Multi-task Multi-kernel regression

Yucong Lin, Jiaheng Yin, Junwei Lu

March 2021

1 Introduction

In this paper, we introduce a Multi-task Multi-kernel regression model on medical relationship extraction data.

2 Kernel Regression

2.1 Representer Theorem

For any L2-regularized linear model

$$min_w \frac{\lambda}{N} w^T w + \frac{1}{N} \sum_{n=1}^{N} err(y_n, w^T z_n) \tag{1}$$

optimal

$$w^* = \sum_{n=1}^{N} \beta_n z_n \tag{2}$$

Proof by contradiction.

2.2 Kernel Logistic Regression

According to representer theorem, L2-regularized logistic regression

$$min_w \frac{\lambda}{N} w^T w + \frac{1}{N} \sum_{n=1}^{N} \log(1 + exp(-y_n w^T z_n)) \tag{3}$$

has optimal solution

$$w^* = \sum_{n=1}^{N} \beta_n z_n \tag{4}$$

The problem can be converted to the optimization of β

$$min_\beta \frac{\lambda}{N} \sum_{m=1}^{N} \beta_m z_m^T \sum_{n=1}^{N} \beta_n z_n + \frac{1}{N} \sum_{n=1}^{N} \log(1 + exp(-y_n \sum_{m=1}^{N} \beta_m z_m^T z_n)) \tag{5}$$

We use z_i to represent the spatial transformation

$$x_i \to z_i = \Phi(x_i) \tag{6}$$

and kernel function is

$$K(x_i, x_j) = \Phi^T(x_i)\Phi(x_j) \tag{7}$$

The optimization problem can be represented as

$$min_\beta \frac{\lambda}{N} \sum_{m=1}^{N} \sum_{n=1}^{N} \beta_m \beta_n K(x_m, x_n) + \frac{1}{N} \sum_{n=1}^{N} \log\left(1 + exp(-y_n \sum_{m=1}^{N} \beta_m K(x_m, x_n))\right) \tag{8}$$

3 Multi Kernel Regression

3.1 Heuristic Approaches

data-dependent:

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m(x_i, x_j) k_m(x_i, x_j) \tag{9}$$

data-independent:

$$k_\eta(x_i, x_j) = \sum_{m=1}^{P} \eta_m k_m(x_i, x_j) \tag{10}$$

where we select the weights by looking at the performance values obtained by each kernel separately.

3.2 Kernel Alignment

The empirical alignment of two kernels is defined as:

$$A(K1, K2) = \frac{< K1, K2 >_F}{\sqrt{< K1, K2 >_F < K1, K2 >_F}} \tag{11}$$

where

$$< K_1, K_2 >_F = \sum_{i=1}^{N} \sum_{j=1}^{N} k_1(x_i.x_j) k_2(x_i, x_j) \tag{12}$$

We can select the kernel weights using alignment directly:

$$\eta_m = \frac{A(K_m, yy^T)}{\sum_{h=1}^{P} A(K_h, yy^T)} \tag{13}$$

where we call yy^T *ideal kernel*

$$A(K, yy^T) = \frac{< K, yy^T >_F}{N\sqrt{< K, K >_F}} \tag{14}$$

In our model, we try to introduce cost functions of weights η

3.3 Approach of multi-kernel on multi-task

We can combine different tasks' cost functions in one equation, and add another term considering weights.

$$min_{\beta,\eta} \sum_{t=1}^{T} [Loss_{regression,t} + Loss_{\eta_t}] + Penalty_\eta \tag{15}$$

$$Loss_{regression,t} = \frac{\lambda_t}{N_t} \sum_{m=1}^{N_t} \sum_{n=1}^{N_t} \beta_{tm} \beta_{tn} K_t(x_{tm}, x_{tn}) + \frac{1}{N_t} \sum_{n=1}^{N_t} \log \left(1 + exp(-y_n \sum_{m=1}^{N_t} \beta_m K_t(x_m, x_n)) \right) \tag{16}$$

Assume the probability of using m-th kernel given data D_i is

$$p(k_m|D_i) \propto exp \left[\frac{A(K_m, yy^T)}{\sum_{h=1}^{P} A(K_h, yy^T)} \right] \tag{17}$$

thus the likelihood is

$$L = (p(k_1|D))^{I(m=1)} (p(k_2|D))^{I(m=2)} ... (p(k_P|D))^{I(m=P)} \tag{18}$$

Using η_m to replace $I(m)$, and log it

$$loss_i = -\sum_{m=1}^{P} \eta_m \left[\frac{A(K_m, yy^T)}{\sum_{h=1}^{P} A(K_h, yy^T)} \right] \tag{19}$$

for T tasks, each task has data itself

$$loss_\eta = -\sum_{t}^{T} \sum_{m=1}^{P} \eta_{tm} \left[\frac{A(K_{tm}, y_t y_t^T)}{\sum_{h=1}^{P} A(K_{th}, y_t y_t^T)} \right] \tag{20}$$

Constraints on η are for $t = 1, ..., T; i = 1, ..., P$

$$\eta_{ti} \geq 0 \tag{21}$$

$$\sum_{i=1}^{P} \eta_{ti}^2 = 1 \tag{22}$$

3.4 With Joint Sparse Feature Selection

$$L_{penalty} = \lambda P_\infty = \lambda \sum_{j=1}^{P} \max_t(|\eta_{tj}|) \tag{23}$$

So the process is
Step 1: solve the optimization of loss of η

$$\eta = argmin_\eta(loss_\eta + L_{penalty}) \tag{24}$$

Step 2: using derived η to solve the optimization of loss of β: equation 16

3.5 Prediction

After solving problem above, we can derive β and η β_{ti} is the weight of i-th sample in task t. η_{tm} is the weight of m-th kernel in task t.

Given one sample x, and we know it belongs to task t, then we can predict its y:

$$y = \frac{1}{1 + exp\left(-w^T x\right))} = \frac{1}{1 + exp\left(-\sum_{i=1}^{N_t} \beta_{ti} \sum_{m=1}^{P} \eta_{tm} k_m(x, x_i)\right))} \tag{25}$$

4 Experiments

4.1 Data

- positive samples: may cause, may be caused by, ddx, is a risk factor for, is a, diagnoses
- negative samples:
  - get other top frequent relations in umls.MRREL involving DISO/LAB
  - filter potential false negative relations (e.g. disease_may_have_cytogenetic_abnormality) and inverse relations (e.g. "finding_site_of", "has_finding_site")
- statistics
  - 294719 negative samples
  - 262814 positive samples
- filter with prefer terms (delete sample which prefer terms can not be found)
  - 294718 negative samples
  - 262486 positive samples

4.2 Experiment Setup

From raw data, we have entity pairs in 6 types of relations. We view each relation as a task in our model.

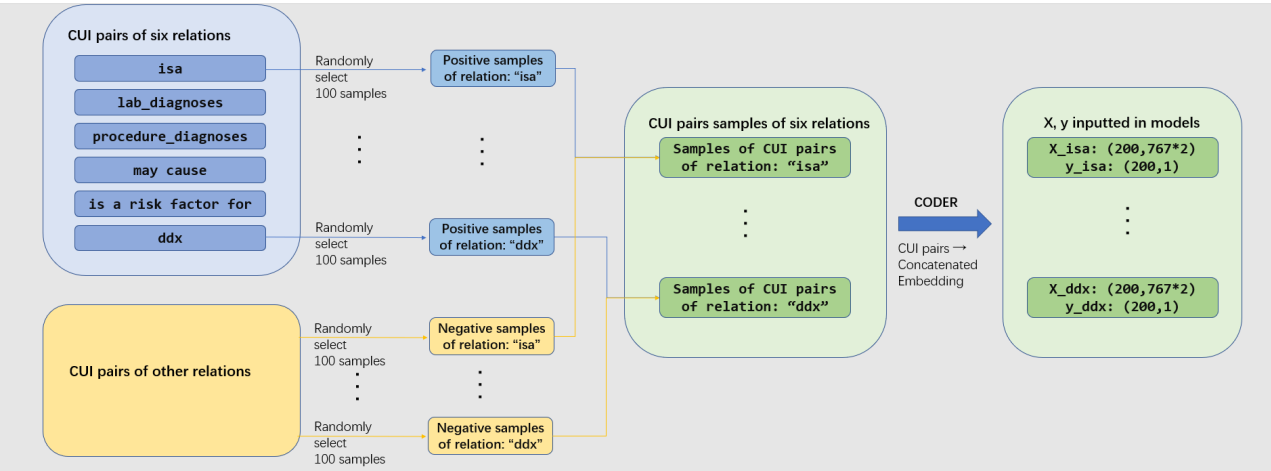

Figure 1: process of dealing data

4.3 Models

- four models: single-task single-kernel, single-task multi-kernel, multi-task single-kernel, multi-task multi-kernel (our model)
- we do 5-fold cross validation for each model, and calculate the average performance on valid-set.
  - single-task single-kernel: logistic regression training on each relation. Calculate the average on different relations.
  - single-task multi-kernel: logistic kernel regression training on each relation. Calculate the average on different relations.
  - multi-task single-kernel: multi-task logistic regression training on all relations.
  - multi-task multi-kernel: our model, training on all relations.

4.3.1 Results

For several datasets we randomly selected, the 5-fold cross validation results are as followed.

fold	1	2	3	4	5	Average
Single-Task Single-Kernel	0.9329	0.9413	0.9438	0.9250	0.9371	0.9360
Single-Task Multi-Kernel	0.9296	0.9483	0.9392	0.9383	0.9479	0.9407
Multi-Task Single-Kernel	0.9271	0.9600	0.9292	0.9196	0.9767	0.9425
Multi-Task Multi-Kernel	0.9350	0.9533	0.9450	0.9504	0.9513	**0.9470**

fold	1	2	3	4	5	Average
Single-Task Single-Kernel	0.9358	0.9154	0.8638	0.9125	0.8942	0.9043
Single-Task Multi-Kernel	0.9517	0.9292	0.9129	0.9254	0.8942	0.9227
Multi-Task Single-Kernel	0.9358	0.9229	0.9250	0.9288	0.9375	0.9300
Multi-Task Multi-Kernel	0.9633	0.9363	0.9129	0.9329	0.9171	**0.9325**