$f(x, A) = \underset{m \times n}{A}\, x, \qquad g(z, B) = \underset{n \times m}{B}\, z$$

Пример 2. Двухслойная сеть с функциями активации σ_f, σ_g:

$$f(x, A) = \sigma_f(Ax + a), \qquad g(z, B) = \sigma_g(Bz + b)$$

Обучение и использование автокодировщиков

Метод обучения:

- Стохастический градиент (SG) по параметрам (α, β)

Способы использования:

- Векторизация данных:
  — понижение размерности (dimensionality reduction)
  — синтез более удачных признаков (feature generation)
  — сжатие данных с минимальной потерей информации
- Обучение с учителем в новом пространстве признаков
- Векторизация объектов для нейронных сетей
- Генерация синтетических объектов, похожих на реальные

Rumelhart, Hinton, Williams. Learning internal representations by error propagation. 1986.

David Charte et al. A practical tutorial on autoencoders for nonlinear feature fusion: taxonomy, models, software and guidelines. 2018.

Линейный автокодировщик и метод главных компонент

Линейный автокодировщик: $f(x, A) = Ax$, $g(z, B) = Bz$,

$$\mathscr{L}_{\text{AE}}(A, B) = \sum_{i=1}^{\ell} \|BAx_i - x_i\|^2 \to \min_{A,B}$$

Метод главных компонент: $f(x, U) = U^{\mathsf{T}} x$, $g(z, U) = Uz$,
в матричных обозначениях $F = (x_1 \dots x_\ell)^{\mathsf{T}}$, $U^{\mathsf{T}} U = I_m$, $G = FU$,

$$\|F - GU^{\mathsf{T}}\|^2 = \sum_{i=1}^{\ell} \|UU^{\mathsf{T}} x_i - x_i\|^2 \to \min_{U}$$

Автокодировщик обобщает метод главных компонент:

- не обязательно $B = A^{\mathsf{T}}$ (хотя часто именно так и делают)
- произвольные A, B вместо ортогональных
- нелинейные модели $f(x, \alpha)$, $g(z, \beta)$ вместо Ax, Bz
- произвольная функция потерь $\mathscr{L}$ вместо квадратичной
- SG оптимизация вместо сингулярного разложения SVD

Разреживающие автокодировщики (Sparse AE)

Применение L_1 или L_2-регуляризации к векторам весов α, β:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda\|\alpha\| + \lambda\|\beta\| \to \min_{\alpha,\beta}$$

Применение L_1-регуляризации к кодовым векторам z_i:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda \sum_{i=1}^{\ell} \sum_{j=1}^{m} |f_j(x_i, \alpha)| \to \min_{\alpha,\beta}$$

Энтропийная регуляризация для случая $f_j \in [0, 1]$:

$$\mathscr{L}_{\text{AE}}(\alpha, \beta) + \lambda \sum_{j=1}^{m} \text{KL}(\varepsilon\|\bar{f}_j) \to \min_{\alpha,\beta},$$

где $\bar{f}_j = \frac{1}{\ell} \sum_{i=1}^{\ell} f_j(x_i, \alpha)$; $\varepsilon \in (0, 1)$ — близкий к нулю параметр,
$\text{KL}(\varepsilon\|\rho) = \varepsilon \log \frac{\varepsilon}{\rho} + (1 - \varepsilon) \log \frac{1-\varepsilon}{1-\rho}$ — KL-дивергенция.

D.Arpit et al. Why regularized auto-encoders learn sparse representation? 2015.

Шумоподавляющий автокодировщик (Denoising AE)

Устойчивость кодовых векторов z_i относительно шума в x_i:

$$\mathscr{L}_{\text{DAE}}(\alpha, \beta) = \sum_{i=1}^{\ell} \mathsf{E}_{\tilde{x} \sim q(\tilde{x}|x_i)} \mathscr{L}\bigl(g(f(\tilde{x}, \alpha), \beta), x_i\bigr) \to \min_{\alpha, \beta}$$

Вместо вычисления $\mathsf{E}_{\tilde{x}}$ в методе SG объекты x_i сэмплируются и зашумляются по одному: $\tilde{x} \sim q(\tilde{x}|x_i)$.

Варианты зашумления $q(\tilde{x}|x_i)$:

- $\tilde{x} \sim \mathcal{N}(x_i, \sigma^2 I)$ — добавление гауссовского шума
- обнуление компонент вектора x_i с вероятностью p_0
- такие искажения $x_i \to \tilde{x}$, относительно которых реконструкция $\hat{x}_i$ должна быть устойчивой

P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. ICML-2008.

Реляционный автокодировщик (Relational AE)

Наряду с потерями реконструкции объектов минимизируем потери реконструкции отношений между объектами:

$$\mathscr{L}_{\text{AE}}(\alpha,\beta) + \lambda \sum_{i<j} \mathscr{L}\big(\sigma(\hat{x}_i^{\mathsf{T}}\hat{x}_j), \sigma(x_i^{\mathsf{T}}x_j)\big) \to \min_{\alpha,\beta}$$

где $\hat{x}_i = g(f(x_i,\alpha),\beta)$ — реконструкция объекта x_i,
$x_i^{\mathsf{T}}x_j$ — скалярное произведение (близость) пары объектов,
$\sigma(s) = (s - s_0)_+$ — пороговая функция с параметром s_0 (если векторы не близки, то неважно, насколько),
$\mathscr{L}(\hat{s}, s)$ — функция потерь, например, $(\hat{s} - s)^2$.

Эксперимент: улучшается качество классификации изображений с помощью кодовых векторов на задачах MNIST, CIFAR-10

Qinxue Meng et al. Relational autoencoder for feature extraction. 2018.

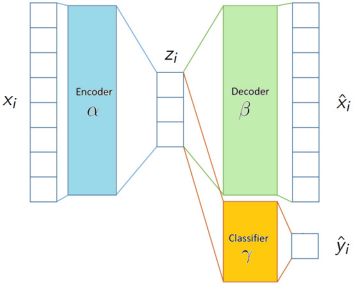
Автокодировщики для обучения с учителем

Данные: размеченные $(x_i, y_i)_{i=1}^k$, неразмеченные $(x_i)_{i=k+1}^\ell$

Совместное обучение кодировщика, декодировщика и предсказательной модели (классификации, регрессии или др.):

$$\sum_{i=1}^{\ell} \mathscr{L}\big(g(f(x_i,\alpha),\beta), x_i\big) + \lambda \sum_{i=1}^{k} \tilde{\mathscr{L}}\big(\hat{y}(f(x_i,\alpha),\gamma), y_i\big) \to \min_{\alpha,\beta,\gamma}$$

$z_i = f(x_i, \alpha)$ — кодировщик
$\hat{x}_i = g(z_i, \beta)$ — декодировщик
$\hat{y}_i = \hat{y}(z_i, \gamma)$ — предиктор



Функции потерь:
$\mathscr{L}(\hat{x}_i, x_i)$ — реконструкция
$\tilde{\mathscr{L}}(\hat{y}_i, y_i)$ — предсказание

Dor Bank, Noam Koenigstein, Raja Giryes. Autoencoders. 2020

Многомерное шкалирование (multidimensional scaling, MDS)

Дано: $(i,j) \in E$ — выборка рёбер графа $\langle V, E \rangle$,
R_{ij} — расстояния между вершинами ребра (i,j).
Например, R_{ij} — длина кратчайшего пути по графу (IsoMAP).

Найти: векторные представления вершин $z_i \in \mathbb{R}^d$, так, чтобы близкие (по графу) вершины имели близкие векторы.

Критерий стресса (stress):

$$\sum_{(i,j)\in E} w_{ij} \bigl(\rho(z_i, z_j) - R_{ij}\bigr)^2 \to \min_Z, \quad Z \in \mathbb{R}^{V\times d},$$

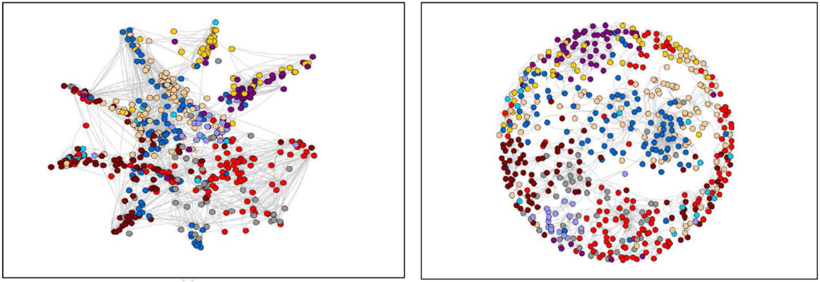
где $\rho(z_i, z_j) = \|z_i - z_j\|$ — обычно евклидово расстояние,
w_{ij} — веса (какие расстояния важнее, большие или малые).

Обычно решается методом стохастического градиента (SG).

I.Chami et al. Machine learning on graphs: a model and comprehensive taxonomy. 2020.

Многомерное шкалирование для визуализации данных

При $d = 2$ осуществляется проекция выборки на плоскость



- Используется для визуализации кластерных структур
- Форму облака точек можно настраивать весами и метрикой
- Недостаток — искажения неизбежны
- Популярная современная разновидность метода — t-SNE

Laurens van der Maaten, Geoffrey Hinton. Visualizing data using t-SNE. 2008

Метод векторного представления соседства (Stochastic Neighbor Embedding, SNE)

Дано: исходные точки $x_i \in \mathbb{R}^n$, $i = 1, \dots, \ell$
Найти: точки на карте-проекции $z_i \in \mathbb{R}^d$, $i = 1, \dots, \ell$, $d \ll n$
Критерий: расстояния $\|z_i - z_j\|$ близки к исходным $\|x_i - x_j\|$

Вероятностная модель события «j является соседом i»
на основе перенормированных гауссовских распределений:

$p(j|i) = \underset{j \neq i}{\mathrm{norm}} \exp\left(-\frac{1}{2\sigma_i^2}\|x_i - x_j\|^2\right)$ — в исходном пространстве;

$q(j|i) = \underset{j \neq i}{\mathrm{norm}} \exp\left(-\|z_i - z_j\|^2\right)$ — в пространстве проекции;

где $p_j = \underset{j}{\mathrm{norm}}(v_j) = \frac{v_j}{\sum_k v_k}$ — операция нормировки вектора v.

Максимизация правдоподобия (стохастическим градиентом):

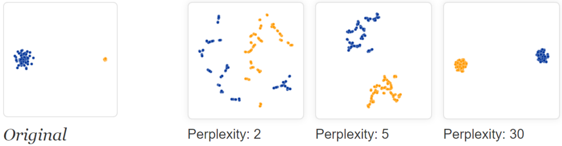
$$\sum_i \sum_{j \neq i} p(j|i) \ln q(j|i) \;\to\; \max_{\{z_i\}}$$

Преимущества метода SNE

- Преобразование расстояний в вероятности устраняет дисбалансы между большими и малыми расстояниями
- Дисбаланс между точками с большой и малой плотностью соседей выравнивается настройкой σ_i по перплексии

$H(i) = -\sum_j p(j|i) \log_2 p(j|i)$ — энтропия распределения $p(j|i)$;
$2^{H(i)}$ — перплексия = «эффективное число соседей у x_i»
(если $p(j|i) = \frac{1}{k}$, то $2^{H(i)} = k$); обычно перплексия = 5..50.

Выбор перплексии может существенно влиять на вид проекции:

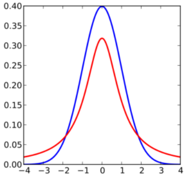


G.E.Hinton, S.T.Roweis. Stochastic Neighbor Embedding. 2002.

Вероятностная модель t-SNE: два усовершенствования SNE

Проблема скученности в SNE: окрестность вмещает гораздо больше точек в n-мерном пространстве, чем в d-мерном

- Использование t-распределения Стьюдента с более тяжёлым хвостом и симметричного совместного распределения $q(i,j)$:



$$q(i,j) = \underset{(i,j):\ i\neq j}{\mathrm{norm}} \left(1 + \|z_i - z_j\|^2\right)^{-1}$$

- Использование совместного распределения $p(i,j)$:

$$p(i,j) = \tfrac{1}{2\ell}\bigl(p(j|i) + p(i|j)\bigr)$$

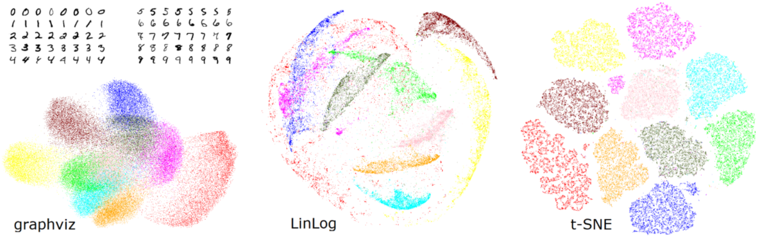
Максимизация правдоподобия (стохастическим градиентом):

$$\sum_{(i,j):\ j\neq i} p(i,j) \ln q(i,j) \to \max_{\{z_i\}}$$

L.J.P. van der Maaten, G.Hinton. Visualizing data using t-SNE. 2008

Преимущества и недостатки t-SNE

Лучшее представление структур сходства по сравнению с другими методами многомерного шкалирования (mnist)



Ложные кластерные структуры при низкой перплексии
Размеры кластеров и расстояния между ними неинформативны
Трудно отличить реальные структуры от артефактов метода
Нет ясного критерия качества для подбора перплексии

M.Wattenberg, F.Viegas, I.Johnson (Google). How to use t-SNE effectively. 2016.
https://distill.pub/2016/misread-tsne

Матричные разложения графа (graph factorization)

Дано: $(i,j) \in E$ — выборка рёбер графа $\langle V, E \rangle$,
S_{ij} — близость между вершинами ребра (i,j).
Например, $S_{ij} = [(i,j) \in E]$ — матрица смежности вершин.

Найти: векторные представления вершин, так, чтобы близкие (по графу) вершины имели близкие векторы.

Критерий для неориентированного графа (S симметрична):

$$\left\| S - ZZ^{\mathsf{T}} \right\|_E = \sum_{(i,j)\in E} \bigl(\langle z_i, z_j \rangle - S_{ij}\bigr)^2 \to \min_Z, \quad Z \in \mathbb{R}^{V\times d}$$

Критерий для ориентированного графа (S несимметрична):

$$\left\| S - \Phi\Theta^{\mathsf{T}} \right\|_E = \sum_{(i,j)\in E} \bigl(\langle \varphi_i, \theta_j \rangle - S_{ij}\bigr)^2 \to \min_{\Phi,\Theta}, \quad \Phi, \Theta \in \mathbb{R}^{V\times d}$$

Обычно решается методом стохастического градиента (SG).

I.Chami et al. Machine learning on graphs: a model and comprehensive taxonomy. 2020.

Векторные представления графов как автокодировщики

Все рассмотренные выше методы векторных представлений графов суть автокодировщики данных о рёбрах:

- многомерное шкалирование: $R_{ij} \to \|z_i - z_j\|$
- SNE и t-SNE: $p(i,j) \to q(i,j) \propto K(\|z_i - z_j\|)$
- матричные разложения: $S_{ij} \to \langle \varphi_i, \theta_j \rangle$

Вход кодировщика:

- W_{ij} — данные о ребре графа (i,j)

Выход кодировщика:

- векторные представления вершин z_i

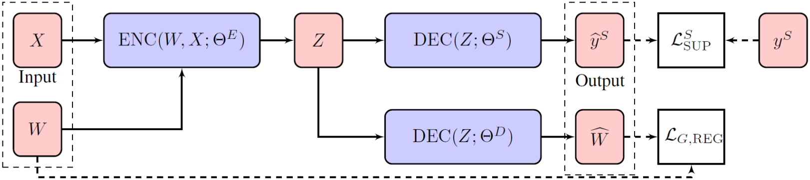
Выход декодировщика:

- аппроксимация $\hat{W}_{ij}$, вычисляемая по (z_i, z_j)

I.Chami et al. Machine learning on graphs: a model and comprehensive taxonomy. 2020.

GraphEDM: обобщённый автокодировщик на графах

Graph Encoder Decoder Model — обобщает более 30 моделей:



$W \in \mathbb{R}^{V \times V}$ — входные данные о рёбрах
$X \in \mathbb{R}^{V \times n}$ — входные данные о вершинах, признаковые описания
$Z \in \mathbb{R}^{V \times d}$ — векторные представления вершин графа
$\mathrm{DEC}(Z; \Theta^D)$ — декодер, реконструирующий данные о рёбрах
$\mathrm{DEC}(Z; \Theta^S)$ — декодер, решающий supervised-задачу
y^S — (semi-)supervised данные о вершинах или рёбрах
$\mathcal{L}$ — функции потерь

I.Chami et al. Machine learning on graphs: a model and comprehensive taxonomy. 2020.

Постановка задачи векторизации слов

Дано: текст $(w_1 \dots w_n)$, состоящий из слов словаря W

Найти: векторные представления (embedding) слов $v_w \in \mathbb{R}^d$, так, чтобы близкие по смыслу слова имели близкие векторы

Модель Skip-gram для предсказания вероятности слов контекста $C_i = (w_{i-k} \dots w_{i-1} w_{i+1} \dots w_{i+k})$ по слову w_i:

$$p(w|w_i) \;=\; \underset{w\in W}{\mathrm{SoftMax}} \langle u_w, v_{w_i} \rangle \;\equiv\; \underset{w\in W}{\mathrm{norm}} \bigl(\exp\langle u_w, v_{w_i}\rangle\bigr),$$

v_w — вектор предсказывающего слова,
u_w — вектор предсказываемого слова, в общем случае $u_w \neq v_w$.

Критерий максимума log-правдоподобия, $U, V \in \mathbb{R}^{|W|\times d}$:

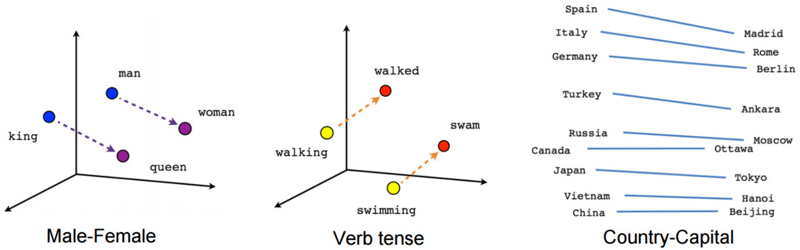
$$\sum_{i=1}^{n} \sum_{w\in C_i} \log p(w|w_i) \to \max_{U,V}$$

T.Mikolov et al. Efficient estimation of word representations in vector space, 2013.

Почему эмбединги слов отражают их смыслы

Основная гипотеза дистрибутивной семантики [Harris, 1954]:
«Смысл слова определяется множеством его контекстов»

Задача семантической аналогии слов:
по трём словам угадать четвёртое



Z.Harris. Distributional structure. 1954.
J.R.Firth. A synopsis of linguistic theory 1930-1955. Oxford, 1957.
P.Turney, P.Pantel. From frequency to meaning: vector space models of semantics. 2010.

Подмена задачи: классификация пар слов на два класса

Критерий log-loss для SGNS (Skip-gram Negative Sampling):

$$\sum_{i=1}^{n} \sum_{w \in C_i} \Bigl(\log p(+1|w, w_i) + \log p(-1|\bar{w}, w_i)\Bigr) \to \max_{U,V}$$

где $p(y|w, w_i) = \sigma\bigl(y\langle u_w, v_{w_i}\rangle\bigr)$ — модель классификации, $y = \pm 1$;
$y = +1$, если пара слов (w, w_i) находится в общем контексте;
$y = -1$, если пара слов (w, w_i) не находится в общем контексте;
$\bar{w} \sim p(w)^{3/4}$ сэмплируется из $W \backslash C_i$ в методе SG.

Эвристики и прочие замечания:

- Dynamic window: случайный выбор $k \sim [3..10]$
- Итоговые векторы слов: $\alpha v_w + (1 - \alpha) u_w$
- Приём NS полезен, когда не хватает второго класса

Модель векторных представлений FastText

Идея: векторное представление слова w определяется как сумма векторов всех его буквенных n-грамм $G(w)$:

$$u_w = \sum_{g \in G(w)} u_g$$

В Skip-gram вместо векторов слов u_w обучаются векторы u_g

Пример: $G(\text{дармолюб}) = \{\text{<да}, \text{арм}, \text{рмо}, \text{мол}, \text{олю}, \text{люб}, \text{юб>}\}$

Преимущества:

- Это решает проблемы новых слов и слов с опечатками
- Подходит для обработки текстов социальных медиа
- Словарь 2- и 3-грамм обычно меньше словаря W
- Существует много предобученных моделей

Bojanowski et al. Enriching word vectors with subword information. 2016.

Разновидности векторизации данных:

- *Квантизация* — сокращение объёма выборки, замена объектов ближайшими центрами кластеров
- *Автокодировщики* — синтез векторных представлений (эмбедингов) объектов, обычно с понижением размерности
- *MDS, t-SNE, GF, word2vec, graph2vec* и др. — синтез эмбедингов объектов по данным об их взаимодействии

Методы обучения — на основе SG

Тексты — это разновидность графов:

- слова — вершины графа, сочетаемость пары слов — ребро
- слова и документы — вершины двух разных долей графа, вхождение слова в документ — ребро двудольного графа