

Обзор оптимизационных задач машинного обучения

Воронцов Константин Вячеславович

vokov@forecsys.ru

<http://www.MachineLearning.ru/wiki?title=User:Vokov>

Полный годовой курс доступен на странице вики-ресурса

<http://www.MachineLearning.ru/wiki>

«Машинное обучение (курс лекций, К.В.Воронцов)»

КМЦ АГУ



8 июля 2021

1 Обучение с учителем

- Регрессия и классификация
- Регуляризация
- Обучение ранжированию

2 Обучение без учителя

- Восстановление плотности
- Кластеризация и частичное обучение
- Обучение представлений и автокодировщики

3 Неклассические парадигмы обучения

- Перенос обучения и многозадачное обучение
- Обучение с привилегированной информацией
- Генеративные состязательные сети (GAN)

Общая оптимизационная задача машинного обучения

Дано: выборка объектов $\{x_i\}_{i=1}^{\ell}$

Найти: вектор параметров w модели $a(x, w)$

Критерий: минимум эмпирического риска

$$Q(w) = \sum_{i=1}^{\ell} L_i(w) \rightarrow \min_w$$

где $L_i(w)$ — функция потерь модели $a(x, w)$ на объекте x_i ,
или минимум регуляризованного эмпирического риска

$$Q(w) = \sum_{i=1}^{\ell} L_i(w) + \sum_{j=1}^r \tau_j R_j(w) \rightarrow \min_w$$

где R_j — регуляризаторы, τ_j — коэффициенты регуляризации

Общие подходы к решению оптимизационных задач

Аналитический подход (напр. метод наименьших квадратов):
Если w — точка минимума *гладкой* функции $Q(w)$, то

$$\frac{\partial Q(w)}{\partial w_j} = 0, \quad j = 1, \dots, n.$$

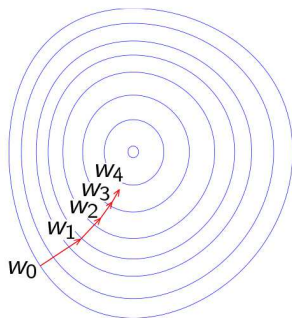
Это система n уравнений с n неизвестными.

Численный метод — градиентный спуск:
начальное приближение w^0 , $t := 0$;

повторять

$$\left| \begin{array}{l} w_j^{t+1} := w_j^t - h^t \cdot \frac{\partial Q(w^t)}{\partial w_j}, \quad j = 1, \dots, n; \\ t := t + 1; \end{array} \right.$$

пока процесс не сойдётся;



Оптимизационная задача восстановления регрессии

Обучающая выборка: $X^\ell = (x_i, y_i)_{i=1}^\ell$, $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$

- 1 Фиксируется модель регрессии, например, *линейная*:

$$a(x, w) = \langle x, w \rangle = \sum_{j=1}^n w_j f_j(x), \quad w \in \mathbb{R}^n$$

- 2 Фиксируется функция потерь, например, *квадратичная*:

$$L_i(w) = (a(x_i, w) - y_i)^2$$

- 3 Метод обучения — *метод наименьших квадратов*:

$$\sum_{i=1}^{\ell} (a(x_i, w) - y_i)^2 \rightarrow \min_w$$

- 4 Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:

$$Q(X^k) = \frac{1}{k} \sum_{i=1}^k (a(\tilde{x}_i, w) - \tilde{y}_i)^2$$

Оптимизационная задача обучения классификация

Обучающая выборка: $X^\ell = (x_i, y_i)_{i=1}^\ell$, $x_i \in \mathbb{R}^n$, $y_i \in \{-1, +1\}$

- 1 Фиксируется модель классификации, например, *линейная*:

$$a(x, w) = \text{sign}\langle x, w \rangle = \text{sign} \sum_{j=1}^n w_j f_j(x)$$

- 2 Функция потерь — пороговая или *её верхняя оценка*:

$$L_i(w) = [a(x_i, w)y_i < 0] = [\langle x_i, w \rangle y_i < 0] \leq \mathcal{L}(\langle x_i, w \rangle y_i)$$

- 3 Метод обучения — *минимизация эмпирического риска*:

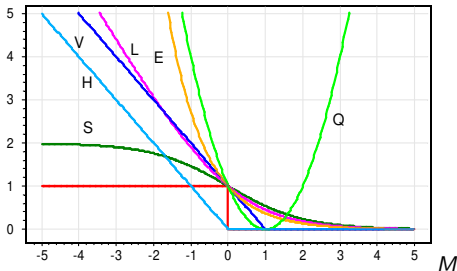
$$\sum_{i=1}^{\ell} [\langle x_i, w \rangle y_i < 0] \leq \sum_{i=1}^{\ell} \mathcal{L}(\langle x_i, w \rangle y_i) \rightarrow \min_w$$

- 4 Проверка по тестовой выборке $X^k = (\tilde{x}_i, \tilde{y}_i)_{i=1}^k$:

$$Q(X^k) = \frac{1}{k} \sum_{i=1}^k [\langle \tilde{x}_i, w \rangle \tilde{y}_i < 0]$$

Непрерывные верхние оценки пороговой функции потерь

Часто используемые непрерывные функции потерь $\mathcal{L}(M)$:



$[M < 0]$

$$V(M) = (1 - M)_+$$

$$H(M) = (-M)_+$$

$$L(M) = \log_2(1 + e^{-M})$$

$$Q(M) = (1 - M)^2$$

$$S(M) = 2(1 + e^M)^{-1}$$

$$E(M) = e^{-M}$$

— пороговая функция потерь

— кусочно-линейная (SVM)

— кусочно-линейная (Hebb's rule)

— логарифмическая (LR)

— квадратичная (FLD)

— сигмоидная (ANN)

— экспоненциальная (AdaBoost)

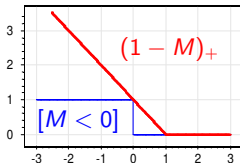
Метод опорных векторов SVM (двухклассовый)

$M_i(w, w_0) = y_i(\langle w, x_i \rangle - w_0)$ — отступ в линейной модели

Кусочно-линейная функция потерь:

$$\sum_{i=1}^{\ell} (1 - M_i(w, w_0))_+ + \frac{1}{2C} \|w\|^2 \rightarrow \min_{w, w_0}$$

- *Функция потерь* штрафует объекты за приближение к границе классов
- *Регуляризация* максимизирует зазор между классами и штрафует за мультиколлинеарность



Важнейшие свойства SVM:

- Задача выпуклого программирования, решение единственно
- Решение *разрежено* — зависит только от *опорных объектов*
- Обобщение на нелинейные модели: $\langle x, x_i \rangle \rightarrow K(x, x_i)$

Логистическая регрессия (двухклассовая)

Линейная модель классификации $a(x, w) = \text{sign}\langle x, w \rangle$

Логарифмическая функция потерь:

$$\sum_{i=1}^{\ell} \ln(1 + \exp(-\langle w, x_i \rangle y_i)) + \frac{\tau}{2} \|w\|^2 \rightarrow \min_w$$

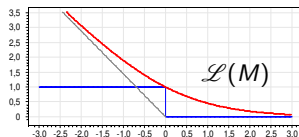
Логарифмическая функция потерь:

$$\mathcal{L}(M) = \ln(1 + e^{-M})$$

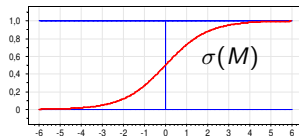
Модель условной вероятности:

$$P(y|x, w) = \sigma(M) = \frac{1}{1 + e^{-M}},$$

где $\sigma(M)$ — сигмоидная функция



M



M

Логистическая регрессия (многоклассовая)

Линейный классификатор при произвольном числе классов $|Y|$:

$$a(x, w) = \arg \max_{y \in Y} \langle w_y, x \rangle, \quad x, w_y \in \mathbb{R}^n$$

Вероятность того, что объект x относится к классу y :

$$P(y|x, w) = \frac{\exp\langle w_y, x \rangle}{\sum_{z \in Y} \exp\langle w_z, x \rangle} = \text{SoftMax}_{y \in Y} \langle w_y, x \rangle,$$

где $\text{SoftMax}: \mathbb{R}^Y \rightarrow \mathbb{R}^Y$ переводит произвольный вектор в нормированный вектор дискретного распределения.

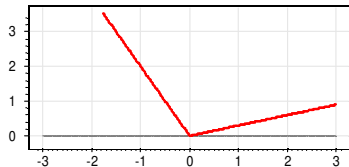
Максимизация правдоподобия (log-loss) с регуляризацией:

$$-\sum_{i=1}^{\ell} \ln P(y_i|x_i, w) + \frac{\tau}{2} \sum_{y \in Y} \|w_y\|^2 \rightarrow \min_w$$

Квантильная регрессия

Функция потерь, $\varepsilon = a(x_i, w) - y_i$:

$$\mathcal{L}(\varepsilon) = \begin{cases} C_+ |\varepsilon|, & \varepsilon > 0 \\ C_- |\varepsilon|, & \varepsilon < 0; \end{cases}$$



Модель регрессии: линейная $a(x_i, w) = \langle x_i, w \rangle$.

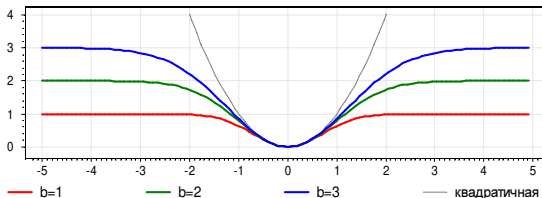
Сведение к задаче линейного программирования:

замена переменных $\varepsilon_i^+ = (a(x_i) - y_i)_+$, $\varepsilon_i^- = (y_i - a(x_i))_+$;

$$\begin{cases} \sum_{i=1}^{\ell} C_+ \varepsilon_i^+ + C_- \varepsilon_i^- \rightarrow \min_w; \\ \langle x_i, w \rangle - y_i = \varepsilon_i^+ - \varepsilon_i^-; \\ \varepsilon_i^+ \geq 0; \quad \varepsilon_i^- \geq 0. \end{cases}$$

Робастная регрессия

Функция Мешалкина: $\mathcal{L}(\varepsilon) = b(1 - \exp(-\frac{1}{b}\varepsilon^2))$, $\varepsilon = a - y$



Модель регрессии: $a(x, w)$

Постановка оптимизационной задачи:

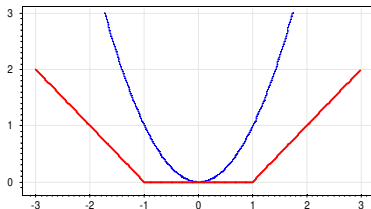
$$\sum_{i=1}^{\ell} \exp\left(-\frac{1}{b}(a(x_i, w) - y_i)^2\right) \rightarrow \max_w$$

Численное решение — методом Ньютона-Рафсона

SVM-регрессия

Модель регрессии: $a(x) = \langle x, w \rangle - w_0$, $w \in \mathbb{R}^n$, $w_0 \in \mathbb{R}$.

Функция потерь: $\mathcal{L}(\varepsilon) = (|\varepsilon| - \delta)_+$

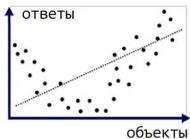


Постановка оптимизационной задачи:

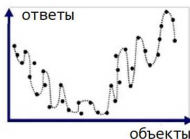
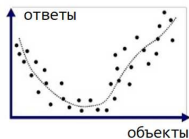
$$\sum_{i=1}^{\ell} (|\langle w, x_i \rangle - w_0 - y_i| - \delta)_+ + \frac{1}{2C} \|w\|^2 \rightarrow \min_{w, w_0}.$$

Сводится к выпуклой задаче квадратичного программирования

Проблемы недообучения и переобучения

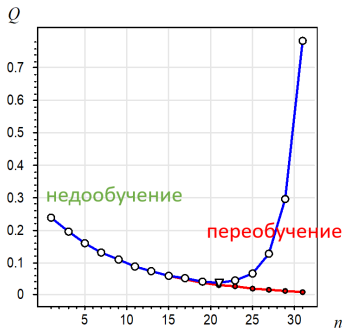


недообучение



переобучение

- **Недообучение** (underfitting): модель слишком проста, недостаточное число параметров n
- **Переобучение** (overfitting): модель слишком сложна, избыточное число параметров n



Проблема переобучения в линейных моделях

Возможные причины переобучения:

- линейная зависимость (мультиколлинеарность) признаков:
пусть построен классификатор: $a(x, w) = \text{sign}\langle w, x \rangle$;
мультиколлинеарность: $\exists u \in \mathbb{R}^n: \forall x_i \in X^\ell \langle u, x_i \rangle = 0$;
неединственность решения: $\forall \gamma \in \mathbb{R} \quad a(x, w) = \text{sign}\langle w + \gamma u, x \rangle$.
- слишком мало объектов; слишком много признаков;

Проявления переобучения:

- слишком большие веса $|w_j|$ разных знаков;
- неустойчивость дискриминантной функции $\langle w, x \rangle$;
- $Q(X^\ell) \ll Q(X^k)$;

Основной способ уменьшить переобучение:

- регуляризация — дополнительные ограничения на w

Регуляризаторы, штрафующие сложность линейной модели

Регуляризатор — аддитивная добавка к основному критерию:

$$\sum_{i=1}^{\ell} \mathcal{L}(\langle x_i, w \rangle, y_i) + \tau \text{штраф}(w) \rightarrow \min_w$$

где $\mathcal{L}(a, y)$ — функция потерь, τ — коэффициент регуляризации

L₂-регуляризация (гребневая регрессия, SVM):

$$\text{штраф}(w) = \|w\|_2^2 = \sum_{j=1}^n w_j^2.$$

L₁-регуляризация (LASSO, ElasticNet — для отбора признаков):

$$\text{штраф}(w) = \|w\|_1 = \sum_{j=1}^n |w_j|.$$

L₀-регуляризация (критерии Акаике AIC, байесовский BIC):

$$\text{штраф}(w) = \|w\|_0 = \sum_{j=1}^n [w_j \neq 0].$$

Негладкие регуляризаторы для отбора признаков

Общий вид регуляризаторов (μ — параметр селективности):

$$\sum_{i=1}^{\ell} \mathcal{L}(\langle x_i, w \rangle, y_i) + \tau \sum_{j=1}^n R_{\mu}(w_j) \rightarrow \min_w.$$

Регуляризаторы с эффектом группировки зависимых признаков:

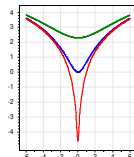
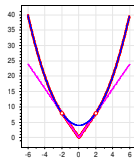
Elastic Net: $R_{\mu}(w) = \mu|w| + w^2$

Support Features Machine (SFM):

$$R_{\mu}(w) = \begin{cases} 2\mu|w|, & |w| \leq \mu; \\ \mu^2 + w^2, & |w| \geq \mu; \end{cases}$$

Relevance Features Machine (RFM):

$$R_{\mu}(w) = \ln(\mu w^2 + 1)$$



Задача обучения ранжированию (Learning to Rank)

Дано: $X^\ell = \{x_1, \dots, x_\ell\}$ — обучающая выборка
 $i \prec j$ — правильный порядок на парах (x_i, x_j)

Найти: модель ранжирования $a: X \rightarrow \mathbb{R}$ такую, что

$$i \prec j \Rightarrow a(x_i, w) < a(x_j, w)$$

Критерий: число неверно упорядоченных пар (x_i, x_j)
или аппроксимированный попарный эмпирический риск:

$$\sum_{i \prec j} [a(x_j, w) < a(x_i, w)] \leq \sum_{i \prec j} \underbrace{\mathcal{L}(a(x_j, w) - a(x_i, w))}_{M_{ij}(w)} \rightarrow \min_w$$

где $\mathcal{L}(M)$ — убывающая функция *парного отступа* $M_{ij}(w)$

Задача восстановления плотности распределения

Дано: обучающая выборка $\{x_i: i = 1, \dots, \ell\}$

Найти: вектор параметров θ в модели $p(x|\theta)$

Критерий: максимум правдоподобия

$$\sum_{i=1}^{\ell} \ln p(x_i|\theta) \rightarrow \max_{\theta}$$

или максимум апостериорной вероятности

$$\sum_{i=1}^{\ell} \ln p(x_i|\theta) + \ln p(\theta|\gamma) \rightarrow \max_{\theta}$$

где γ — вектор гиперпараметров априорного распределения

Задача восстановления смеси плотностей распределения

Дано: обучающая выборка $\{x_i: i = 1, \dots, \ell\}$

Найти: параметры w_j, θ_j в модели $p(x|\theta, w) = \sum_{j=1}^K w_j p(x|\theta_j)$

Критерий: максимум правдоподобия

$$\sum_{i=1}^{\ell} \ln p(x_i|\theta, w) \rightarrow \max_{\theta, w}$$

или максимум апостериорной вероятности

$$\sum_{i=1}^{\ell} \ln p(x_i|\theta, w) + \ln p(\theta, w|\gamma) \rightarrow \max_{\theta, w}$$

где γ — вектор гиперпараметров априорного распределения

Задача кластеризации (clustering)

Дано: обучающая выборка $\{x_i \in \mathbb{R}^n : i = 1, \dots, \ell\}$

Найти:

— центры кластеров $\mu_j \in \mathbb{R}^n, j = 1, \dots, K$

— какому кластеру принадлежит каждый объект $a_i \in \{1, \dots, K\}$

Критерий: минимум внутрикластерных расстояний

$$\sum_{i=1}^{\ell} \|x_i - \mu_{a_i}\|^2 \rightarrow \min_{\{a_i\}, \{\mu_j\}}$$

в случае евклидовой метрики

$$\|x - \mu_j\|^2 = \sum_{d=1}^n (f_d(x) - \mu_{jd})^2$$

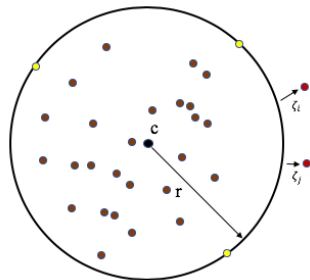
Одноклассовый SVM (one-class SVM, OSVM)

Дано: обучающая выборка $\{x_i \in \mathbb{R}^n : i = 1, \dots, \ell\}$

Найти: центр $c \in \mathbb{R}^n$ и радиус r шара, охватывающего всю выборку кроме аномальных объектов-выбросов

Критерий: минимизация радиуса шара и суммы штрафов за выход из шара:

$$\nu r^2 + \sum_{i=1}^{\ell} \mathcal{L}(\underbrace{r^2 - \|x_i - c\|^2}_{\zeta_i = \text{margin}(c,r)}) \rightarrow \min_{c,r}$$



При $\mathcal{L}(\zeta) = (-\zeta)_+$ свойства решения аналогичны SVM:

- Выпуклая задача квадратичного программирования
- Решение *разрежено* — зависит только от *опорных объектов*
- Обобщение на нелинейные модели: $\langle x_i, x_j \rangle \rightarrow K(x_i, x_j)$

Задача частичного обучения (semi-supervised learning, SSL)

Дано:

$X^k = \{x_1, \dots, x_k\}$ — размеченные объекты (labeled data);
 $\{y_1, \dots, y_k\}$

$U = \{x_{k+1}, \dots, x_\ell\}$ — неразмеченные объекты (unlabeled data).

Найти: классификации $\{a_{k+1}, \dots, a_\ell\}$ неразмеченных объектов

Критерий без модели классификации (transductive learning):

$$\sum_{i=1}^{\ell} \|x_i - \mu_{a_i}\|^2 + \lambda \sum_{i=1}^k [a_i \neq y_i] \rightarrow \min_{\{a_i\}, \{\mu_j\}}$$

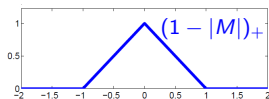
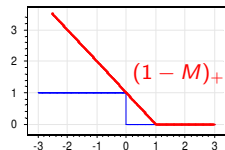
При построении модели классификации, $a_i = a(x_i, w)$:

$$\sum_{i=1}^{\ell} \|x_i - \mu_{a_i}\|^2 + \lambda \sum_{i=1}^k \mathcal{L}(a(x_i, w), y_i) \rightarrow \min_{\{a_i\}, \{\mu_j\}, w}$$

Метод TSVM — трансдуктивный SVM

$M_i = (\langle w, x_i \rangle - w_0) y_i$ — отступ объекта x_i

- Функция потерь $\mathcal{L}(M) = (1 - M)_+$ штрафует за уменьшение отступа
- Функция потерь $\mathcal{L}(M) = (1 - |M|)_+$ штрафует за попадание объекта внутрь разделяющей полосы



Обучение весов w, w_0 по частично размеченной выборке:

$$\sum_{i=1}^k (1 - M_i(w, w_0))_+ + \gamma \sum_{i=k+1}^{\ell} (1 - |M_i(w, w_0)|)_+ + \frac{\|w\|^2}{2C} \rightarrow \min_{w, w_0}$$

Частный случай SSL: PU-learning (Positive and Unlabeled)

Примеры задач, когда известны объекты только одного класса:

- обнаружение мошеннических транзакций
- рекомендательные системы, персонализация рекламы
- медицинская диагностика при неизвестном анамнезе
- автоматическое пополнение базы знаний фактами

Модель двухклассовой классификации $a(x_i, w)$.

Неразмеченные трактуются как негативные с весом $C_- \ll C_+$:

$$C_+ \sum_{i=1}^k \mathcal{L}(a(x_i, w), +1) + C_- \sum_{i=k+1}^{\ell} \mathcal{L}(a(x_i, w), -1) + R(w) \rightarrow \min_w$$

Один из успешных методов — Biased SVM.

Gang Li. A Survey on Positive and Unlabelled Learning. 2013.

J.Bekker, J.Davis. Learning From Positive and Unlabeled Data: A Survey. 2020.

Задачи низкорангового матричного разложения

- Понижение размерности для классификации/регрессии
- Формирование векторных представлений объектов
- Восстановление пропущенных значений в матрице

Дано: матрица $Z = \|z_{ij}\|_{n \times m}$, $(i, j) \in \Omega \subseteq \{1..n\} \times \{1..m\}$

Найти: матрицы $X = \|x_{it}\|_{n \times k}$ и $Y = \|y_{tj}\|_{k \times m}$ такие, что

$$\|Z - XY\| = \sum_{(i,j) \in \Omega} \mathcal{L}\left(z_{ij} - \sum_t x_{it} y_{tj}\right) \rightarrow \min_{X, Y}$$

Почему на практике отказываются от классического SVD:

- неквадратичная функция потерь \mathcal{L}
- неотрицательное матричное разложение: $x_{it} \geq 0$, $y_{tj} \geq 0$
- разреженные данные: $|\Omega| \ll nm$
- ортогональность не нужна или не интерпретируема

Примеры прикладных задач матричного разложения

- 1 **Выявление интересов в рекомендательных системах (recommender systems, collaborative filtering)**

$$z_{iu} = \sum_t p_{it} q_{tu}$$

дано: z_{iu} — рейтинги товаров i , поставленные пользователем u ;

найти: p_{it} — профиль интересов товара i ;

q_{tu} — профиль интересов пользователя u .

- 2 **Латентный семантический анализ коллекций текстов (тематическое моделирование)**

$$z_{wd} = \sum_t \varphi_{wt} \theta_{td}$$

дано: $z_{wd} = p(w|d)$ — частоты слов w в документах d ;

найти: $\varphi_{wt} = p(w|t)$ — распределения слов w в темах t ,

$\theta_{td} = p(t|d)$ — распределения тем t в документах d .

Задача построения автокодировщика (обучение без учителя)

$X^\ell = \{x_1, \dots, x_\ell\}$ — обучающая выборка

$f: X \rightarrow Z$ — кодировщик (encoder), кодовый вектор $z = f(x, \alpha)$

$g: Z \rightarrow X$ — декодировщик (decoder), реконструкция $\hat{x} = g(z, \beta)$

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

$$\mathcal{L}_{AE}(\alpha, \beta) = \sum_{i=1}^{\ell} \mathcal{L}(g(f(x_i, \alpha), \beta), x_i) \rightarrow \min_{\alpha, \beta}$$

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

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

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

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

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

Автокодировщики для обучения с учителем

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

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

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

$z_i = f(x_i, \alpha)$ — кодировщик

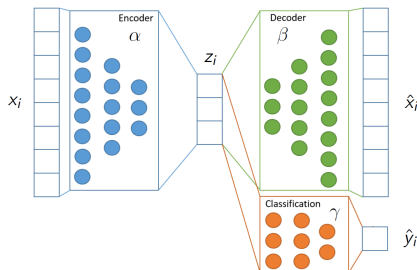
$\hat{x}_i = g(z_i, \beta)$ — декодировщик

$\hat{y}_i = \hat{y}(z_i, \gamma)$ — предиктор

Функции потерь:

$\mathcal{L}(\hat{x}_i, x_i)$ — реконструкция

$\tilde{\mathcal{L}}(\hat{y}_i, y_i)$ — предсказание



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) .

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

Критерий стресса (stress):

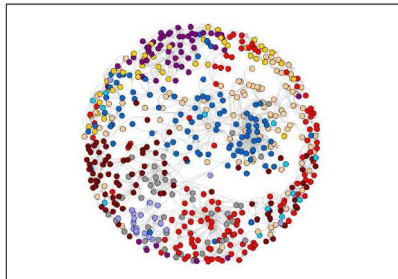
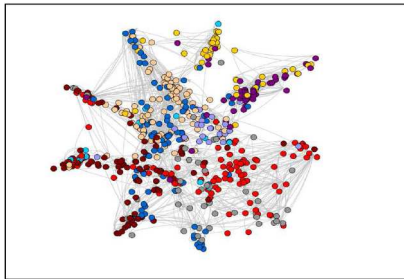
$$\sum_{(i,j) \in E} w(R_{ij}) (\rho(z_i, z_j) - R_{ij})^2 \rightarrow \min_Z, \quad Z \in \mathbb{R}^{V \times d},$$

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

Обычно решается методом стохастического градиента (SG).

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
- Наиболее популярный метод для визуализации — t-SNE

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

Графовые (матричные) разложения (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):

$$\sum_{(i,j) \in E} (\langle z_i, z_j \rangle - S_{ij})^2 \rightarrow \min_Z, \quad Z \in \mathbb{R}^{V \times d}$$

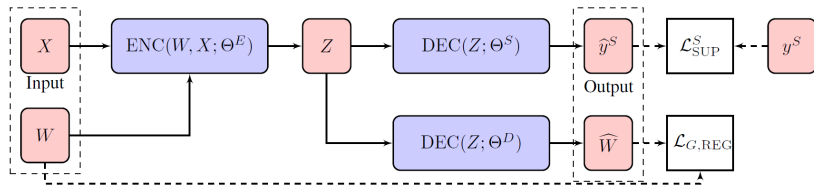
Критерий для ориентированного графа (S несимметрична):

$$\sum_{(i,j) \in E} (\langle \varphi_i, \theta_j \rangle - S_{ij})^2 \rightarrow \min_{\Phi, \Theta}, \quad \Phi, \Theta \in \mathbb{R}^{V \times d}$$

Обычно решается методом стохастического градиента (SG).

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

$\text{DEC}(Z; \Theta^D)$ — декодер, реконструирующий данные о рёбрах

$\text{DEC}(Z; \Theta^S)$ — декодер, решающий supervised-задачу

y^S — (semi-)supervised данные о вершинах или рёбрах

\mathcal{L} — функции потерь

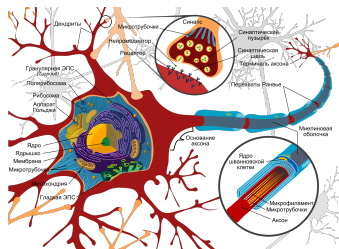
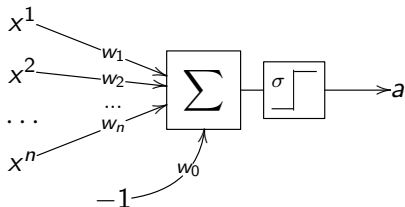
I. Chami et al. Machine learning on graphs: a model and comprehensive taxonomy. 2020.

Линейная модель нейрона [МакКаллок, Питтс, 1943]

$f_j: X \rightarrow \mathbb{R}$, $j = 1, \dots, n$ — числовые признаки;

$$a(x, w) = \sigma(\langle w, x \rangle) = \sigma\left(\sum_{j=1}^n w_j f_j(x) - w_0\right),$$

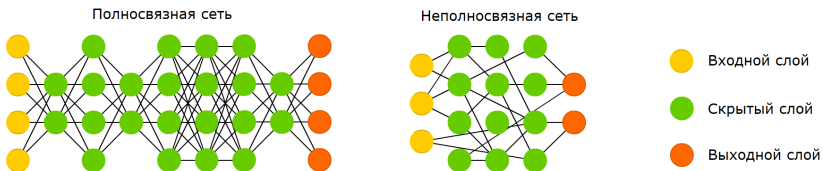
w_j — веса признаков, $\sigma(z)$ — функция активации



Глубокие нейронные сети (Deep Neural Network, DNN)

1965: первые глубокие нейронные сети

2012: свёрточная сеть для классификации изображений AlexNet

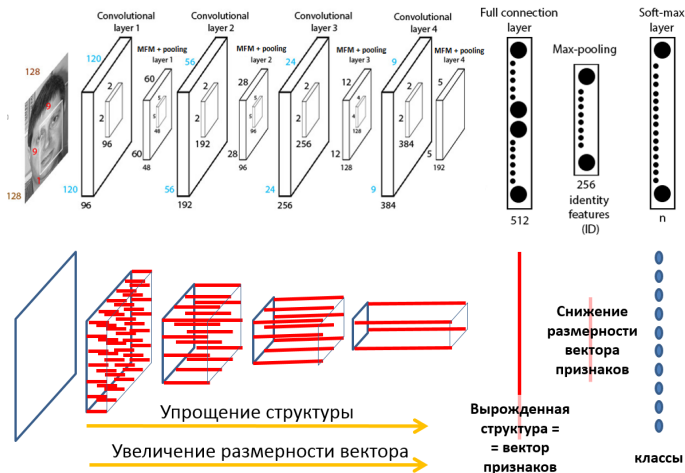


- *Архитектура сети* — структура связей между нейронами, позволяющая наделять DNN нужными свойствами
- DNN позволяют принимать на входе и генерировать на выходе *сложно структурированные данные*

Ивахненко А. Г., Лапа В. Г. Кибернетические предсказывающие устройства. 1965.

Krizhevsky A. et al. ImageNet classification with deep convolutional neural networks. 2012.

Глубокая свёрточная сеть как способ векторизации изображений

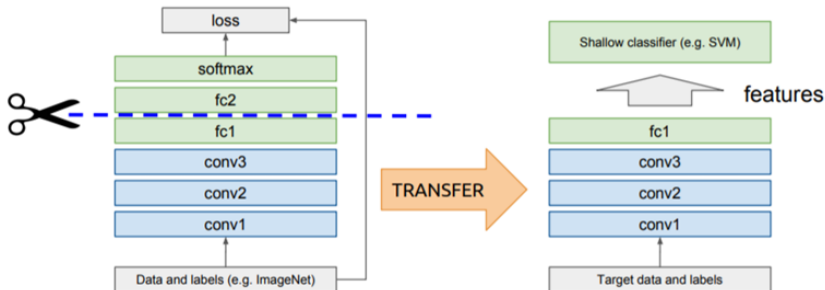


Визильтер Ю.В., Горбачевич В.С. Структурно-функциональный анализ и синтез глубоких конволюционных нейронных сетей. ММРО-2017.

Пред-обучение нейронных сетей (pre-training)

Свёрточная сеть для обработки изображений:

- $z = f(x, \alpha)$ — свёрточные слои для векторизации объектов
- $y = g(z, \beta)$ — полносвязные слои под конкретную задачу



Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson. How transferable are features in deep neural networks? 2014.

Перенос обучения (transfer learning)

$f(x, \alpha)$ — универсальная часть модели (векторизация данных)

$g(x, \beta)$ — специфичная для задачи часть модели

Базовая задача на выборке $\{x_i\}_{i=1}^{\ell}$ с функцией потерь \mathcal{L}_i :

$$\sum_{i=1}^{\ell} \mathcal{L}_i(f(x_i, \alpha), g(x_i, \beta)) \rightarrow \min_{\alpha, \beta}$$

Целевая задача на другой выборке $\{x'_i\}_{i=1}^m$, с другими \mathcal{L}'_i, g' :

$$\sum_{i=1}^m \mathcal{L}'_i(f(x'_i, \alpha), g'(x'_i, \beta')) \rightarrow \min_{\beta'}$$

при $m \ll \ell$ это может быть намного лучше, чем

$$\sum_{i=1}^m \mathcal{L}'_i(f(x'_i, \alpha), g'(x'_i, \beta')) \rightarrow \min_{\alpha, \beta'}$$

Многозадачное обучение (multi-task learning)

$f(x, \alpha)$ — универсальная часть модели (векторизация)

$g_t(x, \beta)$ — специфичная часть модели для задачи $t \in T$

Одновременное обучение модели f по задачам X_t , $t \in T$:

$$\sum_{t \in T} \sum_{i \in X_t} \mathcal{L}_{ti}(f(x_{ti}, \alpha), g_t(x_{ti}, \beta_t)) \rightarrow \min_{\alpha, \{\beta_t\}}$$

Обучаемость (learnability): качество решения отдельной задачи $\langle X_t, \mathcal{L}_t, g_t \rangle$ улучшается с ростом объёма выборки $\ell_t = |X_t|$.

Learning to learn: качество решения каждой из задач $t \in T$ улучшается с ростом как ℓ_t , так и общего числа задач $|T|$.

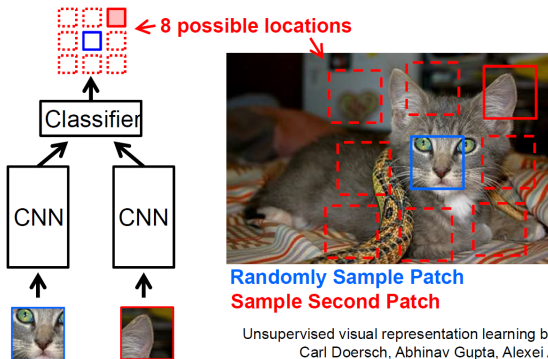
Few-shot learning: для решения задачи t достаточно небольшого числа примеров, иногда даже одного.

M. Crawshaw. Multi-task learning with deep neural networks: a survey. 2020

Y. Wang et al. Generalizing from a few examples: a survey on few-shot learning. 2020

Самостоятельное обучение (self-supervised learning)

Модель векторизации $z = f(x, \alpha)$ обучается предсказывать взаимное расположение пар фрагментов одного изображения



Преимущество: сеть выучивает векторные представления объектов без размеченной обучающей выборки (без ImageNet).

Дистилляция моделей или суррогатное моделирование

Обучение **сложной модели** $a(x, w)$ «долго, дорого»:

$$\sum_{i=1}^{\ell} \mathcal{L}(a(x_i, w), y_i) \rightarrow \min_w$$

Обучение простой модели $b(x, w')$, возможно, на других данных:

$$\sum_{i=1}^k \mathcal{L}(b(x'_i, w'), a(x'_i, w)) \rightarrow \min_{w'}$$

Примеры задач:

- замена сложной модели (климат, аэродинамика и др.), которая вычисляется на суперкомпьютере месяцами, «лёгкой» аппроксимирующей суррогатной моделью
- замена сложной нейросети, которая обучается неделями на больших данных, «лёгкой» аппроксимирующей нейросетью с минимизацией числа нейронов и связей

Задача обучения с привилегированной информацией

x_i^* — информация об объекте x_i , доступная только на обучении

Раздельное обучение модели-ученика и **модели-учителя**:

$$\sum_{i=1}^{\ell} \mathcal{L}(a(x_i, w), y_i) \rightarrow \min_w \quad \sum_{i=1}^{\ell} \mathcal{L}(a(x_i^*, w^*), y_i) \rightarrow \min_{w^*}$$

Модель-ученик обучается повторять ошибки **модели-учителя**:

$$\sum_{i=1}^{\ell} \mathcal{L}(a(x_i, w), y_i) + \mu \mathcal{L}(a(x_i, w), a(x_i^*, w^*)) \rightarrow \min_w$$

Совместное обучение модели-ученика и **модели-учителя**:

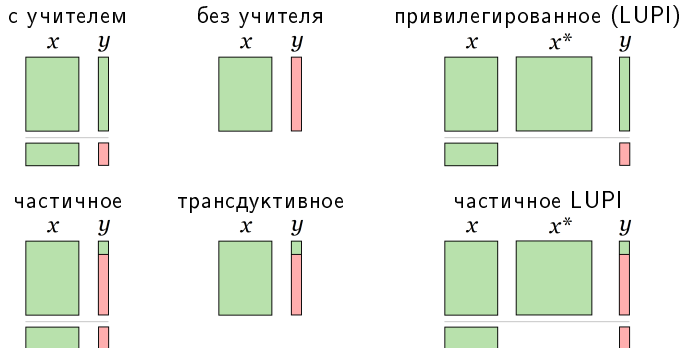
$$\sum_{i=1}^{\ell} \mathcal{L}(a(x_i, w), y_i) + \lambda \mathcal{L}(a(x_i^*, w^*), y_i) + \mu \mathcal{L}(a(x_i, w), a(x_i^*, w^*)) \rightarrow \min_{w, w^*}$$

D. Lopez-Paz, L. Bottou, B. Scholkopf, V. Vapnik. Unifying distillation and privileged information. 2016.

Обучение с использованием привилегированной информации

x_i^* — информация об объекте x_i , доступная только на обучении

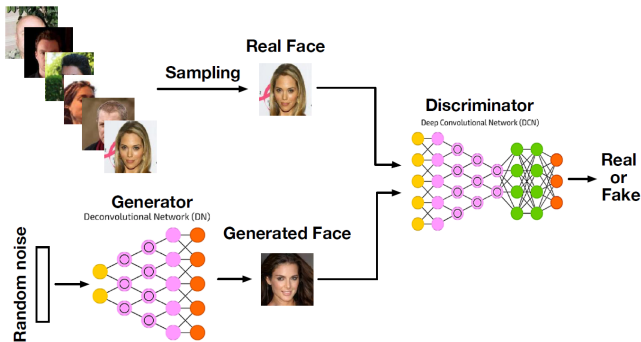
Варианты LUPI (Learning Using Privileged Information):



V. Vapnik, A. Vashist. A new learning paradigm: Learning Using Privileged Information // Neural Networks. 2009.

Генеративная состязательная сеть (Generative Adversarial Net)

Генератор $G(z)$ учится порождать объекты x из шума z
Дискриминатор $D(x)$ учится отличать их от реальных объектов



Antonia Creswell et al. Generative Adversarial Networks: an overview. 2017.
Zhengwei Wang et al. Generative Adversarial Networks: a survey and taxonomy. 2019.
Chris Nicholson. A Beginner's Guide to Generative Adversarial Networks.
<https://pathmind.com/wiki/generative-adversarial-network-gan>. 2019.

Постановка задачи GAN

Дано: выборка объектов $\{x_i\}_{i=1}^m$ из X

Найти:

вероятностную генеративную модель $G(z, \alpha): x \sim p(x|z, \alpha)$

вероятностную дискриминативную модель $D(x, \beta) = p(1|x, \beta)$

Критерий:

обучение дискриминативной модели D :

$$\sum_{i=1}^m \ln D(x_i, \beta) + \ln(1 - D(G(z_i, \alpha), \beta)) \rightarrow \max_{\beta}$$

обучение генеративной модели G по случайному шуму $\{z_i\}_{i=1}^m$:

$$\sum_{i=1}^m \ln(1 - D(G(z_i, \alpha), \beta)) \rightarrow \min_{\alpha}$$

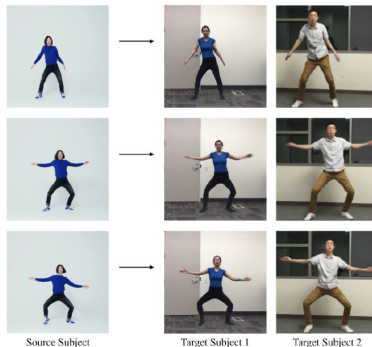
Примеры GAN для синтеза изображений и видео



(d) input image

(e) output 3d face

(f) textured 3d face



Source Subject

Target Subject 1

Target Subject 2

Chuan Li, Michael Wand. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. 2016.

Xiaoxing Zeng, Xiaojiang Peng, Yu Qiao. DF2Net: A Dense Fine Finer Network for Detailed 3D Face Reconstruction. ICCV-2019.

Caroline Chan, Shiry Ginosar, Tinghui Zhou, Alexei A. Efros. Everybody Dance Now. ICCV-2019.

- 1 Предварительная обработка (data preparation)
 - извлечение признаков (feature extraction)
 - отбор признаков (feature selection)
 - восстановление пропусков (missing values)
 - фильтрация выбросов (outlier detection)
- 2 Обучение с учителем (supervised learning)
 - классификация (classification)
 - регрессия (regression)
 - ранжирование (learning to rank)
 - прогнозирование (forecasting)
- 3 Обучение без учителя (unsupervised learning)
 - кластеризация (clustering)
 - поиск ассоциативных правил (association rule learning)
 - восстановление плотности (density estimation)
 - одноклассовая классификация (anomaly detection)
- 4 Частичное обучение (semi-supervised learning)
 - трансдуктивное обучение (transductive learning)
 - обучение с положительными примерами (PU-learning)

- 5 Обучение представлений (representation learning)
 - обучение признаков (feature learning)
 - матричные разложения (matrix factorization)
 - обучение многообразий (manifold learning)
- 6 Глубокое обучение (deep learning)
- 7 Обучение близости/связей (similarity/relational learning)
- 8 Обучение структуры модели (structure learning)
- 9 Привилегированное обучение (privileged learning, distilling)
- 10 Состязательное обучение (adversarial learning)
- 11 Динамическое обучение (online/incremental learning)
- 12 Активное обучение (active learning)
- 13 Обучение с подкреплением (reinforcement learning)
- 14 Перенос обучения (transfer learning)
- 15 Многозадачное обучение (multitask learning)
- 16 Мета-обучение (meta-learning, AutoML)