**基于自适应软参数共享的深度多任务学习方法**

# 引言

大气污染作为中国三大环境公害之一，已被中央政府列入环境治理的重点对象。国务院于2013年出台《大气污染防治行动计划》， 2018年出台《蓝天保卫战三年行动计划》。《中共中央关于制定国民经济和社会发展第十四个五年规划和二〇三五年远景目标的建议》中也明确指出，强化多污染物协同控制和区域协同治理，加强细颗粒物和臭氧（*O3*）协同控制，基本消除重污染天气。*PM2.5*和*O3*的协同控制，成为我国“十四五”及更长时期的一个重要任务。实施大气污染防治，是推进生态文明建设的重要举措，也是解决民生环境问题的必然要求。大气污染物的种类很多，目前引起人们注意的有100多种，如颗粒物、硫氧化物、氮氧化物等，目前影响空气质量的最重要因素是*PM2.5*和*O3*。同时考虑*PM2.5*与*O3*和影响因素的关系，期望实现协同治理，需要用到多任务学习。

多任务学习（MTL）在环境科学、经济科学、空气污染等许多领域中有重要的应用。多任务学习旨在通过利用它们之间的有用信息来提高多个相关学习任务的性能Zhang and Yang (2018)。对于给定的模型，利用任务之间的相关性，对多个任务进行联合训练，来更好地概括我们的原始任务，通过共享表示从而提高模型在多个任务上的学习能力。 Vandenhende et al.(2020)指出多任务学习能够从多任务监督信号中学习共享表示的网络, 如果相关任务共享互补信息或充当彼此的正则化器，它们就有可能提高性能。

基于深度学习的多任务学习最早可追溯到1993年，Caruana(1993)提出了神经网络中硬参数共享的结构，同时基于实证研究提出了多任务学习起作用的几个可能的机制。与多任务学习相结合的深度学习框架主要有下面两种方法：（1）基于硬共享的多任务深度学习方法。硬参数共享可以大大降低过度拟合的风险（Ruder，2017）。它在每一层共享所有参数的信息，例如Collobert 和 Weston（2008）将词性标注、词块分割、命名实体识别及词语相似度任务统一到一个语言模型中, 利用其他任务中自动学习的特征 来 提 升 语 义 角 色 标 注 任 务 的 性 能。Subramanian et al. (2018)和Liu et al. (2019)使用硬共享方法将特定于任务的层堆叠在共享层的顶部。尽管参数的硬共享机制在许多场景中有用，但是若任务间的联系不那么紧密，则硬共享技术很快失效。（2）基于软共享的多任务深度学习方法。参数的软共享机制，可以不用考虑任务的相关性，对于联系不那么紧密的任务也可以起到很好的优化效果。软共享技术可以充分利用多任务学习中的隐世数据增加机制、窃听机制，注意力集中机制，正则化机制等优势。由于每个任务分配一个网络，常常需要增加很多参数，每个任务都有自己的模型和参数。软参数共享网络对模型参数之间的距离进行正则化，以鼓励参数相似，例如Long et al.(2015)使用L2距离进行正则化，而Yang et al. (2017)使用轨迹范数，同时MTL正则化约束也给了软参数共享很大的启发。

目前，在联合训练多个任务时, 提高模型在某一项任务上的性能会损害具有不同需求的任务的性能，这种现象称为负迁移现象（Crawshaw，2020）。最小化负迁移现象是 MTL 方法的一个关键目标。在日常收集到的数据中并不是所有任务的联系都那么紧密，软共享技术就变的尤其重要。软参数共享中正则化约束的选择往往决定了预测模型的性能。基于此，本文在采用范数约束，在Li et al.(2018)均值约束共享的基础上，通过任务之间参数的差异性，设置软参数正则项系数，自适应正则项系数，自适应参数衰减比例，得到累积自适应参数衰减比例，将软共享多任务学习自适应地动态转化为多个单任务和多任务联合学习。在此过程中，寻找预测性能最好的自适应软参数共享模型。

本文的创新点：1. ；2. ；

# 本文结构如下：第 2 章将具体介绍本文的新模型框架，理论价值以及其具体的算法。第3章是模拟, 实例在第 4 章, 第 5 章是理论证明, 最后第 6 章是总结。2. 模型与方法

## 2.1 软参数共享模型

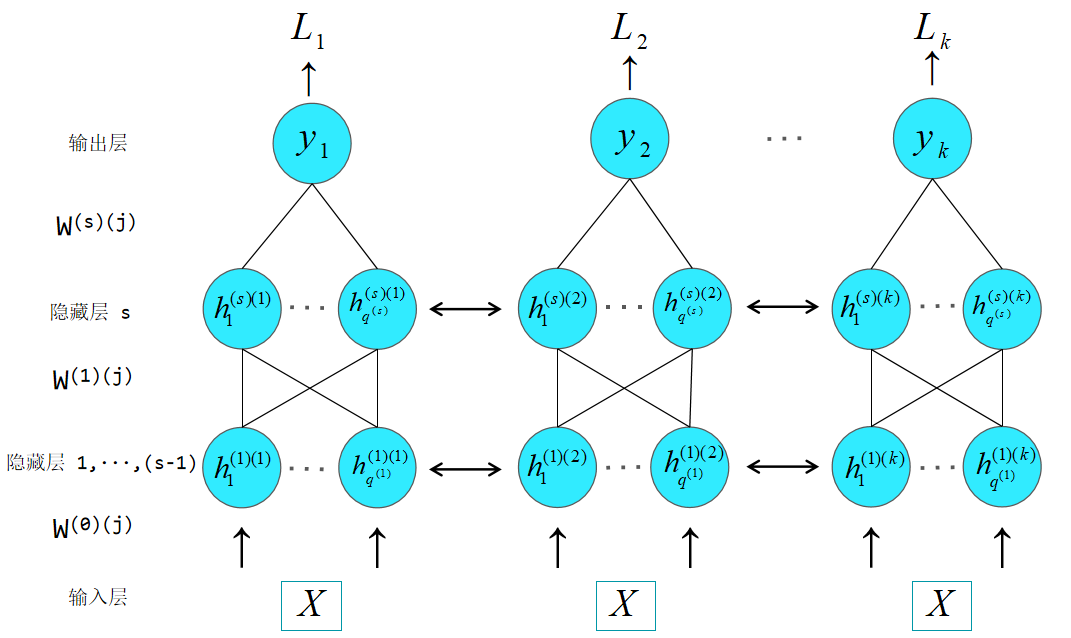


图1 软参数共享模型

软参数共享模型如图1所示，假设空间足够大，可以同时包含多个任务的解。假设一共有维特征，有个任务，则共有个输入，个输出，s个隐藏层。记输入向量为，输出向量，共享层一共有个节点。记节点矩阵为**，**。第j个任务的权重矩阵为，其中，针对第j个任务，共享层到共享层的权重矩阵为，其中，，偏置矩阵为。（检查一下下标是否都正确？）

## 2.2 均值约束共享的软参数多任务学习方法

软参数共享可以不用考虑任务之间的相关性大小，通过参数与参数之间的关系，使得多个任务可以有效地利用它们之间的信息，从而享受多任务联合训练的优势。本文采用的L2距离正则化来建立多个任务之间的关系。在软参数共享中，参数不完全共用，只是用正则化项来建立参数之间的关系。我们记作为任务j的真实值，那么任务j的预测值可以表示为:

。 (1)

其中是模型的参数。本文采用Li et al.(2018)均值约束共享的方法来建立正则化约束。该方法认为任务之间有一定的相关性，但是可以不用知道任务之间相关性的强弱，认为不同模型任务之间的相关性是通过模型参数相互接近来建模的，考虑了每一个任务的模型参数与所有任务的模型参数的均值，来建模任务之间的相关性。

每个样本在训练集上的损失函数计算公式为:

, (2)

, (3)

其中，正则项系数代表了多个任务之间的相似度，越大，任务之间的参数就越相接近，任务之间的关系就越紧密。

在验证集上计算不同任务的损失函数（Lj是什么？），选择使整体损失函数式最小化的参数方案，以测试集上的损失函数来作为预测性能的评估指标，其损失函数的计算公式为:

， (4)

. (5)

（4）（5）右边有何不同？

在深度多任务学习中常采用均方误差(root mean square error, MSE) 作为整体模型预测性能评价指标，计算公式为:

 （哪里有i?n=?后面表示X的维数） (6)

## 2.3 自适应软参数共享的多任务学习算法

软参数共享可以让多个任务联合训练，使得任务之间可以互相利用有效的信息进行学习，但负迁移现象往往难以得到有效的处理。在深度多任务学习中常常存在多个损失的局部极小值点，使得多个任务更精确地达到它们损失函数的全局最小点是我们的目标。基于此，我们提出了自适应软参数共享的多任务学习算法。

软参数共享多任务学习方法使得多个任务同时达到它们相对的损失函数最小值，对于每个任务而言，它们仍有很大的优化空间。在软参数多任务学习后，任务与任务之间的参数差异越大，则任务之间的相似度越低。基于此，我们提出了自适应软参数共享的多任务学习算法，在软参数多任务学习收敛后，将（2）式



中的软参数正则项系数设置为自适应正则项系数。通过去掉训练集损失函数中正则项里权重差最大的部分参数矩阵，使得相似度最低的部分任务独立地进行训练，以达到最小化负迁移现象的效果。我们以两个任务为例，图2，图3，图4分别表示两个任务单独学习，软参数多任务学习以及自适应软参数多任务学习的效果图。图中两条曲线的最低点分别表示两个任务损失函数的极小值点，蓝色和红色虚线分别代表了任务1和任务2在软参数多任务学习结束后，进行自适应软参数多任务学习时两个任务的损失函数关于参数的变化情况。（3幅图都是这样？黄色的点表示什么？没有解释为何自适应软参数多任务学习优于其他两个方法）

图2 两个任务单独学习

图3 软参数多任务学习

图4 自适应软参数多任务学习

本文中的参数衰减指的是在的正则项中进行参数衰减，接下来介绍超参数的含义以及自适应软参数共享的多任务学习算法。

自适应参数衰减比例: 该超参数是指每次进行软参数自适应多任务学习时，在训练集正则项的所有项中删除数值较大的前项。例如在正则项中，一共有项，在进行下一轮迭代前，按数值从高到低排序，删除项。

累计自适应参数衰减比例:该超参数是指在的参数方案前，累计删除的数值为前的项，它是的倍数。

自适应正则项系数:该超参数与在进行软参数多任务学习的正则化系数意义相同，表示多个任务之间的相似度，随着参数衰减，任务的相似度将会降低。

自适应软参数共享的多任务学习算法详细描述了损失函数的变化过程，设置了自适应参数衰减比例, 自适应正则项系数。

设置软参数正则项系数和自适应正则项系数是为了通过任务与任务之间的相关性进行建模。设置了自适应参数衰减比例是为了自适应更精确地找到验证集上损失函数的最小值。

**自适应软参数共享的多任务学习算法**

输入:训练集 其中（Y也是n维？），软参数正则项系数，自适应正则项系数，自适应参数衰减比例。

输出: 第+1轮的自适应参数矩阵**（记号前后要一致）**，偏置矩阵, 训练集上的损失函数，验证集上的损失函数，测试集上的损失函数，累积自适应参数衰减比例。

1. 设置软参数正则项系数，根据adam算法，通过误差反向传播更新参数矩阵**，**偏置矩阵进行迭代。直至收敛后停止迭代。
2. 记，为软参数多任务迭代收敛时的参数矩阵与偏置矩阵，第轮迭代结束时的参数矩阵与偏置矩阵记为和。设置自适应正则项系数，根据自适应参数衰减比例（这个是随着迭代在变的，是t?，或者写清楚怎么变的），在第轮迭代结束时，将中的正则项按数值从高到低排序删除个项。
3. 进行第轮迭代，直至收敛，更新参数矩阵，偏置矩阵。
4. 重复步骤2和3，一共进行轮迭代。
5. 在轮迭代中寻找收敛最低点记为，选择下的参数方案，并该参数方案下的。
6. 在测试集上评估模型性能，计算。

# 3. 模拟

为了简单起见，本节k=2，模型如下：

，（哪里面有i？） 

情形1: ，是一个4\*3维矩阵，其中 

。是一个3\*1维向量，其中 。是一个n\*3维矩阵，其中 。是一个n\*1维向量，其中 。是一个n\*1维向量，其中 。是一个n\*2维矩阵。（所有的n是不是都要换成N）

情形2: 是一个n\*4维矩阵，其中 。是一个4\*3维矩阵，其中 。是一个3\*1维向量，其中 。是一个n\*3维矩阵，其中 。是一个n\*1维向量，其中 。是一个n\*1维向量，其中 。是一个n\*2维矩阵。

情形3: 是一个n\*4维矩阵，其中 。是一个4\*3维矩阵，其中 。是一个3\*1维向量，其中 。 是一个n\*3维矩阵，其中 。是一个n\*1维向量，其中 。是一个n\*1维向量，其中 。是一个n\*2维矩阵。

本文采用Adam算法Kingma and Ba （2015)更新参数







其中代表学习率, 表示更新的步长, 表示过往梯度与当前梯度的均值, 表示过往梯度的平滑与当前梯度平方的均值, 代表经过矫正后过往梯度与当前梯度的均值, 代表经过矫正后过往梯度与当前梯度平方的均值, 是为了维持数值稳定性而添加的常数, 和代表的是指数衰减率。

本文构建的算法程序包括: CPU为r7-4800H, GPU为RTX2060, 操作系统为 64 位 windows，开发语言为Python。 所有方法的实现均基于**pytorch**框架实现的。设置输入层的神经元数为m, 共享层1的神经元数为4, 共享层2的神经元数为3, 输出层的神经元数为2。 Adam 优化算法学习率=0.001，指数衰减率，

=0.9999。

对于情形1，情形2，情形3，经过调参可以得到最优的超参数如下:

表1 模拟的超参数表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| 均匀分布 | 0.001 | 0.00043 | 0.07 | 0.63 |
| 标准正态分布 | 0.04 | 0.0176 | 0.02 | 0.38 |
| 指数分布 | 0.0006 | 0.00034 | 0.1 | 0.80 |

其中表示在进行软参数多任务学习时，使得软参数多任务学习的损失函数达到最小值的最优参数。和表示在进行自适应软参数多任务学习时，使得自适应软参数多任务学习的损失函数达到最小值的最优参数，是与之相对应的最优参数。

对于上述情形分别设置与软参数多任务学习和两个单任务学习的对比实验，

为保证对比的有效性，我们分别将三种学习损失函数的最低值进行对比。对于上面三种模拟情形，分别取样本量N为2000，3000和4000。整体数据集按7:3的比例分别随机划分为训练集和测试集，将训练集中一部分数据作为验证集，用于选择出最优的模型。测试集用于评估模型性能和稳健性，并采用100轮结果的均值作为实验结果。

表2 三种情形100次模拟MSE的均值(10-4)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | N=2000 | | | N=3000 | | | N=4000 | | |
|  | 单任务 | 软参数 | 自适应 | 单任务 | 软参数 | 自适应 | 单任务 | 软参数 | 自适应 |
| 均匀分布 | 3.115 | 2.588 | 2.417 | 2.770 | 2.167 | 1.974 | 1.474 | 1.239 | 1.116 |
| 标准正态分布 | 9.620 | 7.272 | 6.539 | 7.631 | 6.086 | 5.770 | 6.258 | 4.628 | 4.183 |
| 指数分布 | 0.874 | 0.682 | 0.645 | 0.679 | 0.457 | 0.392 | 0.285 | 0.213 | 0.190 |

表2分别为情形1，情形2，情形3在单任务学习，软参数多任务学习和自适应软参数多任务学习上的对比实验结果。从表2可以看出，自适应软参数多任务学习优于软参数多任务学习，软参数多任务学习优于单任务学习，且随着样本量的增大，三种情形的MSE的值越来越小。

性能提升值的计算公式如下:

，

其中s代表软参数多任务学习的MSE， a代表自适应软参数多任务学习的MSE。

表3 二种模型2000个样本100次模拟的平均耗时(s)以及性能提升值(%)

|  |  |  |  |
| --- | --- | --- | --- |
|  | 软参数耗时 | 自适应耗时 | 性能提升值 |
| 均匀分布 | 91 | 106 | 6.6 |
| 标准正态分布 | 82 | 98 | 10.1 |
| 指数分布 | 86 | 97 | 5.4 |

将自适应软参数多任务学习和软参数多任务学习做对比，且估计效率的衡量仍采用MSE, 表3表示模拟数据的性能提升值以及耗时表，就2000个样本而言,从提升值可以看出，自适应软参数多任务学习相比于软参数多任务学习提升值都超过5.4%。在耗时方面，自适应软参数多任务学习在3个模拟数据上的平均每轮模拟耗时与软参数多任务学习相比在16s之内，不难看出，自适应软参数多任务学习的整体性能优于软参数多任务学习，说明了自适应软参数多任务学习相较于软参数多任务学习具有良好的预测和估计能力，代价仅仅是少量的时间。

为了具体展示自适应软参数多任务学习的细节，本文展示了样本量为2000的模拟数据分别为均匀分布、标准正态分布和指数分布在500次迭代之后继续迭代的软参数多任务学习与自适应软参数多任务学习的MSE变化对比，图中黄点代表软参数多任务学习的损失函数最低点，黄点之后代表软参数多任务学习与自适应软参数多任务学习的损失函数变化对比，红点代表累计自适应参数衰减比例取得时的自适应软参数多任务学习的损失函数的最低点。从图4-6中可以看出在软参数多任务学习迭代到达最低点的时候，三个模拟可以通过自适应软参数多任务学习算法进一步优化，使得它们的损失函数进一步下降，提升了预测效果，说明了自适应软参数多任务学习良好的性能。

图4 输入数据为均分分布的软参数与自适应多任务学习的MSE变化对比图

图5 输入数据为标准正态分布的软参数与自适应多任务学习的MSE变化对比图

图6 输入数据为指数分布的软参数与自适应多任务学习的MSE变化对比图

# 4. 实例

本节考虑南京市空气污染数据，该数据来源于中国气象数据网和真气环境大数据中心。数据包含了南京市2013年12月2日至2022年11月23日每一天的PM2.5浓度()、PM10浓度()、CO浓度()、SO2浓度()、NO2浓度()、臭氧8小时O3\_8h()这个7变量，其中PM2.5浓度和臭氧8小时 O3\_8h()分别为输出变量和。PM10浓度、CO浓度、SO2浓度和NO2浓度分别为输入变量,,和，样本量n为3274个。具体考虑输入变量,,和来估计未来两个任务和的输出值。

空气污染预测模型： 。

是一个n\*4维矩阵，是一个4\*3维矩阵， 是一个3\*1维向量，是一个n\*3维矩阵，是一个n\*1维向量，，输出是一个n\*2维矩阵。

对于此预测模型，经过调参可以得到最优的超参数如下:

表4 实例的超参数

|  |  |
| --- | --- |
| 超参数 | 数值 |
|  | 0.0115 |
|  | 0.0071 |
|  | 0.045 |
|  | 0.54 |

将自适应软参数多任务学习和软参数多任务学习做对比，且估计效率的衡量仍采用MSE。由表5可知，自适应软参数多任务学习比软参数多任务学习仅仅多了13s，自适应软参数多任务学习比软参数多任务学习的效果提升了3.4%。

表5 二种模型的MSE值(10-2) 、耗时(s)及性能提升值(%)

|  |  |  |  |
| --- | --- | --- | --- |
|  | MSE | 时间 | 性能提升值 |
| 软参数 | 1.489 | 124 | 3.4 |
| 自适应 | 1.438 | 137 |

图7 南京市空气污染数据软参数与自适应的MSE变化对比图

图7是南京市空气污染数据软参数与自适应的MSE变化对比图，图7中的标记点与模拟数据时的标记点含义一样，从图7中可以看出对于南京市空气污染数据，在软参数多任务学习达到其损失函数的最低点后，通过自适应软参数多任务学习仍可以进一步优化，使得它们的损失函数进一步下降，提升了预测效果，说明了自适应软参数多任务学习良好的性能。

# 参考文献

Li A, Wu Z, Lu H, et al. Collaborative self-regression method with nonlinear feature based on multi-task learning for image classification[J]. IEEE Access, 2018, 6: 43513-43525.(多任务学习在图像分类)

Zhang Y, Yang Q. An overview of multi-task learning[J]. National Science Review, 2018, 5(1): 30-43.

M.Gao,L.Yin,J.Ning,Artifificial neural network model for ozone concentration estimation and Monte Carlo analysis,Atmospheric Environment 184(2018) 129-139.

S.Park,M.Kim,M.KIM,H.-G.Namgung,K-T.Kim,K.H.Cho,S.-B.Kwon,Predicting PM10 concentration in Seoul metropolitan subway stations using artifificial neural network(ANN),Journal of Hazardous Materials 341(2018)75-82.

P.J.G.Nieto,G.-G.E,A.B.S，anchez,A.A.R.Miranda,Air quality modeling using the PSO SVM-based approach,MLP neural network,and M5 model tree in the metropolitan area of Oviedo(Northern Spain),Environmental Modeling and Assessment 23(3)(2018)229-247.

Y.LeCun,Y.Bengio,G.Hinton,Deeplearning,Nature 521(7553)(2015)436-444.

J.Schmidhuber,Deep learning in neural networks:an overview,Neural Networks 61(2015)85-117.

Vandenhende S, Georgoulis S, Proesmans M, et al. Revisiting multi-task learning in the deep learning era[J]. arXiv preprint arXiv:2004.13379, 2020, 2(3).

Caruana R A. Multitask Learning: A Knowledge-Based Source of Inductive Bias[J]. Machine Learning Proceedings, 1993, 10(1):41-48.

Collobert R,Weston J.A unified architecture for natural language processing:Deepneural networks with multitask learning[C]Proceedings of the 25th international conference on Machine learning-lCML,08,New York:ACM Press,2008:160-167.

Liu, X.; He, P.; Chen, W.; and Gao, J. 2019. Multi-taskdeep neural networks for natural language understanding. In ACL, 4487–4496.

Subramanian, S.; Trischler, A.; Bengio, Y.; and Pal, C. J.2018. Learning general purpose distributed sentence representations via large scale multi-task learning. In ICLR.

Yang, Y. et. al. 2017. Trace Norm Regularized Deep Multi-Task Learning. ICLR2017 workshop.

Long D, Cohn T, Bird S, et al. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser[C]// Meeting of the Association for Computational Linguistics and the, International Joint Conference on Natural Language Processing. 2015:845-850.

Crawshaw M. Multi-task learning with deep neural networks: A survey[J]. arXiv preprint arXiv:2009.09796, 2020.

Ruder S. An overview of multi-task learning in deep neural networks[J]. arXiv preprint arXiv:1706.05098, 2017.

Kingma D P, Ba J. Adam: A Method for Stochastic Optimization[A]. International Conference on Learning Representations (ICLR), 2015: 1‒15．