A REPRODUCIBILITY STUDY OF TRANSTAB: LEARNING TRANSFERABLE TABULAR TRANSFORMERS ACROSS TABLES

Alberto Tamajo, Jakub Dylag, Alessandro Nerla & Laurin Lanz
Department of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ, United Kingdom
{at2n19,jd8g19,an1g19,lll1u19}@soton.ac.uk

ABSTRACT

The ubiquity of tabular data in machine learning led Wang & Sun (2022) to introduce a versatile tabular learning framework, Transferable Tabular Transformer (TransTab), capable of modelling variable-column tables. Furthermore, they proposed a novel technique that enables supervised or self-supervised pretraining on multiple tables, as well as finetuning on the target dataset. Given the potential impact of their work, we aim to verify their claims by trying to reproduce their results. Specifically, we try to corroborate the *'methods'* and *'results'* reproducibility of their paper.

1 INTRODUCTION

Despite the ubiquitousness of tabular data in machine learning, recent works on tabular data modelling can only handle supervised learning or pretraining on the same-structure tables. This limitation originates from these works' assumptions of fixed-structure tabular data. To address this limitation, Wang & Sun (2022) propose Transtab, a Transferable Tabular Transformer for variable-column tables. This tabular learning architecture tokenises and embeds each row in a table by combining each cell with its corresponding column description. The token-level embeddings are then further encoded by applying a stack of gated transformers. Finally, the feature vectors are fed into a *classifier* for label classification or a *projector* for contrastive learning. Wang & Sun (2022) also introduce a novel contrastive learning methodology named *VPCL* that enables both self-supervised and supervised pretraining on multiple tables.

Transtab's flexible architecture enables it to be utilised in the following application scenarios: (1) supervised learning, (2) feature incremental learning, (3) transfer learning, (4) zero-shot learning and (5) supervised and self-supervised pretraining. The authors conduct extensive experimentation for each of the above application scenarios, comparing their proposal with multiple baseline methods on diverse benchmark datasets.

The main aim of this report is to investigate the *'methods'* and *'results'* reproducibility (Goodman et al., 2016) of those results. *'Methods'* reproducibility is verified by thoroughly inspecting the author's code with the intent of reporting any substantial information not included in the paper and detecting discrepancies between the code implementation and the paper's functional descriptions. Given limited resources, strict time constraints and the extensive number of experiments, *'results'* reproducibility is verified by leveraging Transtab's code repository, implemented in PyTorch and publicly available at `https://github.com/RyanWangZf/transtab`. This allows us to save time on implementation and run a very extensive number of experiments, verifying the paper's results fairly.

2 METHODS REPRODUCIBILITY

We extensively inspected Transtab's code by means of PyCharm Professional and its integrated debugger. This facilitated code inspection within a controlled environment, in order to thoroughly investigate the current state of the program step by step. Thus, allowing us to approach the authors'

code with critical thinking and verify that it is consistent with their paper. Designing and running unit tests on the authors' code is unnecessary as a result of its implementation in PyTorch, which enables thorough reproducible code inspection using a debugger.

2.1 DISCREPANCIES

We discovered the following discrepancies between the authors' implementation and their paper.

Firstly, the paper states that numerical embeddings are yielded as $\mathbf{E}_u = x_u \times \mathbf{E}_{u,col}$, where x_u is a numerical feature and $\mathbf{E}_{u,col}$ is a column description embedding. However, numerical embeddings are yielded as $\mathbf{E}_u = x_u \times \mathbf{E}_{u,col} + b$ in the implementation, where b is a bias term. Also, unlike in the paper, the numerical embeddings $\mathbf{E}_u$ do not pass the layer normalisation (Ba et al., 2016).

Secondly, gated transformers are not implemented as described in the paper. Note that there is even a discrepancy between Equation (3) and Figure (2) in the original paper. The code implementation is more similar to Figure (2). Here, we rewrite Equation (3) so that it reflects the actual implementation

$$\mathbf{Z}^{l+1} = \text{LayerNorm}\left(\text{Linear}\left(\mathbf{L}\right) + \mathbf{Z}_{att}^{l+1}\right)$$

where $\mathbf{Z}_{att}^{l+1} = \text{LayerNorm}\left(\mathbf{Z}_{att}^{l} + \mathbf{Z}^{l}\right)$ and $\mathbf{L} = \text{ReLU}\left(\mathbf{g}^{l}\left(\mathbf{Z}_{att}^{l+1}\right) \odot \text{Linear}\left(\mathbf{Z}_{att}^{l+1}\right)\right)$

Finally, unlike in the paper, the [cls] embedding $\mathbf{z}^{[cls]}$ passes first a layer normalisation rather than being directly fed as input to the classifier.

2.2 MISSING INFORMATION

The original paper fails to mention crucial information about the proposed tabular framework. In this section, we fill these gaps by leveraging Transtab's code package.

The authors use the BERT base model (uncased) (Devlin et al., 2018) for tokenisation with a vocabulary size of 512. Different weight initialisation techniques are used across the framework. The Kaiman initialisation (He et al., 2015) is used for the word embeddings. On the contrary, the numerical and [cls] embeddings are initialised according to the uniform distribution $\mathcal{U}(-\frac{1}{128}, \frac{1}{128})$, where 128 is the dimension of the embeddings. The remaining weights are initialised using PyTorch's default initialisation methods.

2.3 OUTCOME

In light of sections 2.1 and 2.2, we argue that the original paper is not *'methods'* reproducible as it fails to provide crucial details, and there are substantial discrepancies between the content of the paper and the actual implementation.

3 RESULTS REPRODUCIBILITY

Wang & Sun (2022) conducted extensive experiments for each of Transtab's supported application scenarios, comparing their proposal with multiple baseline methods on diverse benchmark datasets. In this section, we aim to replicate the results of those experiments.

The original study leverages five clinical trial datasets and eight public tabular datasets. Due to access limitations, we were unable to use the sensitive clinical datasets and resorted to only sourcing the public ones. Note that the *'insurance-co'* dataset was unavailable, limiting our study to seven datasets.

We stuck to the dataset pre-processing protocols and model architecture outlined in the paper. Direct reimplementation of the training protocols, however, was not possible due to insufficient details. The ten random seeds used are omitted. Moreover, it is unclear which learning rate in {2e-5, 5e-5, 1e-4} and batch size in {16, 64, 128} were utilised for each experiment. In light of this absence of information, we sampled ten random seeds from a uniform distribution $\mathcal{U}(0, 1000)$, as well as computed results for all possible combinations of the learning rate, batch size and random seed. Similarly to the paper, we then averaged the test AUROC scores for each random seed against all combinations of batch size and learning rate and chose the best result as the outcome of each experiment. Despite significant computational expenditure, this approach allows us to comprehensively verify results in a fair manner. Thus, it enables us to state whether the original experiments are *'results'* reproducible.

In total, we ran an extensive number of 6,840 comprehensive training procedures, over a period spanning four weeks. We leveraged a Google Colab Pro subscription, using either an Nvidia V100 or T4 GPU. The Python scripts used and the entire range of experiments' results are publicly available at `https://github.com/COMP6258-Reproducibility-Challenge/TransTab-Reproducibility`. It is vital to note that the experiments' implementations are not publicly available; thus, we had to re-implement them leveraging Transtab's code package. For the experiments in sections 3.2, 3.3 and 3.4, although we used the same random seed as in the original paper, we cannot guarantee that the splits are identical to those in the original experiments because we are very likely to have used a different implementation procedure.

In the following tables, we abbreviate the dataset names as in the original paper. For each result, we state the learning rate and batch size yielding maximum AUROC across all seeds. We recompute Transtab's rank, as a comparison measure with respect to the baseline methods, for each dataset and application scenario. The overall rank is Transtab's performance rank for a given application scenario across all datasets compared to the multiple baseline methods used in the original paper. Green and red values indicate difference between our metrics and those in the original paper.

3.1 SUPERVISED LEARNING

The original paper conducted several experiments to verify how Transtab compares with baselines under the vanilla supervised setting. Although our results, shown in Table 1, are comparable to the original, we observe reduced performance, with Transtab ranking fifth as opposed to third originally amongst baseline methods.

Methods	CG	CA	DS	AD	CB	BL	IC	Avg. rank(Std)	Overall rank
Original	**0.768**	0.881	**0.643**	**0.907**	**0.851**	**0.845**	**0.919**	**3.42** (2.69)	**3**
Ours	0.738 (0.007)	**0.906** (0.009)	0.628 (0.024)	0.904 (0.000)	0.814 (0.032)	0.835 (0.002)	0.913 (0.001)		
Learning rate	2E-05	5E-05	1E-04	2E-05	1E-04	1E-04	1E-04	+1.43 (-0.58)	+2
Batch size	128	64	64	128	16	16	64		
Our rank	3 (+2)	1 (-1)	6 (±0)	10 (+2)	5 (+1)	5 (+4)	5 (+3)	4.85 (2.11)	5

Table 1: Comparison between our best supervised learning test AUROC scores and those in the original paper. Standard deviations are in parenthesis.

3.2 FEATURE INCREMENTAL LEARNING

Transtab's paper demonstrates that TransTab makes the best of incremental features to learn better, outperforming the baseline methods on the public datasets by a significant margin. Our experiments, shown in Table 2, support these findings, although our maximum AUROC scores are negligibly lower.

Methods	CG	CA	DS	AD	CB	BL	IC	Avg. rank(Std)	Overall rank
Original	**0.741**	0.879	**0.665**	0.894	0.791	**0.841**	0.897	**1.14** (0.37)	1
Ours	0.733 (0.014)	**0.896** (0.007)	0.605 (0.025)	**0.904** (0.000)	**0.822** (0.033)	0.832 (0.004)	**0.912** (0.001)		
Learning rate	5E-05	2E-05	5E-05	5E-05	1E-04	5E-05	2E-05	+0.57 (+1.12)	±0
Batch size	16	128	128	64	16	16	16		
Our rank	1 (±0)	1 (±0)	2 (+1)	1 (±0)	1 (±0)	5 (+4)	1 (-1)	1.71 (1.49)	1

Table 2: Comparison between our best feature incremental learning test AUROC scores and those in (Wang & Sun, 2022). Standard deviations are in parenthesis.

3.3 TRANSFER LEARNING

Our results in Table 3 corroborate Transtab's ability to benefit from knowledge transfer across tables to boost its performance. Note that although our results are considerably inferior, 2.78 average rank against 1.50 in the original paper, Transtab still ranks first overall in our experiments, displaying that the baseline methods lag significantly behind.

3.4 ZERO-SHOT LEARNING

Although the original paper indicates zero-shot learning results superior to supervised learning, our results in Table 4 do not confirm this statement. We observe inconsistent knowledge retainment from sets 1 and 2 when predicting set 3, with consistently lesser performance than supervised learning and the paper's published results. On the other hand, our results reinforce the higher average performance of transfer learning when compared to supervised and zero-shot learning.

Methods	CG		CA		DS		AD		CB		BL		IC		Avg. rank	Overall rank
	set 1	set 2	set 1	set 2	set 1	set 2	set 1	set 2	set 1	set 2	set 1	set 2	set 1	set 2		
Original	**0.74**	**0.76**	**0.87**	0.89	0.55	**0.66**	0.88	**0.90**	0.80	**0.80**	0.79	**0.84**	**0.91**	0.91	1.50 (0.94)	1
Ours	0.70 (0.013)	0.72 (0.008)	0.83 (0.01)	**0.92** (0.005)	**0.58** (0.033)	0.62 (0.014)	**0.89** (0.001)	0.89 (0.00)	**0.81** (0.037)	0.78 (0.020)	**0.81** (0.004)	0.83 (0.003)	0.90 (0.001)	**0.91** (0.000)		
Learning rate	5E-05	1E-04	1E-04	5E-05	5E-05	5E-05	2E-05	5E-05	1E-04	5E-05	1E-04	5E-05	2E-05	2E-05	+1.28 (+1.17)	±0
Batch size	16	64	16	64	128	64	16	128	16	16	16	16	64	128		
Our rank	4 (+3)	1 (±0)	4 (+3)	1 (±0)	1 (±0)	5 (+3)	1 (-2)	5 (+4)	1 (±0)	7 (+3)	1 (±0)	2 (+1)	5 (+3)	1 (±0)	2.78 (2.11)	1

Table 3: Comparison between our best transfer learning test AUROC scores and those in Transtab's paper. Standard deviations are in parenthesis.

Methods	CG	CA	DS	AD	CB	BL	IC
Original Supervised	0.581	0.635	**0.571**	**0.898**	0.733	**0.822**	**0.875**
Our Supervised	**0.706** (0.009)	**0.905** (0.004)	0.559 (0.014)	0.731 (0.001)	**0.766** (0.014)	0.797 (0.001)	0.871 (0.001)
Learning rate	2E-05	1E-04	1E-04	2E-05	1E-04	5E-05	1E-04
Batch Size	128	16	128	128	16	16	16
Original Transfer	**0.719**	0.758	0.561	**0.900**	**0.854**	**0.831**	**0.880**
Our Transfer	0.712 (0.01)	**0.903** (0.008)	**0.566** (0.024)	0.731 (0.001)	0.764 (0.020)	0.799 (0.001)	0.870 (0.002)
Learning rate	5E-05	1E-04	2E-05	1E-04	1E-04	2E-05	1E-04
Batch Size	128	16	16	16	16	64	64
Original Zero-shot	**0.685**	**0.721**	**0.538**	**0.892**	**0.710**	**0.804**	0.**874**
Our Zero-shot	0.538 (0.059)	0.565 (0.197)	0.509 (0.054)	0.509 (0.061)	0.479 (0.064)	0.612 (0.139)	0.565 (0.172)
LR	5E-05	2E-05	1E-04	1E-04	5E-05	2E-05	1E-04
Batch Size	128	128	16	16	64	64	16

Table 4: Comparison between our best zero-shot learning test AUROC scores and those in the original paper. Standard deviations are in parenthesis.

4 Conclusion & Future Work

In this paper, we investigated the *'methods'* and *'results'* reproducibility of Transtab (Wang & Sun, 2022). We showed that the content of the original paper differs from the actual implementation. Furthermore, critical details are omitted. Consequently, the original work is not *'methods'* reproducible as it is impossible to reproduce its results based solely on the paper. Furthermore, we verified Transtab's *'results'* reproducibility by leveraging its publicly available code package. Although we observe minorly inferior test AUROC scores, our experiments indicate that Transtab retains its supremacy among the baselines across the public datasets in both the feature incremental learning and transfer learning application scenarios. In contrast, Transtab performs comparatively worse in our supervised learning experiments, ranked fifth among the baseline methods rather than third as in the original paper. Our zero-shot learning results also do not support the original paper's claim that Transtab retains the knowledge learned from a given dataset for predicting on a different dataset without further training. We set as future work the completion of the supervised and self-supervised pretraining experiments, which we did not fully terminate due to their high time complexity. Furthermore, we also deem it crucial to reproduce the baseline methods results to provide a faithful comparison between their and Transtab's performance.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Steven N. Goodman, Daniele Fanelli, and John P. A. Ioannidis. What does research reproducibility mean? *Science Translational Medicine*, 8(341):341ps12–341ps12, 2016. doi: 10.1126/scitranslmed.aaf5027. URL `https://www.science.org/doi/abs/10.1126/scitranslmed.aaf5027`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Zifeng Wang and Jimeng Sun. Transtab: Learning transferable tabular transformers across tables. *arXiv preprint arXiv:2205.09328*, 2022.