Neural Network Theory and Applications：Lecture 6
神经网络理论及应用：第六讲

Bao-Liang Lu（吕宝粮）
Dept. of Computer Science & Engineering
Shanghai Jiao Tong University
Office: 3-431 Dianxin Bld, Minhang
Tel: 3420-5422
bllu@sjtu.edu.cn

课程后半段安排

- 4月11日：迁移学习、无监督学习
- 4月18日；25日：论文阅读
  - 每人精读至少一篇论文并作PPT；4月17日12：00之前提交PPT;
  - 挑选18位同学上台介绍；每人15分钟介绍，5分钟讨论；
  - 4月18日15：00之前通知入选者；PPT制作和上台介绍计入课堂表现分数
  - PPT不允许直接把论文原作者的PPT直接拿来用，要加上自己的见解和评论
- 4月28日 （上5月2日课）
  - 强化学习基础
- 5月9日
  - 18:00-19:00 深度强化学习（张伟楠老师）
  - 19:10-19:55 课堂小测验
  - 20:05-20：25 课程总结，大作业介绍

Office Hour

- Time: PM 3:30-4:30, Tuesday
- Venue: 3-414, Dianxin Building (电信学院3号楼414教室)
- Teaching Asistants:

刘伟

罗赟

赵黎明

Lecture Six

- Generative adversarial networks (GAN) and applications
- Transfer learning and applications

1、Generative Adversarial Network and Applications

History of Generative Adversarial Network

- Unsupervised adversarial networks were proposed by Jürgen Schmidhuber in 1990.
- The basic idea of generative adversarial networks was published in a 2010 blog post by Olli Niemitalo
- The name "GAN" was introduced by Ian Goodfellow et al. in 2014. Their paper popularized the concept and influenced subsequent work.

Jürgen Schmidhuber

Generative Model vs. Discriminative Model

- In machine learning, two main approaches are called the generative approach and the discriminative approach.
- Given an observable variable X and a target variable Y, a **generative model** is a statistical model of the joint probability distribution on $X \times Y$, $\boldsymbol{P(X,Y)}$**.** For example, RBM, GAN
- A **discriminative model** is a model of the conditional probability of the target Y, given an observation X, $\boldsymbol{P(Y|X)}$. For example, MLP, SVM

Generative Model

- A generative model is a model for generating all values for a phenomenon, both those that can be observed in the world and "target" variables that can only be computed from those observed.
- Types of generative models are:
  - Gaussian mixture model
  - Hidden Markov model
  - Probabilistic context-free grammar
  - Naive Bayes
  - Averaged one-dependence estimators
  - Latent Dirichlet allocation
  - **Restricted Boltzmann machine**
  - **Generative adversarial networks**

Generative Adversarial Network

- Generative adversarial networks (GANs) were introduced by Ian Goodfellow et al. in 2014. GANs are unsupervised deep generative networks, which are implemented by a system of two neural networks contesting with each other in a zero-sum game framework.
- GANs are used to generate realistic-like signals such as images, voice, text, EEG and so on.

GANs Model

- Normal GANs consist of two components: generator and discriminator.
- Generator G: creates (fake) samples that the discriminator cannot distinguish.
- Discriminator D: determine whether samples are fake or real.

Generator

- Generator:

  A differentiable function which is modeled as a neural network.

- Input:

  z: Random noise vector from some simple prior distribution.

- Output:

  x = G(z): Generated samples.

Generator

- Generator learns the underlying distribution of real data and the generated data G(z) confuse discriminator.
- The dimension of z should be at least as large as that of x.

Discriminator

- Discriminator is also modeled as a neural network.
- Input: Real sample and generated sample x.
- Output:
- –1 for real samples
- –0 for fake samples
- Discriminator tries to distinguish whether a sample comes from real distribution or generated distribution.

GANs Formulation

- The adversarial training is formulated as a minimax game, where:
  - The discriminator is trying to maximize its reward V(D, G)
  - The generator is trying to minimize Discriminator's reward

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

- Maximize discriminator:

$$\max_D V(D,G) = E_{x\sim p_{data}(x)}[log(D(x))] + E_{z\sim p_z(z)}[log(1 - D(G(z)))]$$

- Minimize generator

$$\min_G V(D,G) = E_{z\sim p_z(z)}[log(1 - D(G(z)))]$$

Zero-sum Game

- In game theory and economic theory, a zero-sum game is a mathematical representation of a situation in which each participant's gain or loss of utility is exactly balanced by the losses or gains of the utility of the other participants. If the total gains of the participants are added up and the total losses are subtracted, they will sum to zero.
- Thus, cutting a cake, where taking a larger piece reduces the amount of cake available for others, is a zero-sum game if all participants value each unit of cake equally.
- In psychology, zero-sum thinking refers to the perception that a situation is like a zero-sum game, where one person's gain is another's loss.

Training Algorithm

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Training Algorithm (1)

- The Algorithm takes *k* steps to optimize for the Discriminative net, D, and then one step of optimizing G, the generative net based on the outcome from D after above *k* steps.
- While optimizing for D, we work on the entire value function. For G, we only use the second term of the value function, as the gradient w.r.t. for the first term is zero.
- It is important to note that we ascend the gradient when optimizing for D, as it is a maximization problem, and descend the gradient when optimizing for G as it is a minimization problem.
- Initially when G is far from optimal, the gradients in second optimization might be very small. Instead, we can ascend the gradient for log(D(G(z))) initially.

Generative Adversarial Network

Advantages of GANs

- Plenty of existing work on Deep Generative Models
  - Boltzmann Machine
  - Deep Belief Nets
  - Variational AutoEncoders (VAE)
- Advantages of GANs
  - Sampling (or generation) is straightforward.
  - Training doesn't involve maximum likelihood estimation.
  - Robust to overfitting since generator never sees the training data.
  - Empirically, GANs are good at capturing the modes of the distribution.

Problems of GANs

- Non-convergence: It's difficult to converge for GANs.
- Finding equilibrium is a game of two players
- Exploiting convexity in function space, GAN training is theoretically guaranteed to converge if we can modify the density functions directly, but:
  - Instead, we modify G (sample generation function) and D (density ratio), not densities.
  - We represent G and D as highly non-convex parametric functions.
- 'Oscillation': can train for a very long time, generating very many different categories of samples, without clearly generating better samples.

Problems of GANs

- Mode collapse: most severe form of non-convergence.
- Generator fails to output diverse samples.
  - D in inner loop: convergence to correct distribution.
  - G in inner loop: place all mass on most likely point.

Target

Expected

Output

Applications of GANs

- GANs have been widely used to generate artificial data. Different models have been proposed to improve the stability of the training procedure and the quality of the generated data.
  - DCGAN
  - CGAN
  - InfoGAN
  - CycleGAN,
  - DiscoGAN
  - DualGAN
  - LSGAN
  - Co-GAN
  - Triple-GAN
  - …

Deep Convolutional GANs (DCGANs)

- The core to the DCGAN architecture uses a standard CNN architecture on the discriminative model. For the generator, convolutions are replaced with upconvolutions, so the representation at each layer of the generator is actually successively larger, as it maps from a low-dimensional latent vector onto a high-dimensional image.

Deep Convolutional GANs (DCGANs)

Generator:

Discriminator:

Deep Convolutional GANs (DCGANs)

- DCGANs tricks:
  - Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
  - Use batch normalization in both the generator and the discriminator.
  - Remove fully connected hidden layers for deeper architectures.
  - Use ReLU activation in generator for all layers except for the output, which uses Tanh.
  - Use LeakyReLU activation in the discriminator for all layers.

Deep Convolutional GANs (DCGANs)

Bedroom images

Image Super-resolution with GAN

4× SRGAN (proposed)

original

Image Super-resolution with GAN

Image Super-resolution with GAN

$$l^{SR} = \underbrace{l_{\mathrm{X}}^{SR}}_{\text{content loss}} + \underbrace{10^{-3} l_{Gen}^{SR}}_{\text{adversarial loss}}$$

Face Aging with Conditional GANs

- Differentiating Feature: Uses an Identity Preservation Optimization using an auxiliary network to get a better approximation of the latent code (z*) for an input image.
- Latent code is then conditioned on a discrete (one-hot) embedding of age categories.

Face Aging with Conditional GANs

EEG Data Augmentation for Emotion Recognition Using a Conditional Wasserstein GAN

Yun Luo and Bao-Liang Lu, EEG Data Augmentation for Emotion Recognition Using a Conditional Wasserstein GAN, Proc. of IEEE EMBC'2018

EEG-based Emotion Recognition

- Comparing with images and speech signals, it is hard to collect large-scale EEG data.
  - The high price of EEG acquisition devices.
  - It is difficult to acquire sufficient labeled data from subjects in EEG-based emotion recognition experiments.
- Data augmentation is one of promising ways to solve the data scarcity problem. This approach generates the realistic-like data by applying a transformation to the real data .

rotation

translation

Data Augmentation

- It is common to generate artificial images through geometric transformation (translation, rotation, scaling, horizontal shearing) in the field of computer vision.
- Similar with CV, geometric transformation methods have been applied to generate EEG

Enlarge dataset by translation

Data Augmentation

- These data augmentation methods only lead to a signal-level transformation through distortions, which is not helpful for dividing a clear boundary between data manifolds.
- Recently, GANs have revealed their potential in generating realistic-like data such as images by adopting an adversarial training.
- In the filed of computer vision, researchers have adopted GANs-based methods for data augmentation and made significant progresses.

Data Augmentation and GANs

- Unlike geometric transformation, GANs capture the latent distribution of the data by unsupervised learning.
- Data generated by GANs enlarge the data manifold, which leads to a better margin for the classifier.

Pictures generated by GANs

EEG data augmentation

- A Wasserstein GAN with gradient penalty is adopted to generate realistic-like EEG data in differential entropy (DE) form.
- Three indicators are used to judge the qualities of the generated high-dimensional EEG data, which supplement the data manifold and lead to better classification of different emotions.

EEG data augmentation

- CWGAN consists of two parts: the generator generates realistic-like data and the discriminator distinguishes real and generated distribution. An auxiliary label information is concatenated to control the categories of the generated data.

EEG data augmentation

- Two-dimensional visualizations of the real and generated DE data of one subject in SEED dataset. Data points with red, green and blue colors represent three emotions of negative, neutral and positive, respectively. The lines represent the real data and the thin points represent the generated data.

Experimental Results

- We evaluate the proposed CWGAN framework on two public EEG datasets for emotion recognition, namely SEED and DEAP. The experimental results demonstrate that using the EEG data generated by CWGAN significantly improves the accuracies of emotion recognition models.

Data Appended	SEED		DEAP-Arousal		DEAP-Valence	
	Mean	Std.	Mean	Std.	Mean	Std.
0×Dataset	0.8399	**0.0972**	0.6902	0.1361	0.5376	0.1308
1×Dataset	**0.8696**	0.1272	**0.7817**	**0.0958**	**0.7389**	**0.1082**
2×Dataset	0.8659	0.1283	0.7437	0.1227	0.6543	0.1506
3×Dataset	0.8638	0.1276	0.7479	0.1133	0.6625	0.1477
4×Dataset	0.8623	0.1275	0.7450	0.1013	0.6496	0.1504
5×Dataset	0.8600	0.1294	0.7434	0.1151	0.6467	0.1372

Experimental Results

The confusion matrixes (SEED dataset) trained by (a) original training set and (b) appended training set (1 time).

Visualization-Positive

(a) Real Data (b) Generated Data

Visualization-Neutral

(a) Real Data (b) Generated Data

Visualization-Negative

(a) Real Data

(b) Generated Data

References

1. Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets, International Conference on Neural Information Processing Systems. MIT Press, 2014:2672-2680.
2. Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. Computer Science, 2015.
3. Mirza M, Osindero S. Conditional Generative Adversarial Nets. Computer Science, 2014:2672-2680.
4. Ledig C, Wang Z, Shi W, et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. 2016:105-114.
5. Zhang H, Xu T, Li H, et al. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. 2016:5908-5916.
6. Antipov G, Baccouche M, Dugelay J L. Face Aging With Conditional Generative Adversarial Networks. 2017.
7. Liu M Y, Tuzel O. Coupled Generative Adversarial Networks. 2016.

2、 Transfer Learning and Applications

Transfer Learning

- Domain Adaption
- Domain Generalization
- Concept Drift
- One-Shot Learning
- Zero-Shot Learning

Transfer of Learning

- A psychological point of view
- The study of dependency of human conduct, learning or performance on prior experience.
- Thorndike and Woodworth explored how individuals would transfer in one context to another context that share similar characteristics.
- C++ → Java
- Maths/Physics → Computer Science/Economics

Thorndike and Woodworth, The Influence of Improvement in one mental function upon the efficiency of the other functions, 1901

Transfer Learning

- In the machine learning community
- The ability of a system to recognize and apply knowledge and skills learned in previous tasks to novel tasks or new domains, which share some commonality.

- Given a target task, how to identify the commonality between the task and previous (source) tasks, and transfer knowledge from the previous tasks to the target one?

Traditional ML vs. TL

Traditional ML in multiple domains

Humans can learn in many domains.

Transfer of learning across domains

Humans can also transfer from one domain to other domains.

Traditional ML vs. TL

- **Traditional ML Assumptions**
  - **Training and testing samples must be in the same feature distributions.**
  - **Training samples must be enough.**
- **Transfer Learning Conditions**
  - **Source and target domains do not need to be in the same distributions.**
  - **Less training samples, even none.**

(a) Traditional Machine Learning (b) Transfer Learning

Notations

Domain:

It consists of two components: A feature space $\mathcal{X}$, a marginal distribution

$\mathcal{P}(X)$, where $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$

In general, if two domains are different, then they may have different feature spaces or different marginal distributions.

Task:

Given a specific domain and label space $\mathcal{Y}$, for each x_i in the domain, to predict its corresponding label y_i, where $y_i \in \mathcal{Y}$

In general, if two tasks are different, then they may have different label spaces or different conditional distributions

$\mathcal{P}(Y|X)$, where $Y = \{y_1, ..., y_n\}$ and $y_i \in \mathcal{Y}$

Notations

For simplicity, we only consider at most two domains and two tasks.

Source domain:

$$\mathcal{P}(X_S), \text{ where } X_S = \{x_{S_1}, x_{S_2}, ..., x_{S_{n_S}}\} \in \mathcal{X}_S$$

Task in the source domain:

$$\mathcal{P}(Y_S|X_S), \text{ where } Y_S = \{y_{S_1}, y_{S_2}, ..., y_{S_{n_S}}\} \text{ and } y_{S_i} \in \mathcal{Y}_S$$

Target domain:

$$\mathcal{P}(X_T), \text{ where } X_T = \{x_{T_1}, x_{T_2}, ..., x_{T_{n_T}}\} \in \mathcal{X}_T$$

Task in the target domain

$$\mathcal{P}(Y_T|X_T), \text{ where } Y_T = \{y_{T_1}, y_{T_2}, ..., y_{T_{n_T}}\} \text{ and } y_{T_i} \in \mathcal{Y}_T$$

Why Transfer Learning?

- In some domains, labeled data are in short supply.
- In some domains, the calibration effort is very expensive.
- In some domains, the learning process is time consuming.
- How to extract knowledge learnt from related domains to help learning in a target domain with a few labeled data?
- How to extract knowledge learnt from related domains to speed up learning in a target domain?

Transfer learning framework

Transfer Component Analysis

Let the source and target domain D_S and D_T be in the same space χ, but X_S and X_T have different marginal distributions $P(X_S) \neq Q(X_T)$ (or $P \neq Q$ for short).
The key assumption: $P(Y_S|X_S) = P(Y_T|X_T)$.

Pan, Sinno Jialin, et al. Domain adaptation via transfer component analysis IEEE Transactions on Neural Networks 22.2 (2011): 199-210.

Transfer Component Analysis

To find a common latent representation for both X_S and X_T, a nonlinear transformation $\phi: \chi \to \mathcal{H}$ ($\mathcal{H}$ is an RKHS) is introduced.

We desire that $P'(X'_S) \approx Q'(X'_T)$, which is complemented by minimizing MMD (*Maximum Mean Discrepancy*):

$$\mathrm{Dist}(X'_S, X'_T) = \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(x_{S_i}) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi(x_{T_i}) \right\|_{\mathcal{H}}^2$$

Transfer Component Analysis

By kernel trick: $k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$, the distance can be written as:

$$Dist(X_S', X_T') = tr(KL),$$

Where

$$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix}$$

K is a $(n_1 + n_2) \times (n_1 + n_2)$ kernel matrix, the subscripts represents where the x_i, x_j come from in $k(x_i, x_j)$.

Matrix K can be decomposed as $\mathrm{K} = (KK^{-1/2})(K^{-1/2}K)$. Use a $(n_1 + n_2) \times m$ matrix $\widetilde{W}$ to transform the feature vectors in K's decomposed components

$$\widetilde{K} = (KK^{-1/2}\widetilde{W})(\widetilde{W}^T K^{-1/2} K) = KWW^T K$$

Where $W = K^{-1/2}\widetilde{W}$.

Transfer Component Analysis

Using the definition of $\widetilde{K}$, we have

$$Dist(X_S', X_S') = tr\big((KWW^TK)L\big) = tr(KWLW^TK)$$

The kernel learning problem is then reduced to

$$\min_{W} \ \mu\, tr(KWLW^TK) + tr(W^TW)$$

$$s.t. \quad W^TKHKW = I$$

Where $tr(W^TW)$ is a regularization term . Restriction term is added to avoid the trivial solution ($W = 0$), where H is the centering matrix.

The solution to W is then the eigenvectors corresponding to the ***m* leading eigenvalues** of $(I + \mu KLK)^{-1}$KHK.

Transductive Parameter Transfer

Enver Sangineto, Gloria Zen, Elisa Ricci, and Nicu Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In ACM International Conference on Multimedia, pages 357–366. ACM, 2014.

Transductive Parameter Transfer

An introductory example

Suppose we have labeled data sets from three domains (source domains) and an unlabeled target domain data set.

E. Sangineto, G. Zen, E. Ricci, and N. Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In ACM Multimedia, 2014.

Transductive Parameter Transfer

An introductory example

Suppose we have labeled data sets from three domains (source domains) and an unlabeled target domain data set.

For the source domains, we can easily train SVMs on the labeled data.

E. Sangineto, G. Zen, E. Ricci, and N. Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In ACM Multimedia, 2014.

Transductive Parameter Transfer

An introductory example

Suppose we have labeled data sets from three domains (source domains) and an unlabeled target domain data set.

For the source domains, we can easily train SVMs on the labeled data.

Now the problem is how can we transfer the parameters in the trained SVMs to the target domain data set.

E. Sangineto, G. Zen, E. Ricci, and N. Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In ACM Multimedia, 2014.

Transductive Parameter Transfer

Learn Individual Classifiers

Train Regression Model (Multi-output support vector regression)

Obtain Target Classifier

Wei-Long Zheng and Bao-Liang Lu*, Personalizing EEG-based Affective Models with Transfer Learning, Proc. of the 25th International Joint Conference on Artificial Intelligence (IJCAI-16), New York

Transductive Parameter Transfer

Demonstration of the objective

We denote the source domain data sets as

$$D_1^s, \ldots, D_N^s, \; D_i^s = \{\mathbf{x}_j^s, y_j^s\}_{j=1}^{n_i^s}$$

where D_i^s indicates the *i*th data set. Also let us use X_i^s to indicate the data without label. Then we assume that X_i^s is drawn from the marginal distribution $P_{X_i^s}$: $X_i^s \sim P_{X_i^s}$.

For each data set, we can easily train an SVM parameterized with $\boldsymbol{\theta}_i^s$.

The question is: for target domain data $X^t \sim P_{X^t}$, how can we obtain the corresponding SVM parameter $\boldsymbol{\theta}^t$.

Transductive Parameter Transfer

Analysis

By observing the problem setting, we assume there is a function from marginal distribution to the SVM parameter vector:

$$f : \mathcal{P} \rightarrow \Theta$$

where $\mathcal{P}$ and Θ indicate the space of the marginal distribution and the SVM parameter vectors respectively.

So we have

$$\boldsymbol{\theta}_i^s = f(P_{X_i^s})$$

If the function can be obtained, the target domain SVM parameter vector is simply

$$\boldsymbol{\theta}^t = f(P_{X^s}).$$

Transductive Parameter Transfer

Analysis (Cont.)

However, the marginal distributions $P_{X_i^s}$ can not be easily depicted.

Because we know $X_i^s \sim P_{X_i^s}$, we can use X_i^s instead of $P_{X_i^s}$.

We then assume another function:

$$\hat{f} : 2^{\mathcal{X}} \to \Theta.$$

$\hat{f}$ is a function from the feature set space $2^{\mathcal{X}}$ to the parameter space.

Similarly, we have $\boldsymbol{\theta}_i^s = \hat{f}(X_i^s)$ and $\boldsymbol{\theta}^t = \hat{f}(X^t)$.

Now the problem becomes how to learn the function $\hat{f}$.

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR

The training set is $\mathcal{T} = \{X_i^s, \boldsymbol{\theta}_i^s\}_{i=1}^N$.

It is a simple multivariate regression problem. Various methods can be used to solve it. In the original paper, M-SVR (Tuia et al.) was applied.

D. Tuia, J. Verrelst, L. Alonso, F. Perez-Cruz, andG. Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameterestimation. IEEE Geoscience and Remote SensingLetters, 8(4):804–808, July 2011.

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR Cont.

In the M-SVR framework, $\hat{f}$ can be defined by

$$\hat{f}(X) = \phi(X)'B + \boldsymbol{c}'$$

where B is a matrix and $\boldsymbol{c}$ is a vector. $\phi(X)$ is a nonlinear mapping from the matrix space to a vector space.

The corresponding loss is defined as

$$L(B, \boldsymbol{c}; \mathcal{T}) = \frac{1}{2}||B||_F^2 + \lambda_E \sum_{i=1}^{N} E(||\boldsymbol{\theta}_i^{s\prime} - \hat{f}(X_i^s)||)$$

where $||\cdot||_F$ indicates the Frobenius norm: $||A||_F = \sqrt{\text{trace}(A'A)}$, the λ_E is a tradeoff hyperparameter, and $E(\cdot)$ is a loss function parameterized by ϵ :

$$E(u) = \begin{cases} 0 & u < \epsilon \\ u^2 - 2u\epsilon + \epsilon^2 & u \geq \epsilon \end{cases}$$

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR Cont.

By applying the kernel trick, the prediction function can be rewritten as

$$\hat{f}(X) = [\kappa(X_1^s, X), \ldots, \kappa(X_N^s, X)] \cdot V + \boldsymbol{c}'$$

where V is obtained by solving the dual problem.

To optimize V and $\boldsymbol{c}$, we adopt iterative procedure. In iteration k, suppose we have the temporal values of B and $\boldsymbol{c}$ being B^k and $\boldsymbol{c}^k$. Let us denote the error $\boldsymbol{e} = \hat{f}(X)' - \boldsymbol{\theta}$, then the loss function can be expand with Taylor expansion:

$$L^*(B, \boldsymbol{c}) = \frac{1}{2}||B||_F^2 + \lambda_E(\sum_{i=1}^{N} E(e_i^k) + \left.\frac{dE(u)}{du}\right|_{u_i^k} \frac{(e_i^k)'}{u_i^k}[e_i - e_i^k])$$

where $u = ||e||$. The superscripts ‘k’s indicate the values at the kth iteration.

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR Cont.

Then we can further approximate the loss function by using a quadratic approximation:

$$L^{**}(B, \boldsymbol{c}) = \frac{1}{2}||B||_F^2 + \lambda_E \frac{1}{2} \sum_{i=1}^{N} a_i u_i^2 + \tau$$

where

$$a_i = \frac{\lambda_E}{u_i^k} \frac{dE(u)}{du}\bigg|_{u_i^k}$$

and τ is a value that does not depend on B and $\boldsymbol{c}$.

It can be easily observed that the $L^{**}(\cdot)$ is simply a quadratic function of B and c, so the solution B^{sol} and $\boldsymbol{c}^{sol}$ can be obtained.

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR Cont.
The update rule of B and $\boldsymbol{c}$ is

$$\begin{bmatrix} B^{k+1} \\ (\boldsymbol{c}^{k+1})' \end{bmatrix} = \begin{bmatrix} B^{k} \\ (\boldsymbol{c}^{k})' \end{bmatrix} + \eta_k \begin{bmatrix} B^{sol} - B^{k} \\ (\boldsymbol{c}^{sol})' - (\boldsymbol{c}^{k})' \end{bmatrix}$$

where the step size η_k is decided by back tracking algorithm. Specifically, we initially set it to 1, and check whether the loss function drops. If not, the step size is multiplied by a value less than 1 and retry.

Transductive Parameter Transfer

Learning problem of $\hat{f}$ with M-SVR Cont.

Solving B is not convenient because map function $\phi(X)$ is usually not analytical, so we substitute V into the previous equations by utilizing $B_{\cdot,j} = KV_{\cdot,j}$ where K is the kernel matrix defined by

$$K_{ij} = \kappa(X_i, X_j)$$

Transductive Parameter Transfer

Some points

1. Other types of paramterized models (e.g., LDA) should also work well with TPT.
2. When learning $\hat{f}$, the features are matrices. So some kind of mapping should be applied to project the features to vector spaces. For M-SVR, the mapping corresponds to specially designed kernels.
3. The regression method used in TPT is also free to choose. But the authors found M-SVR performed better on their tasks.

Domain Adversarial Neural Network

DANN is a domain adaptation approach with deep architectures that can be trained with labeled source domain data and unlabeled target domain data. Its adaptation behavior is achieved by augmenting a normal feed-forward model with an adversarial manner.

Ganin and Lempitsky, Unsupervised domain adaptation by backpropagation, Proc. Of International Conference on Machine Learning, 2015, pp. 1180–1189.

Domain Adversarial Neural Network

Feature Extractor

The first layers works as a feature extractor, it maps the low level features to a non-linear transformed high level feature f

Domain Adversarial Neural Network

The upper layers take as input the feature f and outputs the label prediction for the required task

These two parts works like a normal multilayer network.

Domain Adversarial Neural Network

The lower layers take as input the feature f and outputs the domain label. It tries to distinguish whether the data come from X_T or X_S.

Domain Adversarial Neural Network

Each classifier back propagates the gradient to update the parameters

Domain Adversarial Neural Network

The gradient of label classifier passes to the feature extractor normally

Domain Adversarial Neural Network

The gradient of domain classifier passes to the feature extractor with a reversal parameter $-\lambda$

Domain Adversarial Neural Network

The reversal parameter $-\lambda$ makes the domain classifier and feature extractor updates their parameters in an adversarial manner.

The feature extractor wants to cheat the domain classifier and tries to hide any domain dependent information by removing it from feature f

Domain Adversarial Neural Network

When the model converges, the feature f would be containing no domain related information. Thus it works as function of domain adaptation.

Adversarial Discriminative Domain Adaptation

- ADDA is based on deep adversarial network. It aligns the distributions of source domain data and target domain data by training a feature extractor and a domain classifier in an adversarial manner.

E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in IEEE Conference on Computer Vision and Pattern Recognition, July 2017, pp. 2962–2971.

Training of ADDA (step 1)

- First, we train a simple multilayer perceptron on the source domain data with source domain data as input. The network is divided into 3 parts: input layer, feature extractor and label predictor. We can follow the general training procedure for this step (e.g., BP algorithm).

Training of ADDA (step 2)

- After the network converge, another feature extractor is constructed. The new feature extractor only accepts input from the target domain. Its parameters are initialized by copying the ones from the original feature extractor.

Training of ADDA (step 3)

- In this step, a domain classifier sub-network is trained when the label predictor and the source domain feature extractor are not updated. The domain classifier is trained to discriminate which domain (source or target?) the input is from.

Training of ADDA (step 3 Cont.)

- However, the target domain feature extractor is trained to deceive the domain classifier. The domain classifier and the target domain feature extractor compete with each other (i.e., adversarial training).

Source domain training case:

Training of ADDA (step 3 Cont.)

- However, the target domain feature extractor is trained to deceive the domain classifier. The domain classifier and the target domain feature extractor compete with each other (i.e., adversarial training).

Target domain training case:

Training of ADDA (step 3 Cont.)

- As the training procedure progresses, the target feature extractor becomes better and better in deceiving the domain classifier. In another word, the target feature extractor extracts features that are similar to the source ones. In this way, the domain discrepancy is reduced.

Training of ADDA (step 4)

- After the adversarial training converges, we can predict the labels of the target domain data by using the pass as shown in the figure.

Negative Transfer

- Most approaches to transfer learning assume transferring knowledge across domains be always positive.
- However, in some cases, when two tasks are too dissimilar, brute-force transfer may even hurt the performance of the target task, which is called **negative transfer**.
- Some researchers have studied how to measure relatedness among tasks.
- How to design a mechanism to avoid negative transfer needs to be studied theoretically.

Personalizing EEG-based Affective Models with Transfer Learning

Wei-Long Zheng and Bao-Liang Lu, Personalizing EEG-based Affective Models with Transfer Learning, Proc. of the 25th International Joint Conference on Artificial Intelligence (IJCAI-16).

Affective Brain-Computer Interface (aBCI)

- A brain-computer interface (BCI) is a direct communication pathway between a brain and an external device.
- Affective Brain-Computer Interface (aBCI) aims to enhance brain-computer interface systems with the ability to detect, process, and respond to user's emotion.

Motivation of this work

- Most of the existing studies focus on training subject-specific affective models regarding individual differences.
  - In calibration phase, to acquire a large number of labeled data is time-consuming and annoying.
- Personalizing affective models for target subjects with knowledge transfer from labeled data of other subjects.

Challeges of Subject Transfer

The feature distributions of EEG data of source subjects and target subjects are not independently and identically distributed (i.i.d.) due to:

- The structural and functional variability between subjects
- The non-stationary nature of EEG signals
- The inherent changes of environmental variables

Challeges of Subject Transfer: An Example

Subject 1: □
Subject 2: ○

Feature Reduction based Subject Transfer

Although the distributions of source domain and target domain in high dimensional space are different, we may find a low dimensional manifold space where the distributions of both domains are similar.

TCA and KPCA try to learn a set of common transfer components underlying both the source domain and the target domain. When projected to this subspace, the difference of feature distributions of
both domains can be reduced.

$$P(\phi(X_S)) \approx P(\phi(X_T))$$
$$P(Y_S|\phi(X_S)) \approx P(Y_T|\phi(X_T))$$

TCA-based Subject Transfer

TCA algorithm is firstly proposed by Pan et al. for cross-domain indoor WiFi localization and cross-domain text classification.

Algorithm 1 TCA-based Subject Transfer

input : Source domain data set $\mathcal{D}_S = \{(x_{S_i}, y_{srci})\}_{i=1}^{n_1}$, and target domain data set $\mathcal{D}_T = \{x_{T_j}\}_{j=1}^{n_2}$.
output : Transformation matrix W.

1: Compute kernel matrix K from $\{x_{S_i}\}_{i=1}^{n1}$ and $\{x_{T_j}\}_{j=1}^{n2}$, matrix L, and the centering matrix H.
2: Eigendecompose the matrix $(KLK + \mu I)^{-1}KLK$ and select the m leading eigenvectors to construct the transformation matrix W.
3: **return** tranformation matrix W.

Pan S J, Tsang I W, Kwok J T, et al. Domain adaptation via transfer component analysis [J]. IEEE Transactions on Neural Networks, 2011, 22(2): 199-210.

Transductive Parameter Transfer

Enver Sangineto, Gloria Zen, Elisa Ricci, and Nicu Sebe. We are not all equal: Personalizing models for facial expression analysis with transductive parameter transfer. In ACM International Conference on Multimedia, pages 357–366. ACM, 2014.

SJTU Emotion EEG Dataset (SEED)

- No. subjects: 15, No. EEG Channels: 62
- EEG Features: differential entropy in five frequency bands.
- Three emotions: positive, neutral and negative

http://bcmi.sjtu.edu.cn/~seed/

Wei-Long Zheng, and Bao-Liang Lu, "Investigating Critical Frequency Bands and Channels for EEG-based Emotion Recognition with Deep Neural Networks",, IEEE Transactions on Autonomous Mental Development (IEEE TAMD), 2015.

Evaluation Details

- Leave-one-subject-out cross validation method
- Baseline method
  - We concatenate data from all available subjects as training data and train a generic classifier with linear SVM.
- Transductive SVM (T-SVM)
  - It is developed to learn a decision boundary and maximize the margin with unlabeled data.
- One vs. one strategy for multi-class classification
- All the algorithms are implemented in MATLAB.

Experimental Results

Comparison of KPCA and TCA approaches for different dimensionality of the subspace.

Experimental Results

Stats.	Generic	KPCA	TCA	T-SVM	TPT
Mean	0.5673	0.6128	0.6364	0.7253	**0.7631**
Std.	0.1629	0.1462	0.1488	**0.1400**	0.1589

Generalization Performance

Multimodal Vigilance Estimation with Adversarial Domain Adaptation Networks

He Li, Wei-Long Zheng and Bao-Liang Lu, Multimodal Vigilance Estimation with Adversarial Domain Adaptation Networks, to appear in Proc. IJCNN'18,

Vigilance Estimation during Driving

- Vigilance estimation during driving is a very important and interesting topic, because low vigilance level during driving directly relates to car crashes.
- We adopt EEG and EOG to build a multimodal vigilance estimation system. Both modalities are considered tightly related to the vigilance level.

Difficulty in Data Acquisition

- Conventionally, to train a vigilance estimation system for subject A, we have to obtain his EEG&EOG signals during the driving procedure, as well as the corresponding labels (here, the vigilance level indices).
- However, it is usually expensive or effort demanding to obtain such labeled data set:
  - Some kinds of vigilance level indices require the subject perform a parallel task while driving, which is disturbing.
  - Some kinds of vigilance level indices are obtained by watching the video of the subject and labeling the vigilance level by hand.
  - Some kinds of vigilance level indices demand for extra device, e.g., eye tracking glasses, which is both expensive for the facilitators and inconvenient for the subjects.

Cross-Subject Vigilance Estimation

- To solve the problem, in this study, we utilize the existing labeled data from other subjects (source domain) to the training of the system for the target subject (target domain). In this way, label information is not necessary for the target subject and the existing labeled data are reused.

Inter-Subject Discrepancy

- However, directly using the existing data for training is not a good idea. The data distributions vary a lot for different subjects due to the inter-subject discrepancy. As the result, the model trained on the blue source domain data can perform badly on the red target domain data.

Eliminating the Discrepancy with TL

- In order to eliminate the inter-subject discrepancy, we adopt transfer learning methods: DANN and ADDA.
- In our study, the source domain is the existing labeled data, and the target domain is the unlabeled target subject data.
- Because our task is a regression problem, we use single neuron with identity activation function as the output layer for the neural networks.
- ADDA does not support transfer learning with multiple source domains, so we combine all the source domains to form a single source domain.

The SEED-VIG Dataset

- The SEED-VID is a publicly available dataset contributed by the BCMI Lab. It contains 23-subject EEG and EOG signals recorded during driving procedures. The vigilance levels are also labeled in terms of PERCLOS indices.

Wei-Long Zheng and Bao-Liang Lu, A multimodal approach to estimating vigilance using EEG and forehead EOG. Journal of Neural Engineering, 14(2): 026017, 2017.

Training of the Models

- We applied leave-one-subject-out cross validation for better estimation of the performance. For each of the target subjects, we train the models with labeled data from the other 22 subjects and test on the target one.
- We also trained a linear kernel SVR without any transfer learning method applied. This result is used as baseline for performance comparison.
- In fact, the EEG and EOG signals must be preprocessed before they are used. The detailed procedure can be found in paper by Zheng et al.

Wei-Long Zheng and Bao-Liang Lu, A multimodal approach to estimating vigilance using EEG and forehead EOG. Journal of Neural Engineering, 14(2): 026017, 2017.

Experimental Results

- Because our problem is a regression problem, we adopt Pearson correlation coefficients (PCC) and root-mean-square errors (RMSE) for the evaluation of the results.

$$\text{PCC} = \frac{\sum_{i=1}^{n_{\text{target}}} (\hat{y}_i - \overline{\hat{y}}_i)(y_i - \overline{y})}{\sigma_{\hat{y}} \sigma_y}$$

$$\text{RMSE} = \sqrt{\frac{1}{n_{\text{target}}} \sum_{i=1}^{n_{\text{target}}} (\hat{y}_i - y_i)^2}$$

		Baseline	DANN	ADDA
PCC	AVG	0.7606	0.8402	**0.8442**
	STD	0.2314	0.1535	**0.1336**
RMSE	AVG	0.1689	0.1427	**0.1405**
	STD	0.0673	0.0588	**0.0514**

Experimental Results

- Here we show the estimations made by the baseline, the DANN and the ADDA methods for some of the subjects.

Data Distributions

- To show how the methods influence the data distribution, we reduce the dimension of the data with PCA and plot the distributions as follows.

Sleep Quality Estimation with Adversarial Domain Adaptation: From Laboratory to Real Scenario

Jia-Jun Tong, Yun Luo, Bo-Qun Ma, Wei-Long Zheng, Bao-Liang Lu, Xiao-Qi Song and Shi-Wei Ma, Sleep Quality Estimation with Adversarial Domain Adaptation: From Laboratory to Real Scenario, to appear in Proc. IJCNN'18,

Instance of DANN: Sleep Quality

- Sleep Quality Estimation with Adversarial Domain Adaptation: From Laboratory to Real Scenario
- Problem Formulation
  - Estimate last night sleep quality with EEG
  - Transfer from lab to real scenario

Transfer

Instance of DANN: Sleep Quality

- Laboratory Experiment Settings
  - 10 students (6M, 4F), age range: 21~26
  - Last night Sleep: 4-h, 6-h, 8-h
  - 62 channel wet electrodes
  - 30 min/experiment

4h
6h
8h

Instance of DANN: Sleep Quality

- Laboratory Experiment Settings
  - 70 drivers (70M), age range: 25–49
  - Last night Sleep: 4~10 hour
  - 18 channel Dry electrodes
  - 6 min/experiment

Sleep：
4~10h

Instance of DANN: Sleep Quality

Laboratory

Real Scenario

Instance of DANN: Sleep Quality

- Difference between laboratory and real scenario (Detailed)

Categories	Specifications	Laboratory	Real-scenario
Subjects	Age	21-26	25-49
	Sex-distribution	4 males, 6females	70 males
	Occupation	Students	High-speed train drivers
	Experiments per subject	3	1
Controlled conditions	Sleep time	4,6,8 hours	Not controlled
	Monitoring sleep	Yes	No
	Head Cleaning	Yes	No
	Experiment Starts	$\leq$1 hour after wake up	1-10 hour after wake up
Experiment settings	Experiment duration	30 min	6 min
	Task	Eyes open	Eyes open and close
	Environment	Quite and isolated	Noisy
Devices	Electrods type	Wet electrodes	Dry electrodes
	Resolution	62 channels	18 channels
	Reference	REF(Between CPZ and PZ)	A1 and A2 (On the ears)
	Common Mode Follower	0	1 (CMF)
	Sample rate	1000 Hz	300 Hz

Instance of DANN: Sleep Quality

- Preprocessing to make the data format unified
- Electrodes Selection
  - We selected the corresponding electrodes from the laboratory device
- Resetting Reference
  - laboratory data use a specific REF electrode
  - Real-scenario data use the mean of A1 and A2 electrodes
  - Reset the reference to the mean signal of all 18 electrodes

Instance of DANN: Sleep Quality

- Feature Extraction
  - DE features for each electrode with 5 frequency bands
- Baseline Methods
  - TCA, ITL, GFK, JDA, SA, TJM, MIDA
- Domain Adversarial Neural Network

Instance of DANN: Sleep Quality

- Tasks
  - Cross-Subject Task: 5 fold CV on real-scenario data
  - Cross-Scenario Task: Train on Lab data, test on real-scenario data

ACCURACY (%) OF CROSS-SCENARIO TASK

Method	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean±Std
SVM	43.86	40.35	42.18	38.98	43.59	41.79±2.10
TCA	54.60	61.19	38.89	67.34	56.87	55.60±10.6
ITL	53.69	63.10	24.68	60.71	49.41	50.32±13.7
GFK	49.64	62.46	40.99	**70.95**	49.25	54.66±11.9
JDA	47.70	61.43	35.71	66.11	66.94	55.58±13.5
SA	42.74	62.54	37.50	61.23	54.17	51.64±11.1
TJM	41.67	60.28	38.97	59.88	60.60	52.28±10.9
MIDA	56.47	**66.35**	38.69	68.93	56.87	57.46±11.9
DANN	**69.10**	64.68	**60.79**	62.6296	**69.25**	**65.29**±3.80

Instance of DANN: Sleep Quality

- The effects of adaptation on cross-scenario task

(a) None-adapted (b) TCA (c) ITL (d) GFK (e) JDA

(f) SA (g) TJM (h) MIDA (i) DANN

Instance of DANN: Sleep Quality

- The confusion matrix of DA models on cross-scenario task