Методы машинного обучения. Нейронные сети: обучение без учителя

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-21-22
орг.вопросы по курсу: ml.cmc@mail.ru

МФТИ • 22 февраля 2022

Содержание

Постановка задачи кластеризации (обучения без учителя)

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):

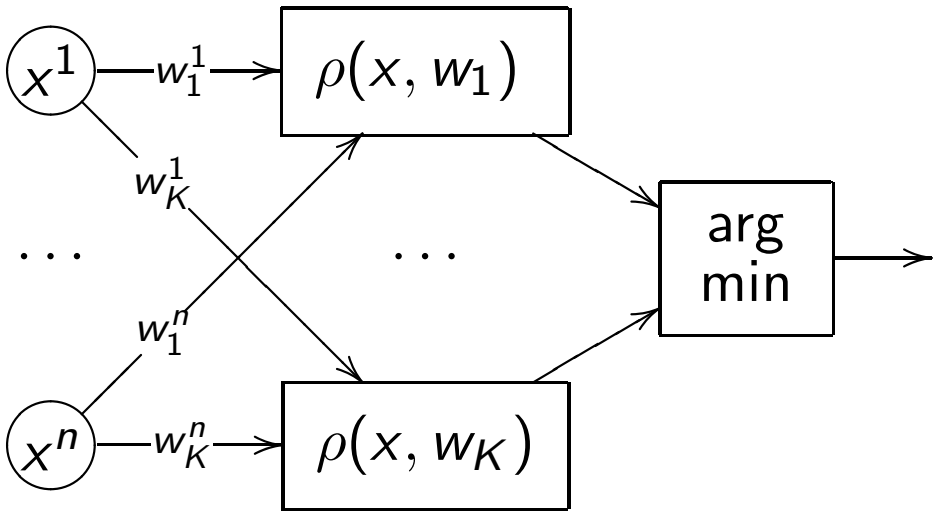
$$a(x) = \arg\min_{y \in Y} \rho(x, w_y)$$

Критерий: среднее внутрикластерное расстояние

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

Сеть Кохонена (сеть с конкурентным обучением)

Структура алгоритма — двухслойная нейронная сеть:



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

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:
$Q := \sum_{i=1}^{\ell} \rho^2(x_i, w_{a(x_i)})$;
повторять
  выбрать объект x_i из X^ℓ (например, случайно);
  вычислить кластеризацию: $y := \arg\min_{y \in Y} \rho(x_i, w_y)$;
  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 WTM:

- 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

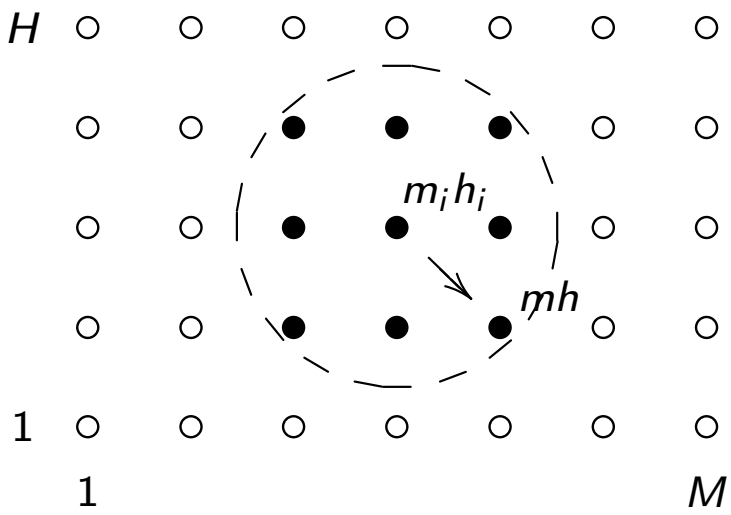
Теперь центры *всех* кластеров смещаются в сторону x_i,
но чем дальше от x_i, тем меньше величина смещения

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:
$(m_i, h_i) := a(x_i) \equiv \arg\min_{(m,h)\in Y} \rho(x_i, w_{mh})$;
для всех $(m, h) \in$ Окрестность(m_i, h_i)
WTM: сделать шаг градиентного спуска:
$w_{mh} := w_{mh} + \eta(x_i - w_{mh})\, K\big(r((m_i, h_i), (m, h))\big)$;

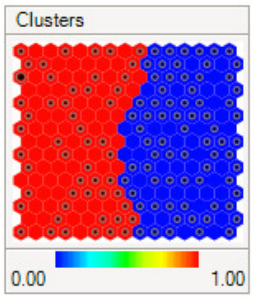
пока кластеризация не стабилизируется;

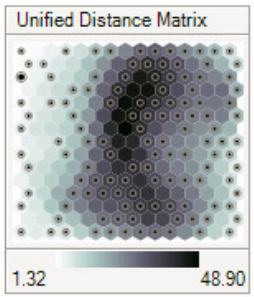
Интерпретация карт Кохонена

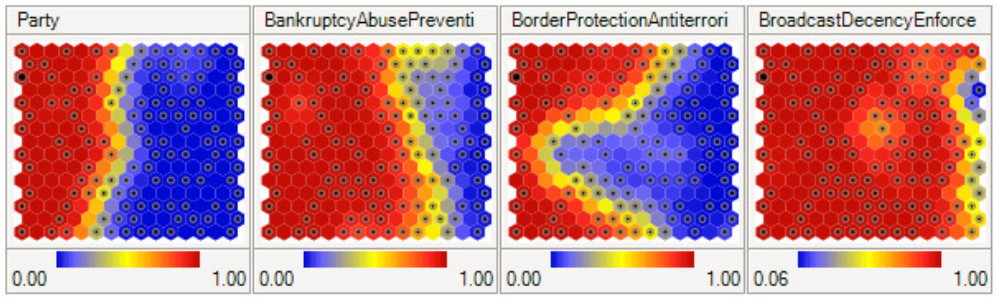
Два типа графиков — цветных карт $M \times H$:

- 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}

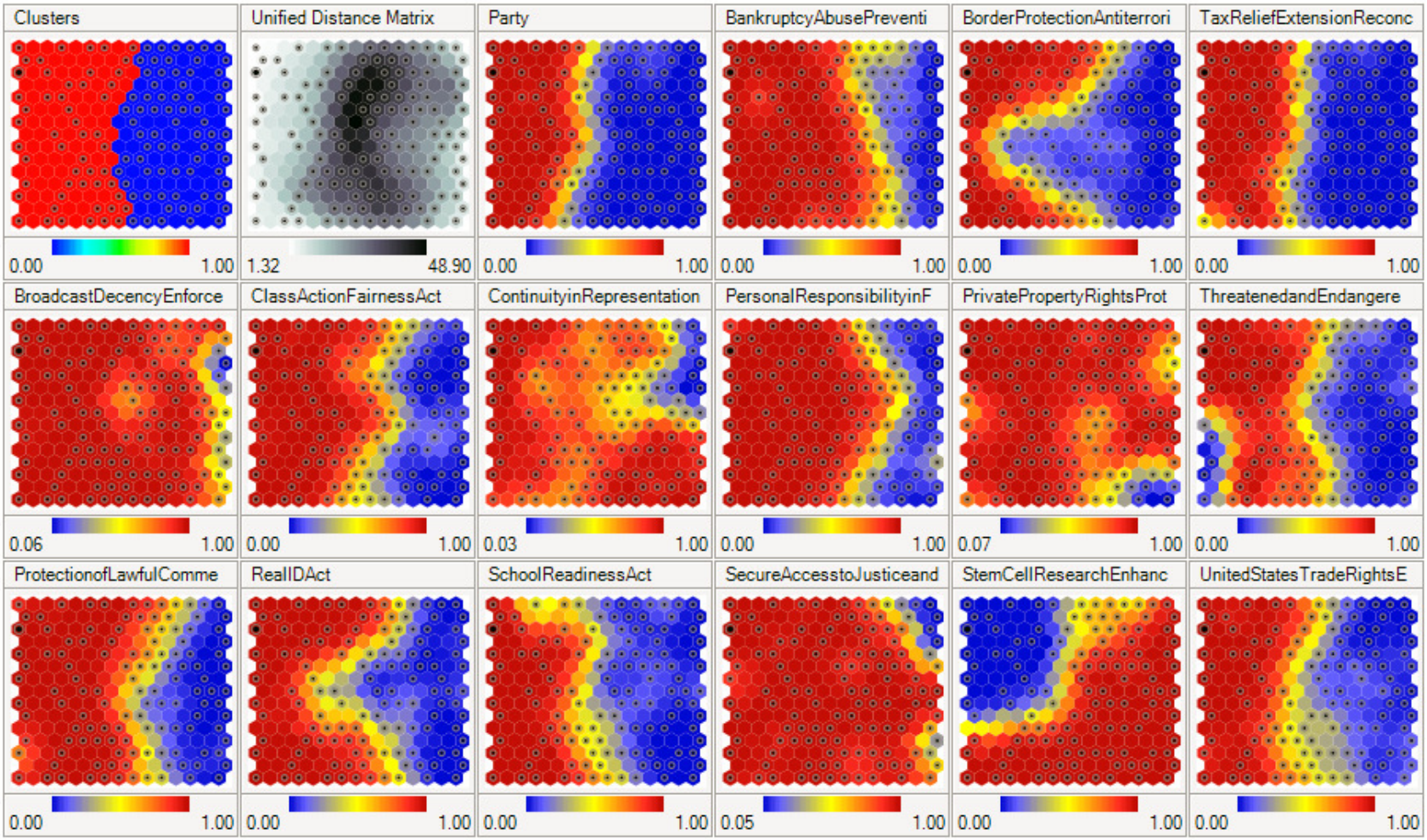






Интерпретация карт Кохонена (продолжение примера)

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

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

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

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

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:

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

Способы использования автокодировщиков

- Генерация признаков (feature generation)
- Снижение размерности (dimensionality reduction)
- Сжатие данных с минимальными потерями точности
- Более эффективное решение задач обучения с учителем в новом признаковом пространстве
- Обучаемая векторизация объектов, встраиваемая в более глубокие нейросетевые архитектуры
- Послойное предобучение многослойных сетей
- Генерация синтетических объектов, похожих на реальные

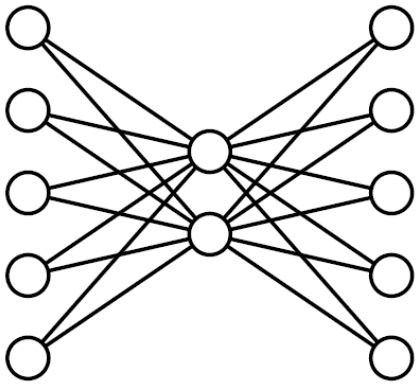
Rumelhart, Hinton, Williams. Learning Internal Representations by Error Propagation. 1986.

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

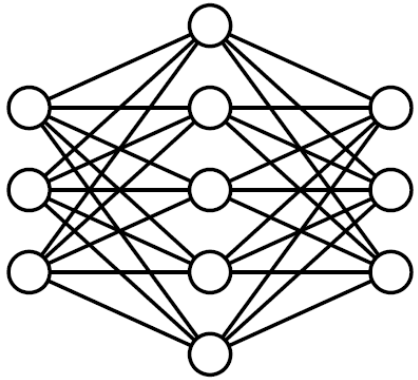
Архитектуры автокодировщиков

однослойный кодировщик/декодировщик

снижение размерности

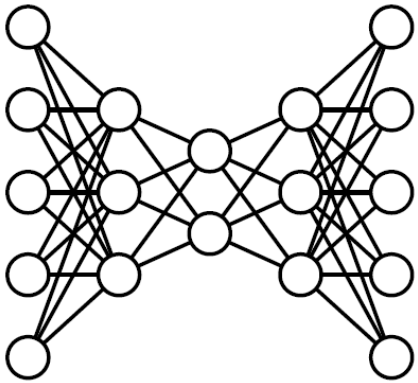


повышение размерности

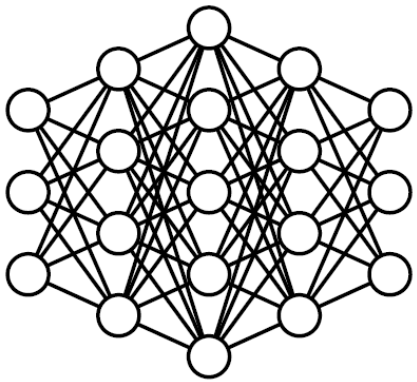


многослойный кодировщик/декодировщик

снижение размерности



повышение размерности



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: $F = (x_1 \dots x_\ell)^\intercal$, $U^\intercal U = I_m$, $G = FU$,

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

Автокодировщик обобщает метод главных компонент:

- не обязательно $B = A^\intercal$ (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
- SGD оптимизация вместо сингулярного разложения SVD

Разреживающий автокодировщик (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 $\bar{f}_j = \frac{1}{\ell}\sum_{i=1}^{\ell} f_j(x_i,\alpha)$; $\quad \varepsilon \in (0,1)$ — 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.

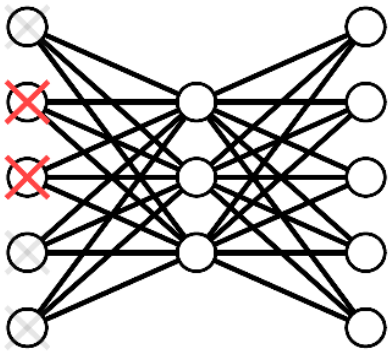
Шумоподавляющий автокодировщик (Denoising AE)

Устойчивость кодовых векторов z_i относительно шума в x_i:

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

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

- $\tilde{x} \sim \mathcal{N}(x_i, \sigma^2 I)$ — гауссовский шум
- обнуление компонент вектора x_i с вероятностью p_0:



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

Сжимающий автокодировщик (Contractive AE)

Устойчивость кодовых векторов z_i относительно шума в x_i:

$$\mathscr{L}_{\text{AE}}(\alpha,\beta) + \lambda \sum_{i=1}^{\ell} \|J_f(x_i)\|^2 \to \min_{\alpha,\beta}$$

где $\|J_f(x)\|$ — L_2-норма матрицы Якоби отображения $f: X \to Z$,

$$\|J_f(x)\|^2 = \sum_{d=1}^{n}\sum_{j=1}^{m}\left(\frac{\partial f_j(x,\alpha)}{\partial x_d}\right)^2$$

В случае $z = f(x,A) = \sigma(Ax+a)$, где σ — сигмоида, $A = (\alpha_{jd})$

$$\|J_f(x)\|^2 = \sum_{d=1}^{n}\sum_{j=1}^{m}\Big(\alpha_{jd} f_j(x,\alpha)\big(1-f_j(x,\alpha)\big)\Big)^2$$

Salah Rifai et al. Contractive auto-encoders: explicit invariance during feature extraction. ICML-2011.

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 $\hat{x}_i = g(f(x_i))$ — реконструкция объекта x_i,
$x_i^{\mathsf{T}}x_j$ — скалярное произведение (близость) пары объектов,
$\sigma(s) = (s - s_0)_+$ — функция активации ReLU с параметром s_0 (незначимые отношения близости не учитываются),
$\mathscr{L}(\hat{s}, s)$ — 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.

Вариационный автокодировщик (Variational AE)

Строится генеративная модель, способная порождать новые объекты x, похожие на объекты выборки $X^\ell = \{x_1, \ldots, x_\ell\}$

$q_\alpha(z|x)$ — вероятностный кодировщик с параметром α
$p_\beta(\hat{x}|z)$ — вероятностный декодировщик с параметром β

Максимизация нижней оценки log-правдоподобия:

$$\mathscr{L}_{\text{VAE}}(\alpha,\beta) = \sum_{i=1}^{\ell} \log p(x_i) = \sum_{i=1}^{\ell} \log \int q_\alpha(z|x_i) \frac{p_\beta(x_i|z)p(z)}{q_\alpha(z|x_i)} dz \geqslant$$

$$\geqslant \sum_{i=1}^{\ell} \int q_\alpha(z|x_i) \log \frac{p_\beta(x_i|z)p(z)}{q_\alpha(z|x_i)} dz =$$

$$= \sum_{i=1}^{\ell} \int q_\alpha(z|x_i) \log p_\beta(x_i|z) dz - \text{KL}\big(q_\alpha(z|x_i) \,\big\|\, p(z)\big) \to \max_{\alpha,\beta}$$

D.P.Kingma, M.Welling. Auto-encoding Variational Bayes. 2013.
C.Doersch. Tutorial on variational autoencoders. 2016.

Вариационный автокодировщик (Variational AE)

Оптимизационная задача для вариационного автокодировщика:

$$\sum_{i=1}^{\ell} \underbrace{\mathsf{E}_{z\sim q_\alpha(z|x_i)} \log p_\beta(x_i|z)}_{\substack{\text{качество реконструкции} \\ \approx \log p_\beta(x_i|z),\ z\sim q_\alpha(z|x_i)}} - \underbrace{\mathsf{KL}\big(q_\alpha(z|x_i) \,\big\|\, p(z)\big)}_{\text{регуляризатор по } \alpha} \to \max_{\alpha,\beta}$$

где $p(z)$ — априорное распределение, обычно $\mathcal{N}(0, \sigma^2 I)$

Репараметризация $q_\alpha(z|x_i)$: $\quad z = f(x_i, \alpha, \varepsilon),\ \varepsilon \sim \mathcal{N}(0, I)$

Метод стохастического градиента:

- сэмплировать $x_i \sim X^\ell,\ \ \varepsilon \sim \mathcal{N}(0, I),\ \ z = f(x_i, \alpha, \varepsilon)$
- градиентный шаг:
  $\alpha := \alpha + h\nabla_\alpha \big[\log p_\beta\big(x_i|f(x_i, \alpha, \varepsilon)\big) - \mathsf{KL}\big(q_\alpha(z|x_i)\|p(z)\big)\big];$
  $\beta := \beta + h\nabla_\beta \big[\log p_\beta(x_i|z)\big];$

Генерация похожих объектов: $x \sim p_\beta\big(x|f(x_i, \alpha, \varepsilon)\big),\ \ \varepsilon \sim \mathcal{N}(0, I)$

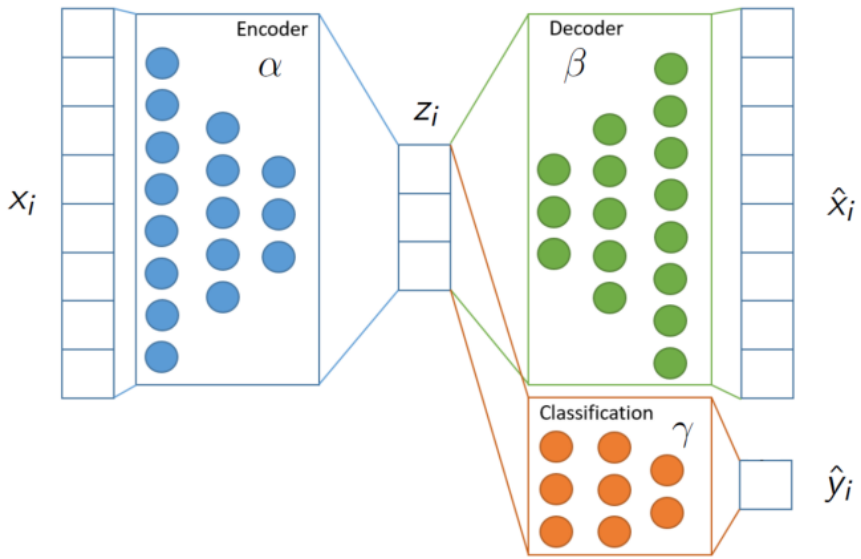
Автокодировщики для обучения с учителем

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

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

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

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



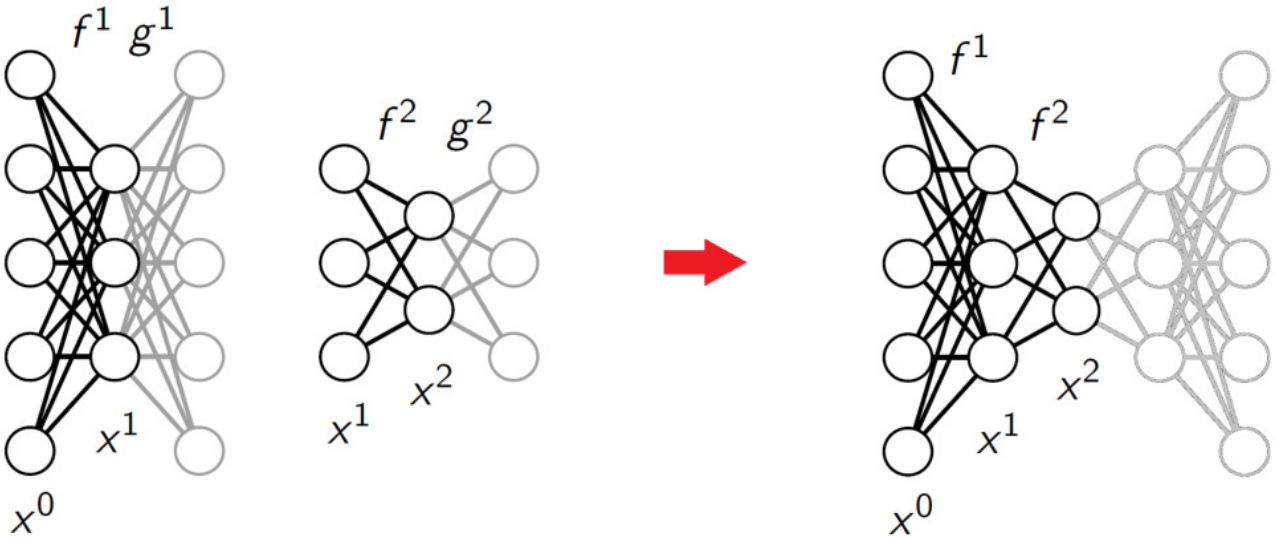
Функции потерь:
$\mathscr{L}(\hat{x}_i, x_i)$ — реконструкция
$\tilde{\mathscr{L}}(\hat{y}_i, y_i)$ — предсказание

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

Многослойный автокодировщик (Stacked AE)

Послойное обучение: $x^h = f^h(x^{h-1}, \alpha^h)$, $x \equiv x^0$, $z \equiv x^H$

- каждая пара f^h, g^h обучается по выборке $\{x_1^{h-1}, \ldots, x_\ell^{h-1}\}$
- декодировщик g^h отбрасывается
- однослойные $f^1, \ldots, f^H$ соединяются в H-слойный



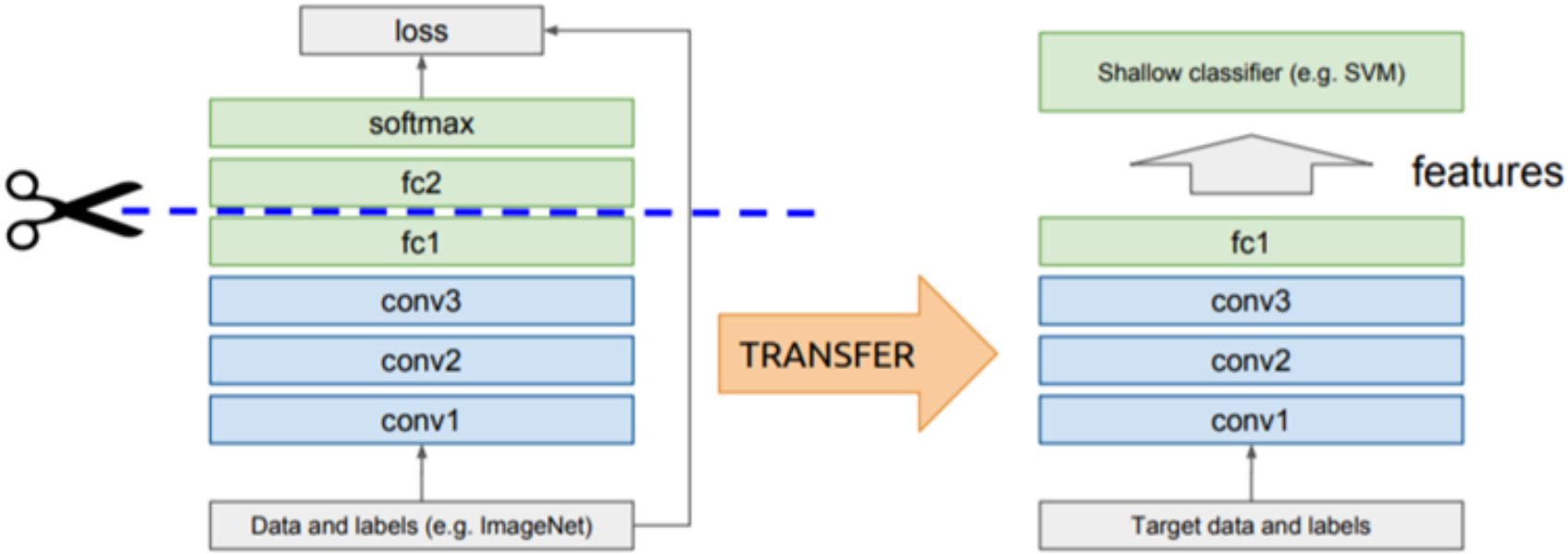
Тонкая настройка (fine tuning): результат послойного обучения используется как начальное приближение для BackProp

Y. Bengio et al. Greedy layer-wise training of deep networks. NIPS 2007.

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

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

$$\sum_{i=1}^{\ell} \mathscr{L}_i\big(f(x_i, \alpha), g(x_i, \beta)\big) \;\to\; \min_{\alpha,\beta}$$

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

$$\sum_{i=1}^{m} \mathscr{L}'_i\big(f(x'_i, \alpha), g'(x'_i, \beta')\big) \;\to\; \min_{\beta'}$$

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

$$\sum_{i=1}^{m} \mathscr{L}'_i\big(f(x'_i, \alpha), g'(x'_i, \beta')\big) \;\to\; \min_{\alpha,\beta'}$$

Sinno Jialin Pan, Qiang Yang. A Survey on Transfer Learning. 2009

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, $t \in T$:

$$\sum_{t \in T} \sum_{i \in X_t} \mathscr{L}_{ti}\big(f(x_{ti}, \alpha), g_t(x_{ti}, \beta_t)\big) \;\to\; \min_{\alpha, \{\beta_t\}}$$

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

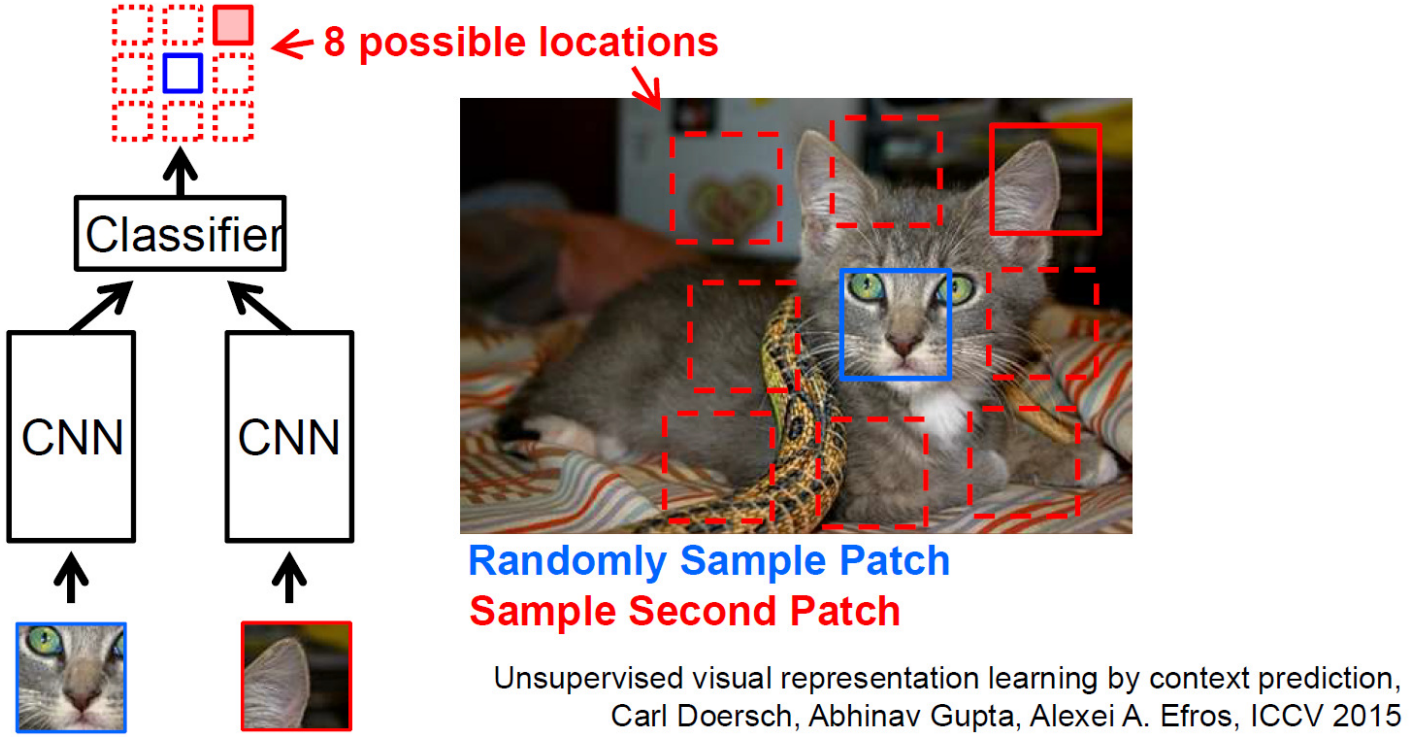
Learning to learn: качество решения каждой из задач $t \in T$ улучшается с ростом ℓ_t и общего числа задач $|T|$.

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



Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

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

Дистилляция моделей или суррогатное моделирование

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

$$\sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i, w), y_i\big) \;\to\; \min_{w}$$

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

$$\sum_{i=1}^{k} \mathscr{L}\big(b(x'_i, w'), a(x'_i, w)\big) \;\to\; \min_{w'}$$

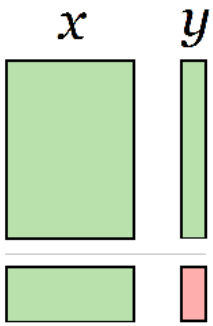
Примеры задач:

- 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

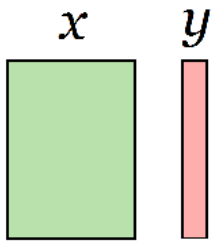
Обучение с использованием привилегированной информации

LUPI — Learning Using Privileged Information

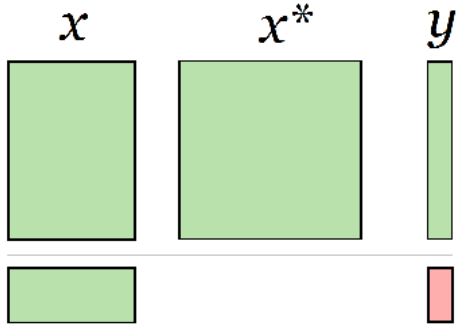
с учителем



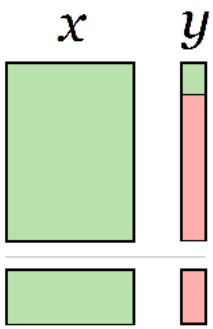
без учителя



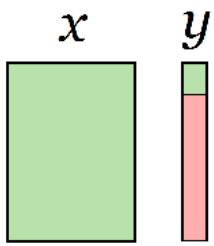
привилегированное (LUPI)



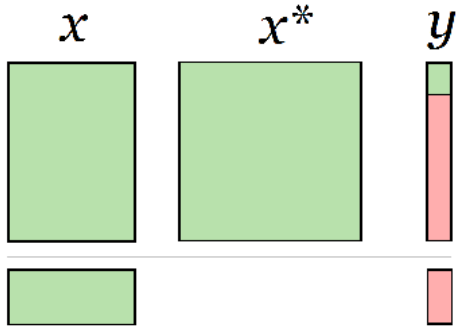
частичное



трансдуктивное



частичное LUPI



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

Примеры задач с привилегированной информацией x^*

- x — первичная (1D) структура белка
  x^* — третичная (3D) структура белка
  y — иерархическая классификация функции белка
- x — предыстория временного ряда
  x^* — информация о будущем поведении ряда
  y — прогноз следующей точки ряда
- x — текстовый документ
  x^* — выделенные ключевые слова или фразы
  y — категория документа
- x — пара (запрос, документ)
  x^* — выделенные асессором ключевые слова или фразы
  y — оценка релевантности

Задача обучения с привилегированной информацией

Раздельное обучение модели-ученика и модели-учителя:

$$\sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i, w), y_i\big) \to \min_w \qquad \sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i^*, w^*), y_i\big) \to \min_w$$

Модель-ученик обучается повторять ошибки модели-учителя:

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

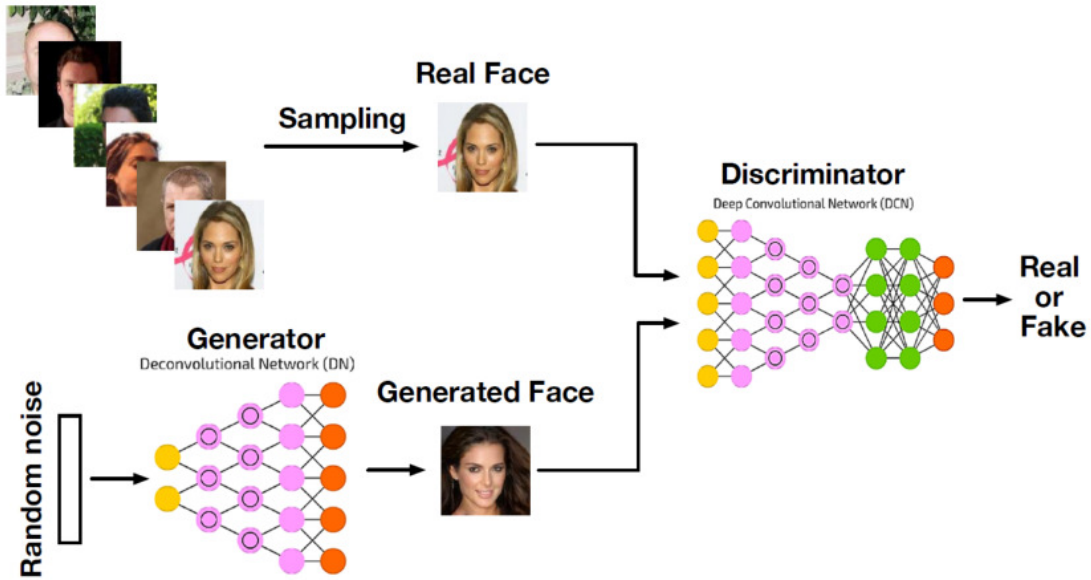
Совместное обучение модели-ученика и модели-учителя:

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

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

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, Qi She, Tomas Ward. 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:

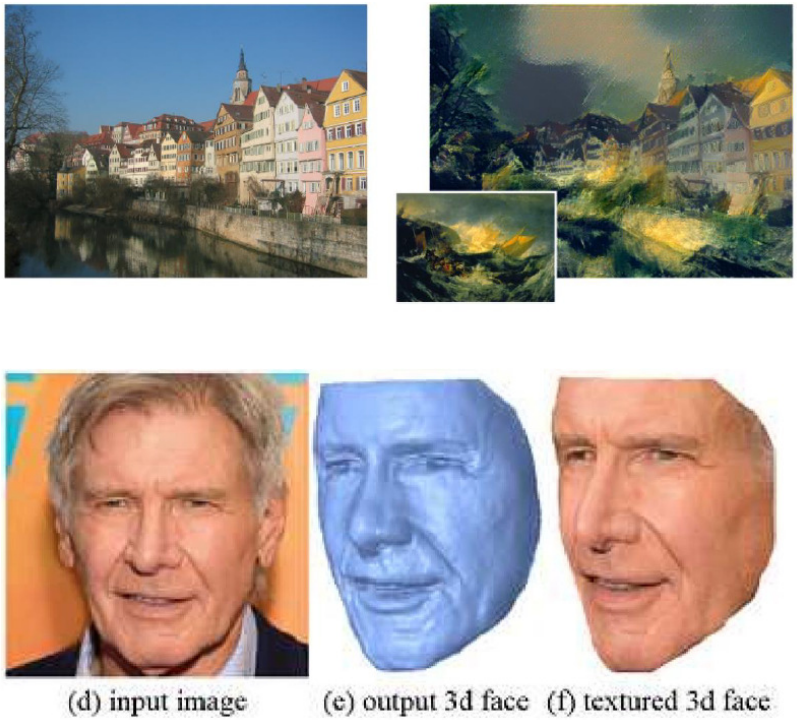
$$\sum_{i=1}^{m} \ln D(x_i, \beta) + \ln\big(1 - D\big(G(z_i, \alpha), \beta\big)\big) \;\to\; \max_{\beta}$$

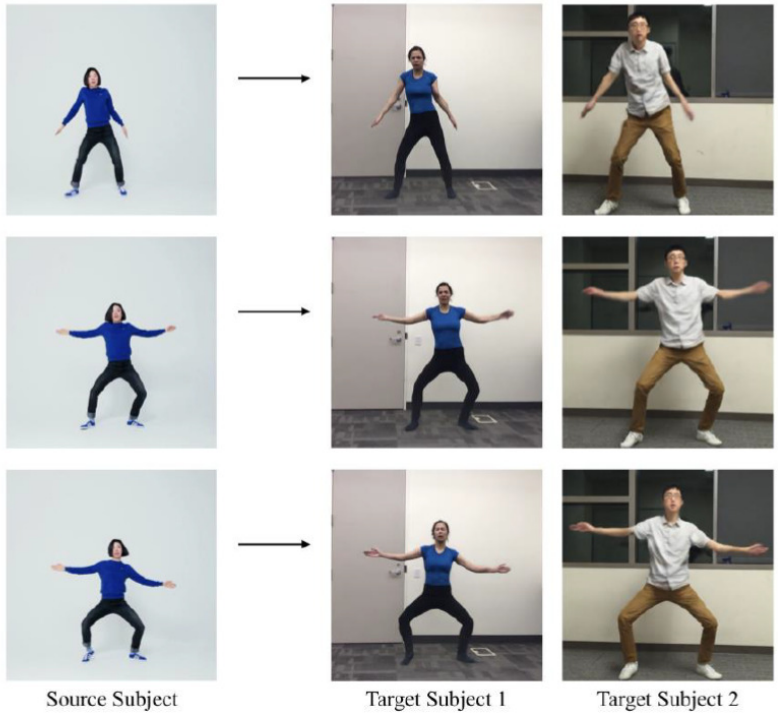
обучение генеративной модели G по случайному шуму $\{z_i\}_{i=1}^m$:

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

Ian Goodfellow et al. Generative Adversarial Nets. 2014

Примеры GAN для синтеза изображений и видео





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.

Резюме

Когда несколько моделей обучаются одновременно:

- Автокодировщики: кодер и декодер
- Автокодировщики для классификации или кластеризации
- Многозадачное обучение
- Обучение с привилегированной информацией
- Состязательные сети (GAN) для генерации фейк-объектов

Когда несколько моделей обучаются последовательно:

- Перенос обучения (transfer learning)
- Предобучение глубоких сетей для векторизации объектов
- Самостоятельное обучение (self-supervised learning)
- Дистилляция и суррогатное моделирование

Обучение без учителя всё чаще используется для оптимизации части модели по большим данным