Машинное обучение (Machine Learning)

Передача знаний и адаптация данных (Transfer Learning and Domain Adaptation)

Уткин Л.В.

Санкт-Петербургский политехнический университет Петра Великого

ПОЛИТЕХ
Санкт-Петербургский политехнический университет Петра Великого

Содержание

Определение Transfer Learning

Transfer Learning - определение

Из Wikipedia:

Transfer Tearning (Inductive transfer) is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.

Мотивация из жизни:

Человек может применить знания, полученные ранее, для более быстрого или качественного решения новых задач!

Transfer Learning - мотивация из жизни

Мы часто используем в жизни знания в новых ситуациях:

- Шахматы → Шашки
- C++ → Java
- Физика/Математика → Компьютерные науки

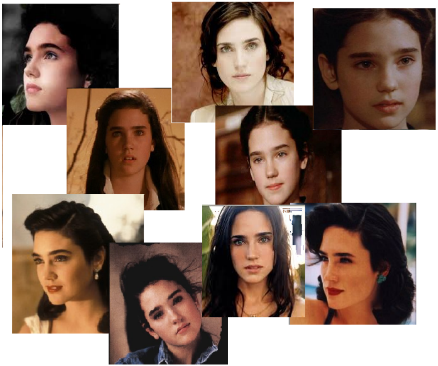
***Transfer Learning**: Способность системы распознавать и применять знания и умения, полученные в предыдущих задачах, к новым задачам или данным.*

Два источника данных

1. Исходные данные (**sourse data**) - их много (**big data**), но не совсем то, что надо
2. Целевые данные (**target data**) - их мало (**small data**), но они соответствуют условиям задачи
3. Цель transfer learning выявить знания из исходных данных и применить их к целевым данным

Transfer Learning - иллюстрация

source data



target data

Другие примеры

- Классификация Web-страниц по категориям:
    - пусть имеется классификатор, обученный на университетских сайтах;
    - для задачи с новым сайтом, где признаки и распределение данных отличны от известных, не всегда можно непосредственно применять классификатор обученный на “университетах”.

Другие примеры

- Задача классификации отзывов - автоматически классифицировать отзывы о продукте (положит. или отрицат.):
    - необходимо собрать много отзывов о продукте, дать метку класса и обучить классификатор
    - так как продуктов много и отзывы различны, то очень дорого их собирать и оценивать.
    - однако можно адаптировать классификатор, обученный на некоторых продуктах, к другим продуктам при помощи Transfer learning.

Типы моделей

Типы моделей Transfer Learning с точки зрения цели

1. **Асимметричная передача**:
   - Большое количество данных с метками классов в нескольких сходных задачах
   - **Цель:** Повысить качество целевой задачи, для которой данных мало
2. **Симметричная передача**:
   - Малое количество обучающих данных для большого числа сходных задач
   - **Цель:** Повысить качество в среднем по всем классификаторам

Типы моделей Transfer Learning

1. **Inductive transfer learning**: есть метки исходных данных, есть метки целевых данных
2. **Transductive transfer learning**: есть метки исходных данных, нет меток целевых данных; признаки целевых и исходных данных различны; признаки одинаковы, но распределения вероятностей различны (**domain adaptation**).
3. **Unsupervised (без учителя) transfer learning**: нет меток классов как для целевых данных, так и для исходных.

Различные типы Transfer Learning

тип TL	области	метки данных исходных	метки данных целевых
inductive	многозадачность	есть	есть
	самообучение	нет	есть
transductive	domain adaptation	есть	нет
unsupervised		нет	нет

Формальное определение Transfer Learning

- Область $\mathcal{D} = \{\mathcal{X}, P(X)\}$ определяется двумя элементами, пространством признаков $\mathcal{X}$ и распределением вероятностей $P(X)$, где пример $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\} \in \mathcal{X}$.
- Для данной области $\mathcal{D}$ задача $\mathcal{T} = \{\mathcal{Y}, f(\cdot)\}$ определяется двумя элементами, пространством меток $\mathcal{Y}$ и функцией $f(\cdot)$, которая вычисляется на основе пары $\{\mathbf{x}_i, y_i\}$, где $\mathbf{x}_i \in X$ и $y_i \in \mathcal{Y}$.
- Исходная область $\mathcal{D}_S = \{(\mathbf{x}_1^S, y_1^S), ..., (\mathbf{x}_n^S, y_n^S)\}$, $\mathbf{x}_i^S \in \mathcal{X}_S$ и $y_i^S \in \mathcal{Y}_S$, исходная задача $\mathcal{T}_S$
- Целевая область $\mathcal{D}_T = \{(\mathbf{x}_1^T, y_1^T), ..., (\mathbf{x}_n^T, y_n^T)\}$, $\mathbf{x}_i^T \in \mathcal{X}_T$ и $y_i^T \in \mathcal{Y}_T$, целевая задача $\mathcal{T}_T$

Формальное определение Transfer Learning

- Transfer learning - это процесс улучшения целевой функции $f_T(\cdot)$ (классификатор), используя информацию из $\mathcal{D}_S$ и $\mathcal{T}_S$, где $\mathcal{D}_S \neq \mathcal{D}_T$ или $\mathcal{T}_S \neq \mathcal{T}_T$.
- Так как $\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\}$ и $\mathcal{D}_T = \{\mathcal{X}_T, P(X_T)\}$, то условие $\mathcal{D}_S \neq \mathcal{D}_T$ означает $\mathcal{X}_S \neq \mathcal{X}_T$ и/или $P(X_S) \neq P(X_T)$.
- Случай $\mathcal{X}_S \neq \mathcal{X}_T$ - **гетерогенный** transfer learning
- Случай $\mathcal{X}_S = \mathcal{X}_T$ - **гомогенный** transfer learning

Transfer Learning (другими словами)

- n_T наблюдений в экстремальных (актуальных) условиях: $\mathcal{D}_T = \{(\mathbf{x}_1^T, y_1^T), ..., (\mathbf{x}_n^T, y_n^T)\}$ - **малая выборка**
- n_S наблюдений в нормальных условиях: $\mathcal{D}_S = \{(\mathbf{x}_1^S, y_1^S), ..., (\mathbf{x}_n^S, y_n^S)\}$ - **большая выборка**
- **Как, используя $\mathcal{D}_S$, работать с $\mathcal{D}_T$ и построить классификатор, ориентированный на $\mathcal{D}_T$?**

Различные подходы к Transfer Learning

Основаны на том, “что передавать” от исходных данных целевым (4 случая):

- Передача примеров (instance-transfer)
- Передача представления признаков (feature-representation-transfer)
- Передача параметров (parameter-transfer)
- Передача относительных знаний (relational-knowledge-transfer)

Передача примеров (instance-transfer)

- Предполагает, что определенная часть данных из $\mathcal{D}_S$ может быть передана для обучения в $\mathcal{D}_T$ посредством переназначения их весов или при помощи метода значимой выборки

Передача представления признаков (feature-representation-transfer)

- Цель - получить “хорошее” представление для $\mathcal{D}_T$
- Знания, используемые для передачи, кодируются в определенное представление признаков
- С новым представлением признаков характеристики целевой задачи $\mathcal{T}_T$ могут быть значительно улучшены

Передача параметров (parameter-transfer)

- Предполагается, что исходная задача $\mathcal{T}_S$ и целевая задача $\mathcal{T}_T$ имеют общие параметры θ моделей или априорные распределения параметров $f_S(\cdot)$ и $f_T(\cdot)$
- Передаваемые данные кодируются так, чтобы оставить только общие параметры или признаки
- Определив общие параметры, данные могут передаваться между задачами

Передача относительных знаний (relational-knowledge-transfer)

-
- Некоторое соотношение между данными в $\mathcal{D}_T$ и $\mathcal{D}_S$ аналогичны
- Знания, которые передаются, являются этими соотношениями.

Различные типы Transfer Learning

Передача:	inductive	transductive	unsupervised
примеров	SVM TrAdaBoost	Sample Reweiting	
представления признаков	SVM sparse coding	SCL	STC
параметров	Regularization		
относительных знаний	TAMAR		

TAMAR - Transfer via Automatic Mapping and Revision

SCL - Structural Correspondence Learning

STC - Self-Taught Clustering

Inductive Transfer Learning

Передача примеров и SVM для Inductive TL (1)

Мы хотим получить разделяющую функцию $f_T(\cdot)$. Как?

1. Можно игнорировать $\mathcal{D}_S$ и использовать стандартный SVM для $\mathcal{D}_T$ и $\mathcal{T}_T$
   - Это хороший подход? Нет, мы теряем $\mathcal{D}_S$ и $\mathcal{T}_S$.
2. Можно учесть $\mathcal{D}_S$ и $\mathcal{T}_S$ и обучиться, используя $\mathcal{D}_S$, $\mathcal{T}_S$, $\mathcal{D}_T$ и $\mathcal{T}_T$ одновременно
   - Это хороший подход? Лучше.
   - Но как это сделать?

Передача примеров и SVM для Inductive TL (2)

Простейший подход - использовать данные из обоих множеств $\mathcal{D}_S = \{(\mathbf{x}_i^S, y_i^S)\}$ и $\mathcal{D}_T = \{(\mathbf{x}_i^T, y_i^T)\}$

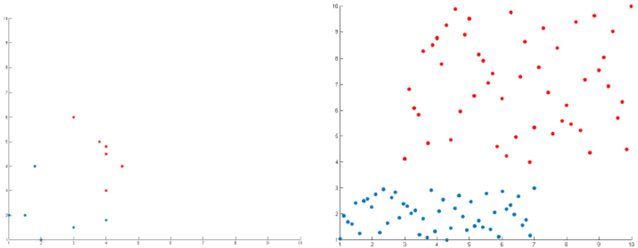
$$\min_{w, \xi_i^{(T)}, \xi_i^{(S)}} J = \|w\|^2 + \lambda_1 \sum_{i=1}^{n_T} \xi_i^{(T)} + \lambda_2 \sum_{i=1}^{n_S} \xi_i^{(S)}$$

при ограничениях

$$y_i^S \cdot w \cdot \mathbf{x}_i^S \geq 1 - \xi_i^{(S)}, \;\; \xi_i^{(S)} \geq 0, \;\; i = 1, \dots, n_S,$$

$$y_i^T \cdot w \cdot \mathbf{x}_i^T \geq 1 - \xi_i^{(T)}, \;\; \xi_i^{(T)} \geq 0, \;\; i = 1, \dots, n_T.$$

Передача примеров и SVM для Inductive TL (3)



- Мы хотим получить $f_T(\cdot)$ с учетом $\mathcal{D}_S$ и $\mathcal{T}_S$ и обучаем, используя одновременно $\mathcal{D}_S$, $\mathcal{T}_S$, $\mathcal{D}_T$ и $\mathcal{T}_T$.
- Это не очень хорошая идея.

Передача примеров и SVM для Inductive TL (4)

- Главное заключается в том, что некоторые из $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$ полезны для $f_T(\cdot)$, а другие могут наоборот все испортить
- Необходимо выбрать $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$, которые полезны и выбросить остальные
- Один из методов - назначить веса ρ_i примеров $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$ в соответствии с их значимостью для $f_T(\cdot)$

Передача примеров и SVM для Inductive TL (5)

- Назначаем веса ρ_i примерам $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$ в соответствии с их значимостью для $f_T(\cdot)$

$$\min_{w,\xi_i^{(T)},\xi_i^{(S)}} J = \|w\|^2 + \lambda \sum_{i=1}^{n_T} \xi_i^{(T)} + \lambda \sum_{i=1}^{n_S} \rho_i \xi_i^{(S)}$$

при ограничениях

$$y_i^S \cdot w \cdot \mathbf{x}_i^S \geq 1 - \xi_i^{(S)},\ \ \xi_i^{(S)} \geq 0,\ \ i = 1, \ldots, n_S,$$

$$y_i^T \cdot w \cdot \mathbf{x}_i^T \geq 1 - \xi_i^{(T)},\ \ \xi_i^{(T)} \geq 0,\ \ i = 1, \ldots, n_T.$$

- Как определить веса ρ_i?

Передача примеров и SVM для Inductive TL (6)

- ρ_i - вес точки $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$, который можно оценить при помощи эвристических методов
- Например, $\rho_i = \sigma\left((\mathbf{x}_i^S, y_i^S), \mathcal{D}_T\right)$, где

$$\sigma\left((\mathbf{x}_i^S, y_i^S),\right) = \frac{1}{|\mathcal{D}_T|}\sum_{j=1}^{|\mathcal{D}_T|} \exp\left\{-\beta \left\|(\mathbf{x}_i^S, y_i^S) - (\mathbf{x}_j^T, y_j^T)\right\|^2\right\}$$

- Только одно различие между стандартным SVM и SVM с передачей примеров: $\lambda \sum_{i=1}^{n_S} \rho_i \xi_i^{(S)}$

Передача параметров и SVM для Inductive TL (1)

Ключевые идеи:

- Задачи $\mathcal{T}_S$ и $\mathcal{T}_T$ связаны друг с другом каким-то образом
- Связь формализуется посредством связи параметров в SVM
- Например, можно предположить, что все параметры w_T и w_S имеют нормальное распределение вероятностей
- Тогда w_T и w_S “близки” к некоторому среднему вектору параметров w_0

Передача параметров и SVM для Inductive TL (2)

- **Параметры**:
  - $w_S = w_0 + v_S$ и $w_T = w_0 + v_T$, где w_S и w_T - параметры SVM для $\mathcal{T}_S$ и $\mathcal{T}_T$;
  - w_0 - общие параметры;
  - v_S и v_T - специфичные параметры SVM для $\mathcal{T}_S$ и $\mathcal{T}_T$.
- **Предположение**: $f_T = w_T \cdot \mathbf{x}$
- **Модификация SVM**:

$$\min_{w_0, v_T, \xi_{r_i}} J = \sum_{r \in \{S,T\}} \sum_{i=1}^{n_T} \xi_{r_i} + \frac{\lambda_1}{2} \sum_{r \in \{S,T\}} \left\| v_r \right\|^2 + \lambda_2 \left\| w_0 \right\|^2$$

при ограничениях

$$y_i^T (w_0 + v_T) \cdot \mathbf{x}_i^T \geq 1 - \xi_{r_i}, \ \xi_{r_i} \geq 0,$$
$$i \in \{1, 2, ..., n_T\}, \ r \in \{S, T\}.$$

Обобщение на многозадачную ситуацию

- t источников данных с параметрами $w_i = w_0 + v_i$, $i = 1, ..., t$; w_0 - общие параметры
- **Модификация SVM**:

$$\min_{w_0, v_T, \xi_{r_i}} J = \sum_{j=1}^{t} \sum_{i=1}^{n_j} \xi_j^{(i)} + \frac{\lambda_1}{t} \sum_{j=1}^{t} \left\| v_j \right\|^2 + \lambda_2 \left\| w_0 \right\|^2$$

при ограничениях

$$y_j^{(i)}(w_0 + v_j) \cdot \mathbf{x}_j^{(i)} \geq 1 - \xi_j^{(i)}, \;\; \xi_j^{(i)} \geq 0,$$
$$i \in \{1, 2, ..., n_j\}, \;\; j = 1, ..., t$$

- $f_j = w_j \cdot \mathbf{x}$

Без учителя

Передача представления признаков без учит. (1)

- В самообучении используются данные без меток классов для повышения качества данных с метками классов
- Главное предположение - данные без меток содержат основную структуру, которая представлена в данных с метками классов
- Цель — сделать обучение проще и менее затратным

Передача представления признаков без учит. (2)

- $\mathcal{D}_T = \{(\mathbf{x}_i^T, y_i^T)\}$, $\mathbf{x}_i^T \in \mathbb{R}^d$, $y^T \in \{1, ..., C\}$
- $\mathcal{D}_S = \{\mathbf{x}_i^S\}$, $\mathbf{x}_i^S \in \mathbb{R}^d$
- Использовать $\mathcal{D}_S$ для улучшения $f_T(\cdot)$
- Raina R., Battle A., Lee H., Packer B. and Ng A.Y. Self-taught Learning: Transfer Learning from Unlabeled Data. ICML. Corvallis, OR, USA, 2007.

Передача представления признаков без учит. (3)

- Решаем следующую задачу оптимизации на $\mathcal{D}_S$:

$$\min_{\mathbf{b},\mathbf{a}} \sum_{i=1}^{n_S} \left\| \mathbf{x}_i^S - \sum_{j=1}^{s} \mathbf{a}_j^{(i)} \mathbf{b}^{(j)} \right\|_2^2 + \beta \left\| \mathbf{a}^{(i)} \right\|_1$$

при ограничениях $\left\| \mathbf{b}^{(j)} \right\|_2 \leq 1$

- Переменные оптимизации
  - $\mathbf{b}^{(j)}$: **базисный вектор** $\mathbf{b}^{(j)} \in \mathbb{R}^d$
  - $\mathbf{a}^{(i)}$: вектор **активаций** $\mathbf{a}^{(i)} \in \mathbb{R}^s$ вектора $\mathbf{b}^{(j)}$ для $\mathbf{x}_i^S$
- 1-е слагаемое реконструирует $\mathbf{x}_i^S$ как весовую линейную комбинацию базисных векторов $\mathbf{b}^{(j)}$ с весами $\mathbf{a}^{(i)}$
- 2-е слагаемое ограничивает веса $\mathbf{a}^{(i)}$ единичной нормой - получаем разреженные веса - **высокоуровневое представление**

Передача представления признаков без учит. (4)

- Конструирование признаков
- Для каждой точки $(\mathbf{x}_i^T, y_i^T)$, вычисляем признаки $\hat{a}(\cdot) \in \mathbb{R}^d$, решая задачу

$$\min_{\mathbf{a}^{(i)}} \left\| \mathbf{x}_i^T - \sum_{j=1}^{s} \mathbf{a}_j^{(i)} \mathbf{b}^{(j)} \right\|_2^2 + \beta \left\| \mathbf{a}^{(i)} \right\|_1$$

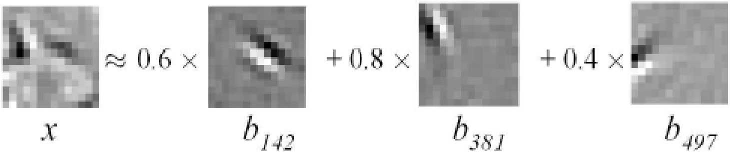
  при ограничениях $\left\| \mathbf{b}^{(j)} \right\|_2 \leq 1$
- Разреженный вектор $\mathbf{a}_j^{(i)}$ - новое **реконструированное представление** вектора $\mathbf{x}_i^T$

Алгоритм TL без учит.

1. Input: $\{(\mathbf{x}_i^T, y_i^T)\}$, $\{\mathbf{x}_i^S\}$
2. Используя $\{\mathbf{x}_i^S\}$, решаем $\min_{\mathbf{b},\mathbf{a}} \sum_{i=1}^{n_S} \left\| \mathbf{x}_i^S - \sum_{j=1}^{s} \mathbf{a}_j^{(i)} \mathbf{b}^{(j)} \right\|_2^2 + \beta \left\| \mathbf{a}^{(i)} \right\|_1$ при ограничениях $\left\| \mathbf{b}^{(j)} \right\|_2 \leq 1$
3. Для $(\mathbf{x}_i^T, y_i^T)$, вычисляем признаки $\hat{a}(\mathbf{x}_i^T) = \min_{\mathbf{a}^{(i)}} \left\| \mathbf{x}_i^T - \sum_{j=1}^{s} \mathbf{a}_j^{(i)} \mathbf{b}^{(j)} \right\|_2^2 + \beta \left\| \mathbf{a}^{(i)} \right\|_1$
4. Output: training set $(\hat{a}(\mathbf{x}_i^T), y_i^T)$ вместо $(\mathbf{x}_i^T, y_i^T)$

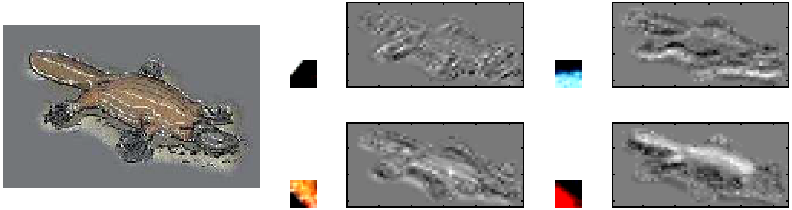
Пример TL без учит. (1)

Пример представления кусочка изображения **x** как разреженной весовой комбинации базовых векторов



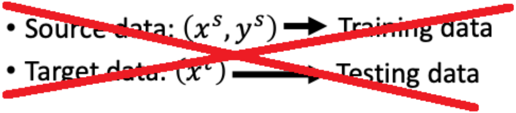
Пример TL без учит. (2)

Признаки, вычисленные для изображения крокодила, используя 4 базовых изображения



Transductive Transfer Learning

Transductive TL



с метками

без меток

Transductive TL

- **Интуиция**: Так как $\mathcal{T}_S$ и $\mathcal{T}_T$ одинаковы, то для получения $f_T(\cdot)$ можно адаптировать функцию $f_S(\cdot)$ для использования в $\mathcal{T}_T$ на данных из $\mathcal{D}_T$
- Передаются примеры (1) и представления признаков (2)

Передача примеров в Transductive TL (1)

- Опять стандартный SVM:

$$\min_{w,\xi} J = \frac{1}{2}\|w\|_2^2 + \lambda \sum_{i=1}^{n} \xi_i$$

при ограничениях

$$y_i \cdot w^{\mathrm{T}} \cdot \mathbf{x}_i \geq 1 - \xi_i, \;\; \xi_i \geq 0, \;\; i = 1, \ldots, n$$

- Разделяющая функция:

$$f(\mathbf{x}_i) = w^{\mathrm{T}} \cdot \mathbf{x}_i = \sum_{j=1}^{m} w_j \mathbf{x}_{ij}$$

Передача примеров в Transductive TL (2)

- Ключевая идея - некоторые $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$ могут помочь в обучении $f_T(\cdot)$, в то время как другие только делают модель хуже
- Следовательно, нужно выбрать те $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$, которые полезны, и выкинуть те, которые мешают
- Путь - назначить веса $(\mathbf{x}_i^S, y_i^S) \in \mathcal{D}_S$, отражающие значимость для обучения $f_T(\cdot)$
- Что-то знакомое уже было (Inductive TL)

Передача примеров в Transductive TL (3)

- Веса ρ_i:

$$\min_{w,\xi_i^{(T)},\xi_i^{(S)}} J = \frac{1}{2}\|w\|^2 + \lambda \sum_{i=1}^{n} \rho_i \xi_i^{(S)}$$

при ограничениях

$$y_i^S \cdot w \cdot \mathbf{x}_i^S \geq 1 - \xi_i^{(S)}, \ \ \xi_i^{(S)} \geq 0, \ \ i = 1, \ldots, n$$

- например, $\rho_i = \sigma\left((\mathbf{x}_i^S, y_i^S), \mathcal{D}_T\right)$, где

$$\sigma\left((\mathbf{x}_i^S, y_i^S), \right) = \frac{1}{|\mathcal{D}_T|} \sum_{j=1}^{|\mathcal{D}_T|} \exp\left\{-\beta \left\|\mathbf{x}_i^S - \mathbf{x}_j^T\right\|^2\right\}$$

Общий подход для Transductive TL (1)

- Оптимальные параметры θ^*:

$$\theta^* = \arg\min_{\theta \in \Theta} \mathbb{E}_{(x,y)\in P}\left[l(\mathbf{x}, y, \theta)\right],$$

где $l(\mathbf{x}, y, \theta)$ - функция потерь (зависит от θ)

$$\theta^* = \arg\min_{\theta \in \Theta} \sum_{(x,y)\in \mathcal{D}_T} P(\mathcal{D}_T) l(\mathbf{x}, y, \theta).$$

Общий подход для Transductive TL (2)

- Так как нет целевых данных с метками классов, необходимо обучать модель yf исходных данных:

$$\theta^* = \arg\min_{\theta\in\Theta} \sum_{(x,y)\in\mathcal{D}_S} \frac{P(\mathcal{D}_T)}{P(\mathcal{D}_S)} P(\mathcal{D}_S) l(\mathbf{x}, y, \theta)$$
$$\approx \arg\min_{\theta\in\Theta} \sum_{i=1}^{n_S} \frac{P_T(\mathbf{x}_i^T, y_i^T)}{P_S(\mathbf{x}_i^S, y_i^S)} l(\mathbf{x}_i^S, y_i^S, \theta)$$
$$= \arg\min_{\theta\in\Theta} \sum_{i=1}^{n_S} \frac{P(\mathbf{x}_i^S)}{P(\mathbf{x}_i^T)} l(\mathbf{x}_i^S, y_i^S, \theta)$$

- Это следует из условия $P(Y_T|X_T) = P(Y_S|X_S)$. Т.о. разность между $P(\mathcal{D}_S)$ и $P(\mathcal{D}_T)$ определяется только $P(X_T)$ и $P(X_S)$ и $\frac{P(\mathbf{x}_i^S)}{P(\mathbf{x}_i^T)}$

Transductive TL и SVM (1)

Необходимо решить три задачи:

1. Минимизация функционала риска на области $\mathcal{D}_S$
2. Минимизация разности между двумя совместными распределениями вероятностей J_s и J_t
3. Максимизация согласованности маргинальных распределений P_s и P_t

Transductive TL и SVM (2)

$$f = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n_s} l(f(\mathbf{x}_i^s), y_i^s) + \sigma \|f\|_K^2$$
$$+ \lambda D_{f,K}(J_s, J_t) + \gamma M_{f,K}(P_s, P_t)$$

K - ядро

σ, λ, γ - положительные параметры регуляризации (ограничения на f)

Первая часть - обычный SVM для исходных (source) данных

А что такое $D_{f,K}(J_s, J_t)$ и $M_{f,K}(P_s, P_t)$?

Transductive TL и SVM (3)

Минимизация разности между двумя совместными распределениями вероятностей J_s и J_t или вычисление $D_{f,K}(J_s, J_t)$

Адаптация маргинальных распределений: (используется разность средних значений функций или MMD - maximum mean discrepancy)

$$D_{f,K}(P_s, P_t) = \left\| \frac{1}{n_s} \sum_{i=1}^{n_s} f(\mathbf{x}_i^s) - \frac{1}{n_t} \sum_{i=1}^{n_t} f(\mathbf{x}_i^t) \right\|_{\mathcal{H}}^2$$

$\mathcal{H}$ определяется $\phi : \mathcal{X} \to \mathcal{H}$

Transductive TL и SVM (4)

Вычисление $D_{f,K}(J_s, J_t)$
Адаптация условных распределений:

$$D^{()}_{f,K}(Q_s, Q_t) = \left\| \frac{1}{n_s^{(c)}} \sum_{i \in \mathcal{D}_s^{(c)}}^{n_s^{(c)}} f(\mathbf{x}_i^s) - \frac{1}{n_t^{(c)}} \sum_{i \in \mathcal{D}_t^{(c)}}^{n_t^{(c)}} f(\mathbf{x}_i^s) \right\|_{\mathcal{H}}^2$$

$\mathcal{D}_s^{(c)}$ - множество примеров из класса c, принадлежащих $\mathcal{D}_s$
$\mathcal{D}_t^{(c)}$ - множество примеров из класса c, принадлежащих $\mathcal{D}_t$, здесь используются псевдо метки классов (примерные)

Transductive TL и SVM (5)

Минимизация разности между двумя совместными распределениями вероятностей J_s и J_t или вычисление $D_{f,K}(J_s, J_t)$

$$D_{f,K}(J_s, J_t) = D_{f,K}(P_s, P_t) + \sum_{=1} D^{()}_{f,K}(Q_s, Q_t)$$

Transductive TL и SVM (6)

- $D_{f,K}(J_s, J_t)$ основана на использовании выборочного мат. ожидания
- Максимизация согласованности маргинальных распределений P_s и P_t основана на использовании выборочной дисперсии:

$$M_{f,K}(P_s, P_t) = \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \left(f(\mathbf{x}_i^s) - f(\mathbf{x}_j^t)\right)^2 W_{ij},$$

где

$$W_{ij} = \begin{cases} \cos(\mathbf{x}_i^s, \mathbf{x}_j^t), & \mathbf{x}_i^s \in \mathcal{N}_p(\mathbf{x}_j^t) \vee \mathbf{x}_j^t \in \mathcal{N}_p(\mathbf{x}_i^s) \\ 0, & \text{иначе} \end{cases}$$

- $\mathcal{N}_p(\mathbf{x}_i)$ - множество p ближайших соседей точки $\mathbf{x}_i$

Transductive TL и SVM (7)

В итоге задача

$$f = \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{n_s} l(f(\mathbf{x}_i^s), y_i^s) + \sigma \|f\|_K^2$$
$$+ \lambda D_{f,K}(J_s, J_t) + \gamma M_{f,K}(P_s, P_t)$$

сводится к стандартной задаче квадратичного программирования

M.Long, J.Wang, G.Ding, S.J.Pan, P.S.Yu Adaptation Regularization: A general Framework for Transfer Learning. IEEE Trans. on Knowledge and Data Eng., vol. 26(5), pp. 1076-1089, 2014

Методы deep domain adaptation

Методы deep domain adaptation - 3 группы:

1. **Discrepancy-based**: подходы основаны на минимизации расстояния между векторными представлениями на исходном и целевом доменах с помощью введения этого расстояния в loss-функцию.
2. **Adversarial-Based**: подходы используют состязательную (adversarial) loss-функцию (из GAN), для обучения сети, инвариантной относительно домена.
3. **Смешанные методы:** применяют идеи из discrepancy-based семейства, а также self-ensembling, новые слои, loss-функции и т.п.

методы

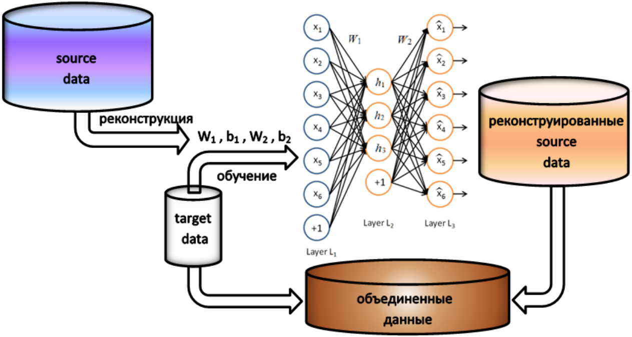
- **Discrepancy-based**: основан на metric learning, суть которого заключается в обучении такого векторного **представления, получаемого из нейронной сети**, что представители одного класса будут близки друг к другу в этом представлении по заданной метрике (чаще всего используют L^2 или косинусную метрики).

Автокодер для Transductive TL

- Glorot X, Bordes A, Bengio Y. Domain adaptation for large-scale sentiment classification: A deep learning approach. In: Proceedings of the twenty-eight international conference on machine learning, vol.27. 2011. p.97–110.
- Chen M, Xu ZE, Weinberger KQ, Sha F (2012) Marginalized denoising autoencoders for domain adaptation. ICML. arXiv preprintarXiv:1206.4683.

1. Обучить стэк автокодеров на основе исходных и целевых данных без меток классов. Это позволит обнаружить общие инвариантные скрытые признаки.
2. Обучить классификатор, используя преобразованные скрытые признаки, добавив метки исходных данных.

Реконструирование данных



Реконструирование данных

1. Обучаем автокодер (веса W_1, W_2 и параметры b_1, b_2) на целевых данных
2. Для каждого класса из исходных данных $\mathbf{x}_k^s$ на основе обученного автокодера реконструируем:

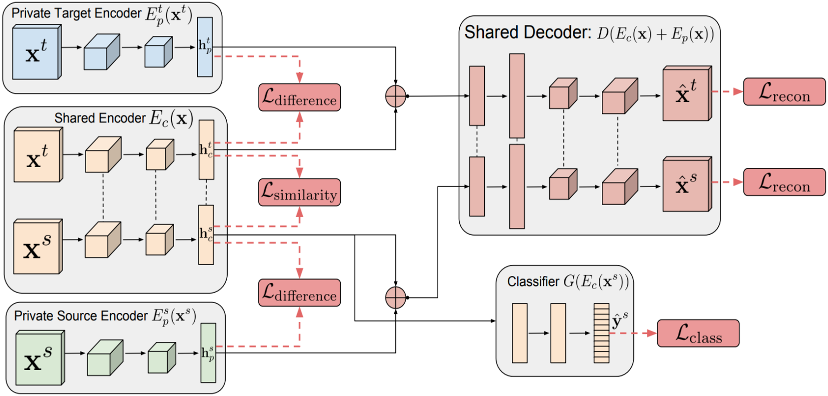
$$\mathbf{x}_k^{s \to t} = SA_{\mathrm{Recon}}(\mathbf{x}_k^s),$$

где

$$SA_{\mathrm{Recon}}(\mathbf{x}) = \sigma(W_2\sigma(W_1\mathbf{x} + b_1) + b_2)$$

- выход автокодера.

Domain Separation Networks (1)



Domain Separation Networks (2)

- K. Bousmalis et al. Domain Separation Networks // arXiv:1608.06019
- Кодер $E_c(x)$ с shared weights учится захватывать компоненты представления для данной входной выборки, которые являются общими для доменов.
- Частный кодировщик $E_p(x)$ (по одному для каждого домена) учится захватывать специфичные для домена компоненты представления.
- Общий декодер учится восстанавливать входную выборку, используя как частные, так и исходные представления.

Domain Separation Networks (3)

- Частные и общие компоненты представления раздвигаются с помощью мягких ограничений ортогональности подпространства $L_{difference}$, тогда как общие компоненты представления остаются подобными с потерей сходства $L_{similarity}$.

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \alpha\,\mathcal{L}_{\text{recon}} + \beta\,\mathcal{L}_{\text{difference}} + \gamma\,\mathcal{L}_{\text{similarity}}$$

- L_{task} обучает модель прогнозировать интересующие нас выходные метки:

$$\mathcal{L}_{\text{task}} = -\sum_{i=0}^{N_s} \mathbf{y}_i^s \cdot \log \hat{\mathbf{y}}_i^s,$$

$\mathbf{y}_i^s$ - one-hot кодирование меток классов для source, $\hat{\mathbf{y}}_i^s$ - softmax предсказание модели

Domain Separation Networks (4)

- L_{recon} - функция реконструкции

$$\mathcal{L}_{\text{recon}} = \sum_{i=1}^{N_s} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^s, \hat{\mathbf{x}}_i^s) + \sum_{i=1}^{N_t} \mathcal{L}_{\text{si\_mse}}(\mathbf{x}_i^t, \hat{\mathbf{x}}_i^t)$$

$$\mathcal{L}_{\text{si\_mse}}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{k}\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 - \frac{1}{k^2}([\mathbf{x} - \hat{\mathbf{x}}] \cdot \mathbf{1}_k)^2,$$

k - количество пикселей в x; $\mathbf{1}_k$ - единичный вектор длины k

Domain Separation Networks (5)

- $L_{difference}$ - функция потерь различий способствует ортогональности между общим и частным представлениями каждого домена:

$$L_{\text{difference}} = \left\| {\mathbf{H}_c^s}^\top \mathbf{H}_p^s \right\|_F^2 + \left\| {\mathbf{H}_c^t}^\top \mathbf{H}_p^t \right\|_F^2$$

- $L_{similarity}$:

$$\mathcal{L}_{\text{similarity}}^{\text{MMD}} = \frac{1}{(N^s)^2} \sum_{i,j=0}^{N^s} \kappa(\mathbf{h}_{ci}^s, \mathbf{h}_{cj}^s) - \frac{2}{N^s N^t} \sum_{i,j=0}^{N^s, N^t} \kappa(\mathbf{h}_{ci}^s, \mathbf{h}_{cj}^t) + \frac{1}{(N^t)^2} \sum_{i,j=0}^{N^t} \kappa(\mathbf{h}_{ci}^t, \mathbf{h}_{cj}^t)$$

Adversarial-Based методы

- Обучение нейронной сети с инвариантным по отношению к исходному и целевому доменам векторным представлением.
- Обученную сеть на размеченном source domain можно будет использовать на target domain, в идеале — практически без потери качества классификации.

Алгоритм DANN

- Y. Ganin et al. Domain-Adversarial Training of Neural Networks // arXiv:1505.07818
- Три части алгоритма:
    - Основная сеть для получения векторного представления (feature extractor);
    - "Голова отвечающая за классификацию на исходном домене;
    - "Голова которая обучается отличать данные из исходного домен от целевого.

Алгоритм ADDA (1)

- E. Tzeng et al. Adversarial Discriminative Domain Adaptation // arXiv:1702.05464
- Подразумевает разделение сети для исходного домена и сети для целевого домена
- Шаги алгоритма:
- Классифицирующая сеть обучается на исходном домене. Ећ векторное представление - M_s, а $\mathbf{X}_s$ - исходный домен.
- Инициализируем нейронную сеть для целевого домена с помощью обученной сети из предыдущего шага. Обозначим ећ M_t, а $\mathbf{X}_t$ - целевой домен.

Алгоритм ADDA (2)

- Перейдём к adversarial-тренировке: будем обучать дискриминатор D при фиксированных M_s и M_t с помощью следующей целевой функции:

$$\min_D L_{adv_D}(\mathbf{X}_s, \mathbf{X}_t, M_s, M_t) = -\mathbb{E}_{x_s \sim \mathbf{X}_s}[\log D(M_s(x_s))] \\ - \mathbb{E}_{x_t \sim \mathbf{X}_t}[\log(1 - D(M_t(x_t)))]$$

- Заморозим дискриминатор и дообучим M_t на целевом домене:

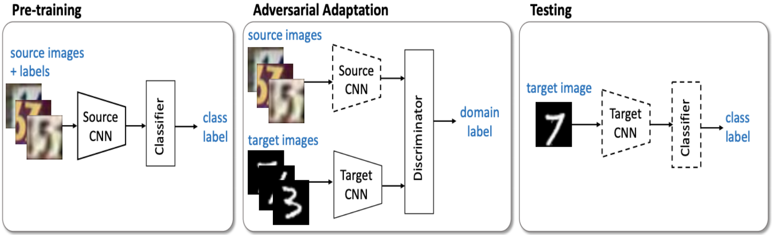
$$\min_{M_s, M_t} L_{adv_M}(\mathbf{X}_s, \mathbf{X}_t, D) = -\mathbb{E}_{x_t \sim \mathbf{X}_t}[\log D(M_t(x_t))]$$

- Шаги 3 и 4 повторяются несколько раз.

Алгоритм ADDA (3)

- Суть ADDA заключается в том, что мы сначала обучаем хороший классификатор на размеченном исходном домене, а затем с помощью adversarial-обучения адаптируем так, чтобы векторные представления классификатора на обоих доменах были близки.

Алгоритм ADDA (4)



Negative Transfer

- Случается, когда передача знаний из $\mathcal{D}_S$ и $\mathcal{T}_S$ приводит к снижению качества $\mathcal{T}_T$
- Если задачи $\mathcal{T}_S$ и $\mathcal{T}_T$ слишком различны, тогда передача "в лоб" может привести к снижению качества $\mathcal{T}_T$
- Важно проанализировать связанность $\mathcal{T}_S$ и $\mathcal{T}_T$ или $\mathcal{D}_S$ и $\mathcal{D}_T$, определить критерий схожести

Методы:

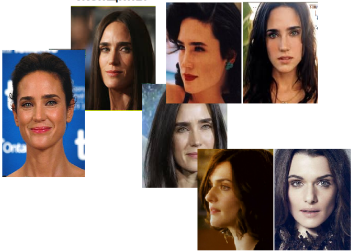
- Схожесть $\mathcal{T}_S$ и $\mathcal{T}_T$ определяется на основе схожести между распределениями вероятностей примеров
- Схожесть $\mathcal{T}_S$ и $\mathcal{T}_T$ определяется на основе введения характеристик задач более высокого уровня, например, признаков, которые известны заранее

Negative Transfer

- Схожесть T_S и T_T определяется на основе введения характеристик задач более высокого уровня, например, признаков, которые известны заранее
- Например, признак - пол

Мужчины

Женщины



Вопросы

?