TransTab: Learning Transferable Tabular Transformers Across Tables

刘晨

摘要

表格数据是机器学习中非常常见的形式，然而机器学习模型要求表的结构在训练和测试中保持不变。因此在使用机器学习算法进行学习时，需要对数据进行预处理，清除掉不同表中不同的列，但是这通常会造成严重的数据浪费。还有很多其它问题，比如模型如何从多个表的部分重叠列学习? 随着时间的推移，列的数量可能会越来越多，如何增量地更新模型? 我们能否在多个不同的表上进行模型预训练? 模型如何预测没有见过的样本?

针对这些问题，本文提出了一种针对表格数据的可转移的 transformer 来解除对表格格式限制的要求。并且对上面的问题也提出了解决方案。Transtab 的目标是将每个（表中的一行）转换为可推广的嵌入向量，然后使用 transformer 来处理这些特征。

关键词：embedding；transformer

1 引言

表格数据在医疗保健、工程、广告和金融领域[1-4]无处不在，通常以表或电子表格的形式存储在关系数据库中。表的一行表一个示数据样本，列则表示不同数据类型的特征变量 (例如，类别、数字、二进制和文本)。传统的机器学习方法要求在训练和测试数据中使用相同的表结构。然而，在现实世界中可能有多个表共享部分重叠的列。因此，跨表学习是不适用的。一种常见的方法是在训练任何机器模型之前先进行数据清理，删除不同表中不匹配的样本，这会造成严重的数据资源浪费[5]。因此，能够进行不同列的表学习和跨表传递知识是将深度学习和预训练成功扩展到表格领域的关键。

表格数据具有高度的结构性和灵活性。因此实现跨表学习的第一步就是找出建模的基本元素。在 CV 领域中，最基本的元素是一个像素点[6]；在 NLP 领域中，最基本的元素是一个单词[7]或者一个 token[8-9]。所以，在表格学习中每一个单元格 (cell) 的值就是一个基本元素。然后列会被映射成唯一的索引，然后模型使用单元格的值进行训练和推断。但是这个前提是所有训练数据中表的结构要保持一致。但是表通常有不同的协议，其中列的命名规则也不同。

2 相关工作

2.1 传统表格学习

传统的机器学习方法要求在训练和测试数据中使用相同的表结构。一旦表格的 column 有一丝丝变化，比如 age->ages，或者 age 这一列被删掉。那么之前训练好的模型就没用了，只能重新进行数据处理-> 特征工程-> 模型训练这一流程。这无疑是不利于我们实现像 CV 和 NLP 里那样做表格学习的大模型训练的。

2.2 transtab 中的表格学习

transtab 中的表格学习和传统的表格学习有很大不同。例如，传统的方法通过引用 codebook{男性: 0, 女性: 1} 表示单元格中值为 0 的性别是男性。而 transtab 将表格输入转换为序列输入 (例如，性别是男性)，这种序列可以用下游序列模型建模。我们认为这种特征化协议是可跨表通用的，因此使模型能够从不同的表中学到知识。

作者认为 transtab 能够成功是因为下面两个关键因素：

- 考虑了单元格语义和列语义，并以此来进行跨表学习
- 作者提出了一种结构 VPCL(Vertical-Partition Contrastive Learning) 使模型能够在多个表上进行预训练同时可以在目标数据集上进行微调。

3 本文方法

3.1 本文方法概述

此部分对本文将要复现的工作进行概述，图的插入如图 1所示：在这一部分中，我们将介绍 TransTab 的详细信息。图 1显示了它的工作流程，包括以下关键组件:

1. 使用 Input Processor 对表格中数据进行转换为嵌入向量；
2. 使用 transformer 对上面得到的嵌入向量进一步编码处理；
3. 最后，模型包括对标记数据进行训练的分类器和用于对比学习的投影仪。

接下来我们将介绍每个组件的详细信息。

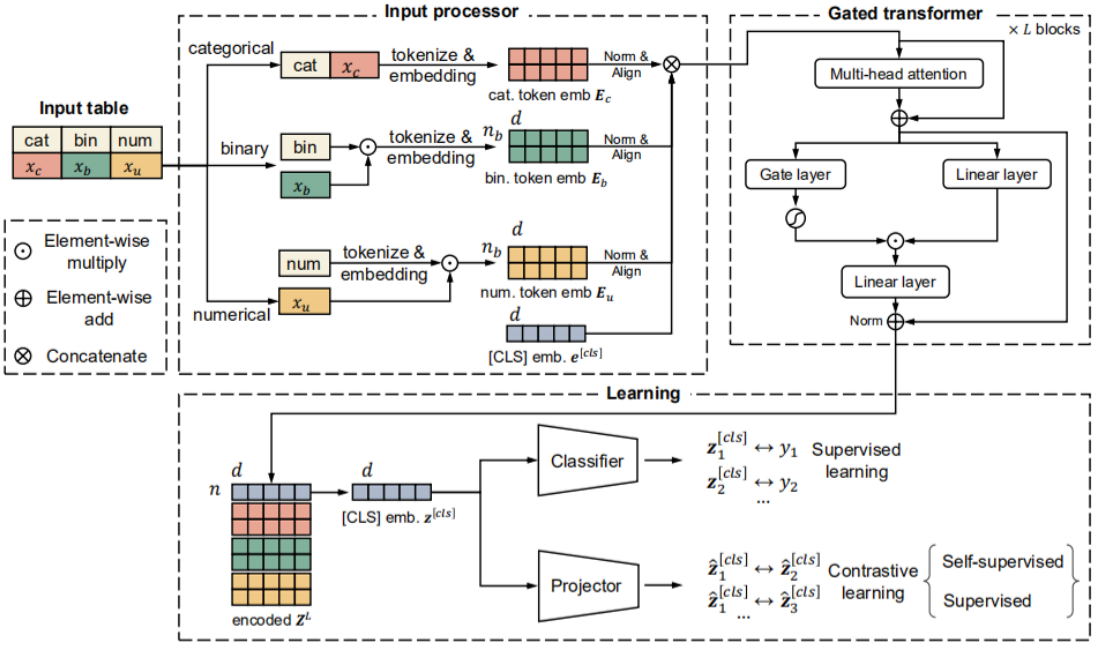

图 1: 模型结构图

3.2 transtab 的应用场景

在详细介绍我们的方法之前，我们首先介绍 TransTab 可以处理的四个新的应用场景，如图 2所示。假设我们的目标是使用多个临床试验表来预测乳腺癌试验的治疗效果，下面是我们经常遇到的几种情况。

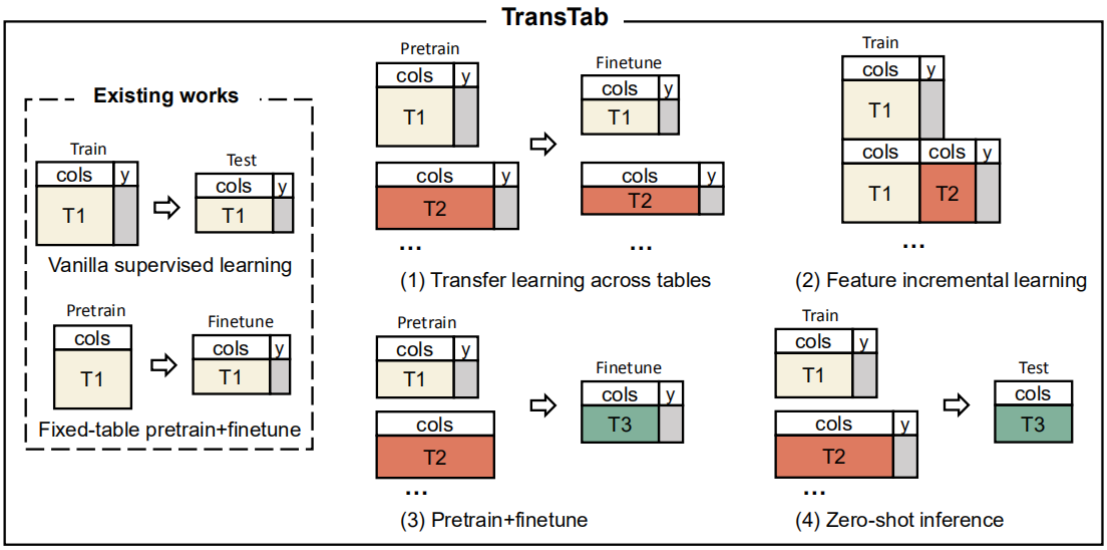

图 2: 应用场景图

1. **Transfer learning**：我们从多个癌症试验中收集数据表，以测试同一种药物对不同患者的疗效。这些表是独立设计的，部分列重叠。我们如何通过利用所有试验的表来学习一个试验的机器学习模型?

2. **Incremental learning**：随着时间的推移，可能会添加其他列。例如，在不同的试验阶段收集额外的特性。我们如何使用所有试验阶段的表来更新机器学习模型?

3. **Pretraining+Finetuning**：试验结果标签 (例如，死亡率) 可能并不总是可以从所有表格来源获得。能够在这些没有标签的表格数据上进行预训练? 我们如何使用标签对目标表上的模型进行微调?

4. **Zero-shot inference**：我们根据我们的试验记录来模拟药物的疗效。下一步是对模型进行推断，以找到可以从药物中获益的患者。然而，患者表不与试验表共享相同的列，因此直接推断是不可能的。

接下来，我们将介绍 TransTab 并演示它如何处理这些场景。

3.3 Input Processor

构建 Input Processor 主要有两个目的：来接受变列表；可以跨表格数据集保留知识。其思想是将表格数据 (列中的单元格) 转换为语义编码的标记序列。事实上，如果我们仔细观察每一个表格，会发现其实表格的列名 (column names) 其实蕴含了丰富的信息。比如在下面中，age 列对应了 25，我们自然就知道是 25 岁；income 列对应了 10000，我们知道可能是 10000 个某种货币单位。如果 age 列改成 weight，25 就不再表示 25 岁，而是 25kg 或者其它重量单位。由此可见，表格中 feature 所代表的语义信息是由 column 所决定的。如果我们想要迁移表格的信息，则必须要考虑到 column 的语义。

根据这个思想，transtab 可以将所有的表格数据视为三大种类型的组合：**类别型 (categorical)**、**数值型 (numerical)**、**布尔型 (bool)**。图 1展示了对这三种不同类型的列数据操作的具体过程。

Categorical/Textual feature。一个类别或者文本特征通常包含一个文本 tokens 序列。所以做法是将列别名和特征值 x_c 直接拼接起来形成一个句子。然后这个句子会进行 tokenized 和 embedding 操作生成特征嵌入 E_c。

Binary feature。这种类型的特征 bin 通常是一个断言判断并且它的值 $x_b \in \{0,1\}$ 如果 $x_b = 1$，bin 会进行 tokenized 和 embedding 操作生成特征嵌入 E_b。否则则不会进行后续处理，这样在某些情况下可以显著降低计算的内存成本。

Numerical feature。不能将列名 num 和值直接拼接成数字特征，因为 tokenization-embedding 在区分数字方面表现的非常糟糕[10]。相反，需要单独处理它们, 列名 num 会上前面一样进行 tokenized 和 embedding 操作被编码成 $E_{u,col}$，然后使用数字特征与前面的列嵌入向量相乘得到 $Eu = E_{u,col} \times x_u$，作者认为这种方式比复杂的数值嵌入更有优势。

因此，所有单元格值都根据相应的列属性进行上下文化，因此一个元素的语义可能会根据上下文组合而变化。这种形式有利于跨表的知识转移。例如，以前吸烟和吸烟史描述的是一样的。以前的方法难以分辨它们，而 TransTab 可以学会识别两个列下的 1 是等价的。

3.4 Gated Transformers

论文中的 transformer 是对 NLP 领域中经典 transformer[11]的改进。主要由两部分组成：multi-head self-attention layer 和 gated feedforward layers。

首先采用第 l 层的输入 z^l 来表示特征之间的关系:

$$\mathbf{Z}_{\text{att}}^{l} = \text{MultiHeadAttn}\left(\mathbf{Z}^{l}\right) = \left[\text{head}_1, \text{head}_2, \ldots, \text{head}_h\right]\mathbf{W}^{O} \tag{1}$$

$$\text{head}_i = \text{Attention}\left(\mathbf{Z}^{l}\mathbf{W}_i^{Q}, \mathbf{Z}^{l}\mathbf{W}_i^{K}, \mathbf{Z}^{l}\mathbf{W}_i^{V}\right) \tag{2}$$

上面 multi-head attention 的输出将会被进一步转换获得 transformer 的输出:

$$\mathbf{Z}^{l+1} = \text{Linear}\left(\left(\mathbf{g}^{l} \odot \mathbf{Z}_{\text{att}}^{l}\right) \oplus \text{Linear}\left(\mathbf{Z}_{\text{att}}^{l}\right)\right) \tag{3}$$

3.5 TransTab 中的自监督和有监督预训练

假如有一系列上游表格，它们可能有标签或者没标签。我们在有标签的数据上更希望能利用到标签信息，那么可能的想法是：对于所有表格共享一个 backbone，然后对每一个表格单独设置一个 classification head 做 supervised learning。但是上面的做法会导致 backbone encoder 很难学到好的表征。因为每个表格的标签类别不同，或者可能是相反的定义（比如死亡率和存活率）。过程如图 3所示：

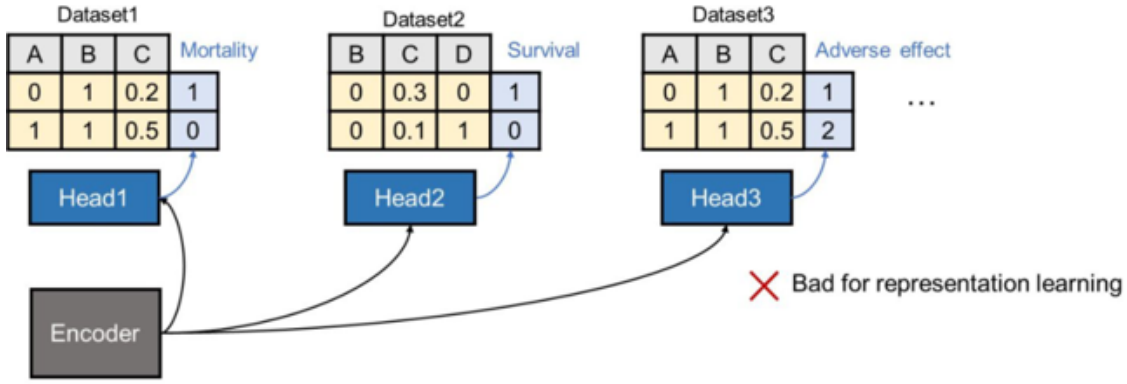

图 3: 多个表格上共享 backbone

考虑到这点，作者提出了一种基于标签的 supervised contrastive learning，在论文里叫做 vertical partition contrastive learning (VPCL)。VPCL 主要分为两类，一类是 Self-supervised VPCL，一类是 Supervised VPCL。

Self-supervised VPCL。大多数的自监督学习在训练时要求训练数据结构保持一致[2,12-13]，这会造成很高的计算成本并且容易发生过拟合现象。相反，transtab 使用表格垂直分区来为对比学习构建正样本和负样本，具体过程如图 4所示。Self-supervised VPCL 的主要思想就是对列数据进行分割，只有来自相同样本的其它分区是正例，其余的都是负例：

$$\ell(\mathbf{X}) = -\sum_{i=1}^{B}\sum_{k=1}^{K}\sum_{k'\neq k}^{K}\log\frac{\exp\psi\left(\mathbf{v}_i^k, \mathbf{v}_i^{k'}\right)}{\sum_{j=1}^{B}\sum_{k^\dagger=1}^{K}\exp\psi\left(\mathbf{v}_i^k, \mathbf{v}_j^{k^\dagger}\right)} \tag{4}$$

与寻常的对比学习 (如 SCARF[13]) 相比，Self-VPCL 显著扩展了正采样和负采样，从而学习到更丰富的嵌入特征。并且，这种垂直分区采样对面向列的数据库[14]非常友好，它支持从大型数据仓库快速查询列的子集。为了提高计算效率，当 K > 2 时，我们随机抽取两个分区。

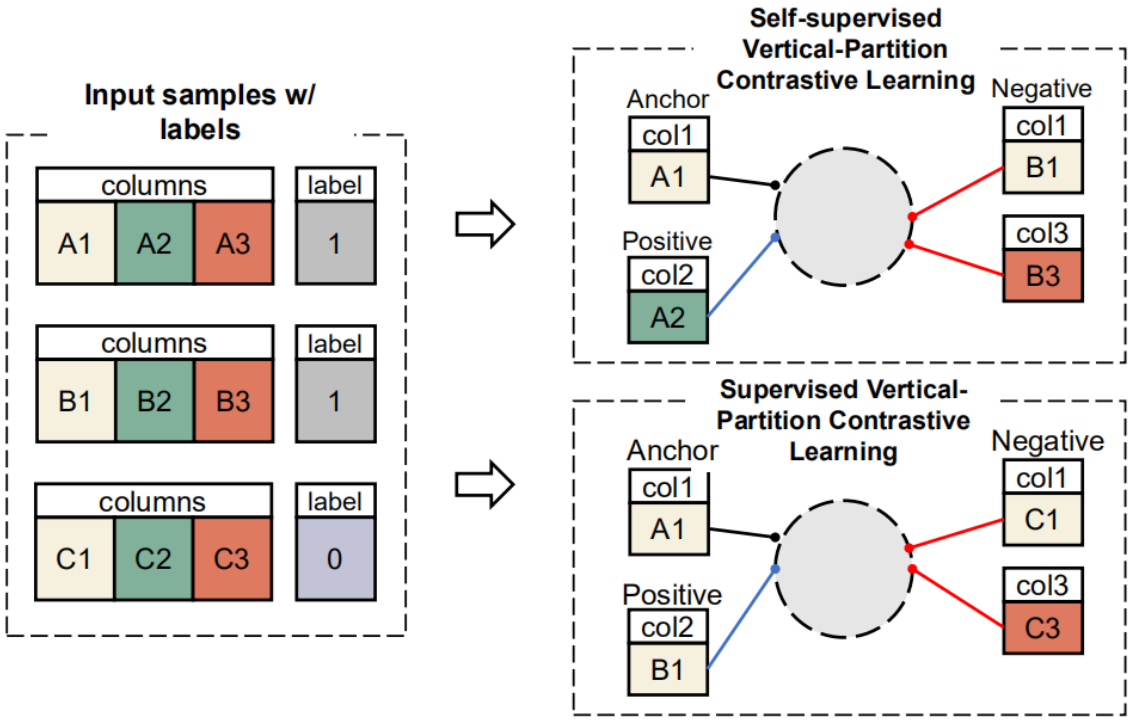

图 4: VPCL 的过程

Supervised VPCL 当我们使用有标签的表格数据进行预训练时，一个自然的想法是采用特定于任务的预测头进行常见监督损失的预训练，例如，交叉熵损失。在微调中，这些头被丢弃，一个新的头将被添加到预训练的编码器之上。然而，我们认为这是次优的，可能会破坏模型的可移植性。其背后的原因是表格数据集在大小、任务定义和类分布方面变化巨大。使用监督损失预训练 TransTab 不可避免地会导致编码器偏向于主要任务和类。此外，当应用监督损失时，适用的超参数范围通常在表格数据之间是不同的。同一组超参数可能导致一个数据集的过拟合和另一个数据集的欠拟合。因此，基于常见的监督损失选择合适的超参数进行预训练是一个棘手的问题。

在本文中，受有监督对比学习[15]的启发，我们提出了用于预训练的 VPCL，该方法对噪声和超参数具有鲁棒性。如图 4所示，只有来自相同一分区并且标签相同的样本是正例，其余的都是负例：

$$\ell(\mathbf{X}, \mathbf{y}) = -\sum_{i=1}^{B}\sum_{j=1}^{B}\sum_{k=1}^{K}\sum_{k'=1}^{K}\mathbf{1}\left\{y_j = y_i\right\}\log\frac{\exp\psi\left(\mathbf{v}_i^k, \mathbf{v}_j^{k'}\right)}{\sum_{j^\dagger=1}^{B}\sum_{k^\dagger=1}^{K}\mathbf{1}\left\{y_{j^\dagger} \neq y_i\right\}\exp\psi\left(\mathbf{v}_i^k, \mathbf{v}_{j^\dagger}^{k^\dagger}\right)} \tag{5}$$

4 复现细节

4.1 与已有开源代码对比

开源代码中只给出了分类部分，我主要跟根据作者给出的框架结构，向模型中添加了回归部分，主要修改了训练模型中的损失函数以及模型最后线性层最后的输出部分。但是实现的效果比较差，还需要进一步的改进。

4.2 实验环境搭建

数据预处理。实验中使用的是 openml 平台提供的数据集。与对于所有实验，如果指定了分类特征，那就使用使用序号编码表示分类特征，否则使用 one-hot 编码。数值特征通过 min-max normalization. 缩放到 [0,1]。对于布尔类型的数据，会将其中的 yes、true、t、1 全部替换为 1，剩下的比如 no、false、f、0 全都替换成 0。实验中使用众数来填充列中的缺失值。

实验参数设置。TransTab 使用 2 层 gated transformer，其中 embedding 和 token 的维数为 128，中间密集层的隐藏维数为 256。激活函数为 ReLU，不使用 dropout。使用 Adam[16]优化器训练 TransTab，学习率为 {2e- 5,5e - 5,1e -4}，没有权重衰减; 批处理大小为 {16,64,128}。最大预训练轮次为 50 次，监督训练轮次为 1000 次。早停策略的参数设置为 10。

实验环境。本次实验的硬件环境是 GPU：A100 RAM：16GB。实验使用的深度学习框架是 Pytorch1.10，Python 版本 3.9.12，同时实验中使用了 sklearn、pandas、pandas 等库进行数据的预处理。集体要求的环境都已经放在了 requirements.txt，只需要运行 pip install -r requirements.txt 即可。

4.3 创新点

作者给的开源代码只给出了分类部分,这个也不太符合现实的生活场景,所以我在作者的基础上添加了回归处理模块。首先是从 kaggle 上找了一个关于回归的数据集 WorldHappiness_Corruption_2015_2020，然后数据目录按照作者要求的格式如 5所示：

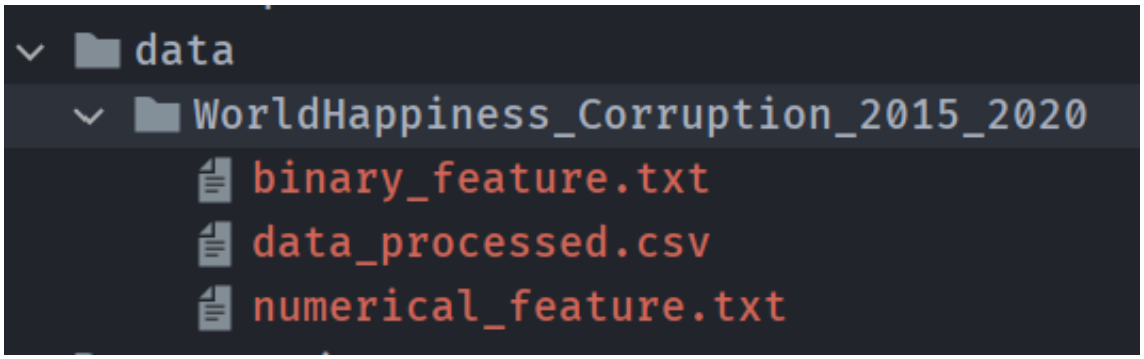

图 5: 数据结构目录

其中 data_processed.csv 是数据源文件，binary_feature.txt 记录了单元格数字类型为 bool 型的列名，numerical_feature.txt 记录了单元格类型为数值型的列名，其它的列都将被视为是类别型，并且我们需要经预测列的列名改为”target_label” 方便后续的数据处理。

主要修改了模型使用的损失函数以及最后的线性层处理，但是经过处理后模型的效果比较差，所以还需要进一步的学习修改。

5 实验结果分析

下面的表 1 描述了再公共数据集上 Transtab 和 XGBoost 的监督学习结果 arouc 指标对比，其中 arouc 值越大越好，所有的监督学习我们都没有执行预训练。我们可以可以看到除了在数据集 credit-approval 上 transtab 的表现不如 xgboost，其它数据集上都比 xgboost 表现的性能更好。

Method	credit-g	dresses-sales	adult	credit-approval
XGBoost	0.7236	0.5873	0.9012	**0.8973**
Transtab	**0.7604**	**0.6215**	**0.9019**	0.8665

表 1: 公共数据集上 Transtab 和 XGBoost 的监督学习结果 arouc 指标对比

下面的表 2 描述了再公共数据集上 Transtab 和 XGBoost 的迁移学习结果 arouc 指标对比。实验中迁移学习使用的方式是将一个数据集分为重合度为 50% 的两个数据集 s1 和 s2，在 s1 上预训练，在 s2 上进行微调，然后再 s2 上测试性能。我们同样可以看到除了 adult 数据集，transtab 的表现都比 xgboost 好。

Method	credit-g	dresses-sales	adult	credit-approval
XGBoost	0.7248	0.4635	**0.8868**	0.8543
Transtab	**0.7431**	**0.5549**	0.8834	**0.8737**

表 2: 公共数据集上 Transtab 和 XGBoost 的迁移学习结果 arouc 指标对比

6 总结与展望

本文提出了接受可变列输入的 TransTab。通过提出的垂直分区对比学习，它可以在多个表格数据集上进行预训练。我们设想它将成为表格基础模型的基础，并在未来广泛应用于表格相关的应用。

当然模型也存在一定的不足，transtab 还存在数据隐私问题，当使用文本序列表示特征时，可能存在一定的隐私泄露问题。此外，TransTab 需要比机器学习模型占用更多的资源。主要是因为在多头注意模块中使用了全注意机制，以及在特征化过程中的标记化。前一个问题可以通过用基于 mlp 的的块 (如门控注意力单元) 替换 transformer 来解决。后者可以在模型训练之前通过预标记化来避免。

参考文献

[1] WANG R, FU B, FU G, et al. Deep & cross network for ad click predictions[G]//Proceedings of the ADKDD'17. 2017: 1-7.

[2] YOON J, ZHANG Y, JORDON J, et al. Vime: Extending the success of self-and semi-supervised learning to tabular domain[J]. Advances in Neural Information Processing Systems, 2020, 33: 11033-11043.

[3] ZHANG Y, TONG J, WANG Z, et al. Customer transaction fraud detection using xgboost model[C]// 2020 International Conference on Computer Engineering and Application (ICCEA). 2020: 554-558.

[4] WANG Z, LI S. Data-driven risk assessment on urban pipeline network based on a cluster model[J]. Reliability Engineering & System Safety, 2020, 196: 106781.

[5] NARGESIAN F, ZHU E, PU K Q, et al. Table union search on open data[J]. Proceedings of the VLDB Endowment, 2018, 11(7): 813-825.

[6] HE K, CHEN X, XIE S, et al. Masked autoencoders are scalable vision learners[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022: 16000-16009.

[7] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[8] DEVLIN J, CHANG M W, LEE K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.

[9] SCHUSTER M, NAKAJIMA K. Japanese and korean voice search[C]//2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). 2012: 5149-5152.

[10] YUCHEN LIN B, LEE S, KHANNA R, et al. Birds have four legs?! NumerSense: Probing Numerical Commonsense Knowledge of Pre-trained Language Models[J]. arXiv e-prints, 2020: arXiv-2005.

[11] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[12] DARABI S, FAZELI S, PAZOKI A, et al. Contrastive Mixup: Self-and Semi-Supervised learning for Tabular Domain[J]. arXiv preprint arXiv:2108.12296, 2021.

[13] BAHRI D, JIANG H, TAY Y, et al. Scarf: Self-supervised contrastive learning using random feature corruption[J]. arXiv preprint arXiv:2106.15147, 2021.

[14] STONEBRAKER M, ABADI D J, BATKIN A, et al. C-store: a column-oriented DBMS[G]//Making Databases Work: the Pragmatic Wisdom of Michael Stonebraker. 2018: 491-518.

[15] KHOSLA P, TETERWAK P, WANG C, et al. Supervised contrastive learning[J]. Advances in Neural Information Processing Systems, 2020, 33: 18661-18673.

[16] KINGMA D P, BA J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.