Towards Tabular Foundation Models

Maximilian Schambach

Towards Tabular Foundation Models

Status quo, challenges, and opportunities

Towards Tabular Foundation Models

Status Quo, Challenges, and Opportunities

Maximilian Schambach*
Merantix Momentum
2024/01/15

1 Introduction

Undoubtedly, deep learning has been the critical driver of many technological advances in recent years. This progress has been particularly notable in the areas of natural language processing and computer vision where deep learning approaches have long outperformed conventional engineering and machine learning paradigms. At the forefront of the current development stand so-called Foundation Models – large general-purpose architectures trained on a vast amount of data that show unprecedented performance across a variety of downstream tasks (Bommasani et al. 2021). For example, in natural language processing, Foundation Models have enabled breakthroughs in language understanding and generation. Most popularly, they are known through their usage in general-purpose chatbots like chatGPT, Bard, or HuggingChat, which are driven by OpenAI's GPT, Google's PaLM, or Meta's LLaMA language Foundation Models, respectively (Brown et al. 2020; Bubeck et al. 2023; Chowdhery et al. 2022; Touvron et al. 2023). In this instance, they are also typically referred to as Large Language Models (LLMs). Similarly, in computer vision, Foundation Models have become popular mostly in the context of generative artificial intelligence, for example through DALL-E, Stable Diffusion, or Midjourney (Ramesh et al. 2021; Rombach et al. 2022). However, they are also successfully used in other vision-related tasks such as classification, object detection, or semantic segmentation, to name a few. More recently, multi-modal Foundation Models have also been investigated, e.g. utilizing both natural languages and images (Radford et al. 2021; Bachmann et al. 2022; Girdhar et al. 2023; Anil et al. 2023).

While not yet fully understood, the success of Foundation Models is often attributed to one key ingredient: scale. Here, scale refers to both the number of parameters of the underlying deep learning model as well as the amount of training data needed (which are generally dependent on each other). The ability to scale deep learning models to hundreds of billions or even trillions of parameters (Du et al. 2021) – magnitudes previously far beyond feasibility – has been mostly facilitated by recent developments in deep learning architectures, training paradigms, and hardware. Namely, Foundation Models have been driven mostly by the use of the Transformer architecture, trained via self-supervision at scale using a large cluster of distributed computing nodes and hardware accelerators.

Besides a predictable increase in performance with scale, Foundation Models also exhibit so-called *emergent* properties, referring to characteristics of the trained model that appear indirectly through generalization rather than by explicit (architectural) construction. Often, but not always, these properties emerge suddenly at large scale and share similarities to phase transitions in physical systems. Examples of emergent properties in LLMs include, in particular, impressive zero- and few-shot inference across a variety of tasks such as word unscrambling or question answering in previously unseen languages (Wei et al. 2022). While generally, emergent properties are widely acknowledged, their underlying characteristics and working principles are actively – and sometimes controversially – discussed in the community (Wei et al. 2022; Schaeffer et al. 2022; Caron et al. 2021; Lu et al. 2023).

Despite their immense success in natural language processing and computer vision, Foundation

*maximilian.schambach@merantix.com
This work was funded by SAP SE.

Models have not yet been widely adapted to other domains (albeit being actively researched). A possible reason for this could be that current key innovators of Foundation Models are US-based tech giants such as Microsoft (mainly via OpenAI), Google, and Meta whose core products gravitate around natural language and vision.

Although tabular data is abundant in the wild and of great importance in many practical applications, showing great potential for AI applications (Chui et al. 2018), it has attracted only little attention in the context of deep learning and Foundation Models in particular. With the immense success of self-supervised learning and Foundation Models in the domain of natural language and computer vision, however, the field has recently gained more traction as shown by an increased number of publications as well as conference workshops at top machine learning conferences specifically dedicated to the tabular domain as depicted in Figure 1. Nevertheless, the technical challenges associated with designing and training a Tabular Foundation Model, i.e. a deep learning model trained via self-supervision using a large corpus of heterogeneous tabular datasets at scale, remain unaddressed in the literature.

To this end, the main contributions of this whitepaper are as follows:

- We provide an overview of the literature most relevant to the context of Tabular Foundation Models, discussing achievements as well as limitations of current approaches.
- We highlight the technical challenges associated with Tabular Foundation Models, in particular those distinguishing them from natural language or computer vision.
- We outline an architecture and training pipeline as a baseline towards a Tabular Foundation Model and show preliminary experimental results.

2 Tabular Data in Deep Learning

In the past years, the technical developments and research efforts within deep learning have been immense. While recurrent and convolutional neural networks have previously been predominant in deep learning in the context of natural language processing and computer vision, respectively, they are now mostly universally replaced by the Transformer architecture (Vaswani et al. 2017). In short, the Transformer architecture is a general-purpose, i.e. mostly modality-unspecific, architecture built upon the principle of self-attention. At its core, the Transformer processes a sequence of tokens, such as word or sub-word embeddings, by contextualizing them via attention. In natural language, the input tokens are obtained from word tokenizers such as WordPiece, SentencePiece, or Byte-Pair Encoding. Simply speaking, in the attention mechanism, the embedding of each token is compared with every other token of the sequence using some attention measure and weighted accordingly. The output of a plain Transformer encoder is thus a sequence of *contextualized* embeddings of the same length as the input. In order to obtain a single representation of the input, e.g. a sentence, the embeddings are aggregated, for example via max or mean pooling. Alternatively, in particular in the supervised learning case, a learnable CLS token is prepended to the input sequence. After its contextualization through the encoder, where it interacts with all tokens of the sequence, the CLS token is used for prediction. To this end, the contextualized CLS embedding needs to incorporate all predictive information of the input token sequence. Notably, the Transformer has shown excellent scaling behavior and unlocked unprecedented performance in natural language as well as computer vision. That is, unlike previous architectures, Transformers show deterministic scaling laws resulting in predictably better performance with increased scale (Kaplan et al. 2020; Dosovitskiy et al. 2021). The ability to scale the architecture in both its size as well as data is one key ingredient to Foundation Models.

In the context of tabular data, however, sound deep learning approaches and evaluations are scarce, and conventional methods, in particular gradient-boosted decision trees, are still considered state-of-the-art. Implementations such as XGBoost or LightGBM (Tianqi Chen and Guestrin 2016; Ke et al. 2017) enjoy high popularity in the community, and more recent developments such as CatBoost achieve state-of-the-art results across common benchmark datasets (Dorogush et al. 2017) due to a number of key challenges which we discuss in detail ins Section 3. In tabular deep learning, architectures utilizing Transformers have gained popularity but we consider them to be at very early stages as compared to developments in other fields. In particular, large Transformer-based tabular-specific architectures are yet to be investigated.

In the following, we discuss the relevant deep learning literature with a focus on tabular representation learning as they build the foundation of most tabular downstream tasks. Roughly speaking, representation learning deals with obtaining

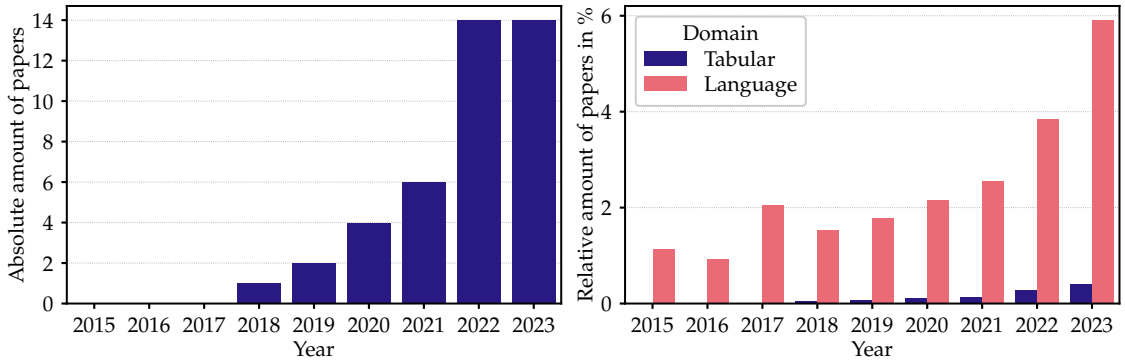

Figure 1: Approximate amount of (main-track) papers published at top ML conferences (NeurIPS, ICLR, and ICML).

high-dimensional vectors, i.e. representations, of the underlying data which describe the data completely within the required use. For example, the representation vector of a row can be used to predict the value of a target column. As previously argued, we believe scale to be the key to unlocking the potential towards a Tabular Foundation Model and thus highlight Transformer-based architectures as well as advances in tabular self-supervised learning with a focus on the cross-table learning case.

While the Transformer allows for a stable scaling in parameters, the amount of data needs to be scaled accordingly. Conventionally, deep learning models are trained in a supervised fashion requiring labeled training data. Arguably, however, it is infeasible to label training datasets of the orders of magnitude required to train Foundation Models. Hence, Foundation Models are mostly trained using self-supervision at scale. Generally, self-supervised learning refers to the principle of generating the pretraining task from the training data itself without explicit target labels (Balestriero et al. 2023). Arguably, scaling tabular data can only meaningfully be achieved when using a large heterogeneous tabular dataset as it is naturally limited in the single-table case. Hence, a tabular architecture towards a model considered foundational needs to be trained on and be able to generalize across multiple tables. However, in the tabular case, certain challenges arise in particular with respect to tokenization, cross-table generalization, and permutation invariance of the architecture, which we discuss in detail later. We do not discuss here the long history of machine learning literature for tabular data as well as early deep learning-based approaches as it would arguably go beyond the scope of this work. For a more general overview of deep learning approaches for tabular data, we refer to a recent survey on the topic (Borisov, Leemann, et al. 2022).

2.1 Single-table deep learning

Conventionally, machine learning approaches in the context of tabular data consider the single-table case, e.g. training and evaluating a classification or regression model on a single table – typically in a supervised fashion. In addition to the aforementioned boosting methods, several deep learning-based architectures have been developed in the past, ranging from simple Multi-Layer Perceptrons (MLPs) to more recent Transformer-based architectures.

Supervised learning A vast amount of work has been proposed in the context of supervised single-table deep learning. Notable highlights include early adaptations of the Transformer to tabular data, namely the TabTransformer architecture (X. Huang et al. 2020) as well as the more refined FT-Transformer (Gorishniy, Rubachev, Khrulkov, et al. 2021). Whereas the TabTransformer showed promising first results by applying the Transformer architecture to obtain contextualized tabular representations, its application is limited to categorical features. More precisely, only the categorical features are embedded in a contextualized fashion using a Transformer. The contextualized categorical embeddings are then concatenated with the (raw) numeric features and subsequently processed by an MLP for the final downstream task. From a technical point of view, this limitation arises naturally, as embeddings of categorical features are straightforward to obtain via look-up embeddings in full analogy to the case in natural language, where the vocabulary is given by the individual unique categories of all columns. Encoding and embedding numerical features, however, is not as straightforward and is therefore neglected in early works.

The FT-Transformer successfully refines this approach by proposing a feature tokenizer to encode and embed both categorical as well as numerical features into the same embedding space. Again,

categorical features are embedded using a simple lookup table, whereas numerical features are projected into the embedding space using a linear layer. To each feature, a column-specific bias embedding is added, effectively resulting in an additive column encoding to replace the positional encoding common in Transformers for language (we discuss this detail further in Section 3). The architecture is then trained using a supervised tabular downstream task estimated using an aggregated representation of each row (via a prepended learnable CLS token). In our opinion, the FT-Transformer is the first clean and simple Transformer architecture for unconstrained tabular data. In that sense, it serves as a good baseline for further investigations and generalizations. However, in the original setup, the FT-Transformer is applied to a supervised training task and is limited to learning representations of a single table only.

Self-supervised learning Particularly relevant to a Tabular Foundation Model are approaches utilizing self-supervised training. Here, a number of interesting works have emerged in recent years, albeit mostly limited to the single-table case. While early works utilize an autoencoder approach (Yoon et al. 2020), similar to early works in computer vision, they were soon either extended (Ucar et al. 2021; Darabi et al. 2021) or fully replaced by contrastive approaches (Bahri et al. 2022). In the contrastive setup, a loss is constructed with the objective of aligning the representations of rows that are considered similar while separating embeddings of dissimilar ones. Inspired by approaches successful in computer vision, namely early works such as SimCLR based on the InfoNCE objective (Oord et al. 2019; Ting Chen et al. 2020), positive pairs are constructed by augmenting a single sample, while different samples, i.e. rows, are considered dissimilar. For tabular data, augmentation can be performed via cell corruption, e.g. by replacing a certain amount of cell values with a random value drawn from the column's corresponding marginalized distribution as for example proposed in the notable approach SCARF (Bahri et al. 2022). That is, a certain percentage of values is replaced with non-contextualized but column-specific random values. Similarly, augmentation can also be performed in the latent space, as is done in the Contrastive Mixup method (Darabi et al. 2021), or in a combined fashion, augmenting samples as well as latent representations as performed by the SAINT approach (Somepalli et al. 2022). In all instances, the positive and negative samples are then passed through a shared backbone architecture to obtain the corresponding embeddings. The loss objective then enhances the similarity of the embeddings of positive samples while making the embeddings of negative pairs dissimilar.

Notably, in computer vision, various further developments of these approaches exist, for example, non-contrastive self-supervised objectives such as self-distillation techniques as introduced and popularized by BYOL and DINO (Grill et al. 2020; Caron et al. 2021) or correlation-based approaches such as BarlowTwins and VicReg (Zbontar et al. 2021; Bardes et al. 2022) which are pushing the state of the art in computer vision. However, to the best of our knowledge, similar non-contrastive approaches have not yet been investigated in the tabular context.

Finally, likely due to the increased use of the Transformer architecture, approaches analogous to Masked Language Modelling or Masked Autoencoding (Devlin et al. 2019; He et al. 2022), which are popular in the language and vision contexts, respectively, have been investigated in the tabular context. While they were first applied to a wide range of LLM-based tabular approaches, which we will discuss later, they have also successfully been applied to table-specific Transformer architectures such as MET (Majmundar et al. 2022) and generally found to be well-performing while more simplistic and easy to implement as compared to contrastive or distillation approaches (Rubachev et al. 2022). There exist two similar yet distinct popular approaches to masked autoencoding, namely Masked Cell Recovery and Corrupted Cell Recovery. Conceptually, Masked Cell Recovery is a straightforward approach where a certain amount of input cells is randomly masked by a specific learnable Mask token. In the case of Corrupted Cell Recovery, the tokens are instead replaced by noisy values, e.g. similar to the row augmentation in contrastive learning using samples drawn from the column's marginalized distribution. The embeddings are then processed by a Transformer backbone and the training objective is to predict the original values from the contextualized embeddings of the masked or corrupted cells. Masked Cell Recovery is a very natural learning objective in the tabular domain, in our opinion, as it corresponds to the problem of missing value imputation which often occurs in practice. This also implies that a model trained via Masked Cell Recovery can be used both for generating general-purpose representations as well as a generative imputation model. Being capable of extracting general-purpose representations after the self-supervised pretraining, the models are typically then finetuned in a supervised fashion on a specific downstream task, in full analogy to the use cases in language or vision. Again, further developments in those areas

– namely more efficient transfer learning ranging from adapters or prompt adaptation to quantization or low-rank approximation techniques (Houlsby et al. 2019; Hu et al. 2022; P. Gao et al. 2023; Dettmers et al. 2023) – have not yet been investigated in the tabular domain to the best of our knowledge. However, when using a Transformer backbone, these approaches should trivially transfer to the tabular domain.

While many works in the context of single-table representation learning exist, it should be noted that they do not yet significantly or consistently outperform conventional methods. In fact, depending on the benchmark used, it is argued that tree-based models still outperform current deep learning architectures across the board (Shwartz-Ziv and Armon 2021; Grinsztajn et al. 2022). We do not believe in the generality of these findings and discuss the details and importance of benchmark data in Section 3. For example, the findings do not take the scaling potential of Transformers architectures into account and often make unequal comparisons using a fixed compute budget. While this is a valid point of concern in the single-table scenario, the additional costs of pretraining should pay off in a general-purpose cross-table Foundation Model.

2.2 Cross-table deep learning

In order to enable large-scale pretraining of tabular models and thus unlock the full scaling potential of the Transformer architecture, cross-table pretraining is indispensable. However, only very few table-specific architectures exist and the majority of works focus on utilizing LLM for tabular data – an approach that we believe to be fundamentally limited for technical reasons which we discuss in detail in Section 3.

Supervised learning To the best of our knowledge, there exist only two supervised cross-table learning approaches: the TransTab as well as the notable TabPFN architecture (Z. Wang and Sun 2022; Hollmann et al. 2023).

TransTab proposes a tokenization approach that generalizes across tables by incorporating explicit column semantics via learnable column name word embeddings. Strictly speaking, TransTab is not solely trained in a supervised fashion but utilizes both a supervised loss and a novel contrastive self-supervised objective. While containing interesting novel ideas around table tokenization, the experimental evaluation of the paper focuses on cross-table learning for comparably small tables with partially overlapping columns. Moreover, they do not perform many ablation studies on the architectural and training choices made, in particular with respect to the proposed tokenizer. In detail, it remains unclear how tokenization and concatenation are performed across multiple tables that each contain a different amount of columns and how they are interleaved during training.

Notably, the recent TabPFN architecture follows a fundamentally different approach to the previously mentioned ones: the downstream task is formulated via approximate Bayesian inference conditioned on the training dataset. The network learns a synthetic cross-table data prior and inference is performed by passing the (small) training set as well as the test dataset to the architecture in a single forward pass. Practically, TabPFN can solve small classification tasks fully at inference rather than finetuning the backbone and is shown to outperform conventional baselines as well as AutoML approaches in that regime. However, the approach is currently limited to classification tasks and comparably small tables with only numerical features without missing values. While being conceptually interesting and quite useful for practitioners in the small-data regime, it is unclear how the approach could generalize and scale towards something considered a Foundation Model. Nevertheless, further refinements and novel approaches in this context are very valuable and we follow the continued research in this direction with great interest.

Although self-supervised learning will likely be a cornerstone in building Tabular Foundation Models, it remains under-researched and underdeveloped in this domain. While there exist some works considering self-supervised learning for tables with only partially overlapping columns (Levin et al. 2023; Onishi et al. 2023), we are only aware of one recent approach that investigates unconstrained cross-table pretraining – namely the XTab architecture (Zhu et al. 2023). In addition, we are aware of one technical preprint presenting the UniTabE framework (Y. Yang et al. 2023), which, in our opinion, is lacking in technical details to fully judge its potential. XTab generalizes the previously discussed FT-Transformer to the cross-table setting by using table-specific encoders with a Transformer backbone that is shared across all tables. We believe this to be the right first step towards generalizable architectures. On the technical side, the pretraining is realized in a federated learning setup, where the table-specific tokenizers are trained on separate GPUs together with the shared backbone. The authors investigate self-supervised recovery losses via Corrupted Cell Recovery, as well as contrastive approaches. They find that, generally, contrastive approaches perform

worse in this case, which is in line with our expectation as contrastive learning is often considered technically challenging and less stable. The federated approach has the advantage of simplicity as each tokenizer and classifier head is restricted to a single table and each GPU node only processes batches of samples of a single table. The disadvantages, in our opinion, are that the scaling of the used hardware is directly coupled to the number of datasets and cannot be independently controlled. Furthermore, we believe the hardware utilization to be suboptimal as the GPUs of simple-to-encode datasets will idle due to the gradient synchronization lock waiting for the slowest-to-encode datasets and GPUs.

While the authors show that pretraining is beneficial in most cases, the detailed performance behavior in different tabular regimes remains an open question. Furthermore, only a comparably small architecture is being investigated with a backbone model size of roughly 740 k parameters which we believe to be insufficient. For comparison, even early-stage language models such as GPT-1 and GPT-2 had a two, respectively four, orders of magnitude higher backbone parameter count. Hence, the potential of cross-table pretraining in order to scale the shared backbone is not made use of. The overall results thus show only decent performance, but again fall behind gradient-boosted trees and CatBoost in particular.

Large Language Model-based methods Despite not being the key to success towards a Tabular Foundation Model in our opinion, we discuss the most relevant recent approaches utilizing pretrained LLM for tabular data here for completeness. For example, they have shown interesting performance for simple tabular data cleaning or integration tasks (Narayan et al. 2022).

Many works for cross-table architectures are built upon pretrained LLMs – in particular the BERT architecture, as used by early works such as the TAPAS table question-answering approach (Herzig et al. 2020). To be able to use the pretrained text tokenizers, some form of table serialization is necessary, the details and shortcomings of which we discuss in Section 3. Notably, TAPAS introduces several encodings in addition to the conventional positional encoding used in natural language: namely column encoding as well as row and rank embeddings to be used for ordinal values. A shortcoming of TAPAS is the lack of column and row order permutation invariance, which is addressed in TableFormer and can be considered its refinement (J. Yang et al. 2022).

Notably, unlike the aforementioned cross-table approaches, masked autoencoding is naturally a common self-supervised learning objective in the context of language models and contrastive approaches are rarely, if at all, considered. Further notable developments include TAPEX, which is trained as a neural SQL executor (Liu et al. 2022), TaBERT, a BERT-style model for joint text and tabular embeddings (Yin et al. 2020), and GReaT, an autoregressive GPT-style table generation architecture (Borisov, Seßler, et al. 2023).

Understanding how tabular data and natural language can be jointly processed is of great practical importance and LLMs might seem like an appealing option to this end. However, we believe that a good understanding of table-specific representation backbones is the root of further investigations. Cross-table representations could then be jointly integrated with natural language, for example via CLIP-style contrastive approaches, which are popular in the context of multimodal language and vision representation learning (Radford et al. 2021; Girdhar et al. 2023), aligning representations in a shared latent space as opposed to fully integrating them into the language model (Zha et al. 2023).

Moreover, it should be noted that a recent preprint proposes an architecture, dubbed AnyPredict, built upon several LLMs, which the authors claim to serve as a Tabular Foundation Model (Z. Wang, C. Gao, et al. 2023). While containing interesting ideas, for example around resolving the issue of limited training data which they extend by reformulating tables into natural language using chatGPT and a custom-trained generative language model, the overall architecture and presented evaluation do not qualify for it being a Foundation Model. In particular, the evaluation context is limited to tables within a fixed domain (namely, health data of specific medical trials). A broader investigation across different data domains would be necessary for further evaluation. Moreover, the approach, while interesting, seems overly convoluted and is unlikely to scale to larger and more complex datasets. In our opinion, recasting tables as natural language using pretrained LLMs seems like an intermediate solution as opposed to developing truly tabular Foundation Models.

In a similar fashion, the recently proposed TabFM framework (H. Zhang et al. 2023) utilizes an LLM-based generative approach and a unified (yet complex) tabular text representation method. While containing interesting ideas and results, TabFM, and in particular its unified text representation approach, heavily relies on human or LLM-based annotations used for its table serialization which adds computational complexity. Furthermore, it does not address the challenges around number tokenization

– in particular, the proposed unified approach is extremely token inefficient – which we discuss in more detail in Section 3.

2.3 A note on relational tables

The previously discussed approaches tackle problems associated with representation learning of single or multiple unrelated tables. Often, however, multiple tables share a relation in practice – most prominently in relational databases. While being an interesting area for further research, we believe that a well-performing cross-table representation learning framework, i.e. a Tabular Foundation Model, constitutes a good starting foundation for further inquiry of relational structures. For example, relational information could be injected using graph embeddings of the relational graph via cross-attention. While multiple approaches to representation learning for knowledge graphs exist (S. Chen et al. 2021; Bi et al. 2022; Diao and Loynd 2023), works investigating the incorporation of additional modalities have only recently been investigated. For example, in the natural language domain, incorporating knowledge graphs with language is an active area of research (Pan et al. 2023; Ioannidis et al. 2023).

In the context of relational databases, only a few dedicated works exist (Arora et al. 2021; Deng et al. 2020; Gorishniy, Rubachev, and Babenko 2022) with a noteworthy mention towards developing Foundation Models for databases that were recently outlined in a vision paper (Vogel et al. 2022). Notably, the outlined architecture at its core is built upon a table representation backbone, which in this particular case is based on a BART Large Language Model.

3 Towards a Tabular Foundation Model

3.1 Opportunities

In the age of AI, business data has become a precious asset for enterprises to optimise and automate their business processes. Still, enterprises fail to fully harness the power contained in this wealth of data, and until very recently have only been able to use it for narrow, company-specific use cases without benefitting from synergies deriving from the deep business and process knowledge inherent in cross-enterprise data assets. This is particularly true for structured data such as tables, time series, and databases, where the recent AI breakthroughs seen in the domain of text and images have not yet been replicated. With a large portion of business data being of this form, the wheel needs to be invented again and again, with each new use case requiring considerable manual effort, large amounts of data, and deep domain knowledge to be implemented. As a result, onboarding and roll-out of new AI solutions is slow and expensive, requiring a large investment in terms of time and resources. Consequently, enterprises have been missing out on opportunities to turn their business data into valuable insights. As highlighted in a 2018 discussion paper by the McKinsey Global Institute, structured data shows the largest potential of AI value impact across several domains and a multitude of use cases (Chui et al. 2018).

A Tabular Foundation Model can provide enterprises with a unique set of AI predictive capabilities on their structured business data, such as predicting invoice payment dates and supplier delivery quality or proposing efficiency improvements to a business process. It offers low-to-no-effort onboarding, high out-of-the-box accuracy, and the ability to scale out to completely new embedded AI scenarios. Furthermore, Tabular Foundation Models could be used as efficient multivariate tabular imputation systems or outperform conventional approaches in challenging use cases such as in the very small and very large data regimes, effectively replacing manual feature engineering. Finally, the integration of tabular understanding and LLMs could offer unprecedented performance in tabular reasoning tasks such as financial reasoning, tabular sentiment analysis, and more.

3.2 Status Quo

As previously outlined, there does not yet exist an architecture that can be considered a Tabular Foundation Model in our opinion. We believe the currently most promising approach is the recently proposed XTab architecture as previously discussed (Zhu et al. 2023). However, in order to understand the potential of a Transformer-based cross-table representation learning architecture, meticulous experimentation is required. For example, to the best of our knowledge, there are currently no works investigating the scaling behavior of Transformer-based tabular representation models, despite scale being the core driver of recent progress in natural language and vision. At the same time, the architectures proposed in the literature are limited to a relatively low parameter count, orders of magnitude smaller than even early language models such as GPT-2, as previously highlighted.

While a vast amount of Large Language Model-

based approaches exist, in particular, based on BERT- or GPT-style pretrained models, we believe them to be ill-suited for tabular data and of mere temporary interest spurred by the significant advances in natural language processing. Fundamentally, tabular-specific architectures are required due to the technical challenges involved, which we discuss next.

3.3 Challenges

Tokenization In our opinion, the main technical challenge associated with cross-table pretraining is the process of full-table or row tokenization, e.g. the encoding and embedding of the diverse data tables into a sequence of tokens to be processed by the Transformer backbone. In particular, each table contains a different number of unique numerical and/or categorical features with table-specific joint probability distributions, as well as dates, timestamps, or free text. In particular, even for features with names that would suggest shared semantics, such as identically named columns "Age", even the corresponding marginalized statistics can be vastly different across different tables. Hence, tabular tokenizers need to be either table-specific or be able to generalize across tables, e.g. by consuming meta information such as the table schema or estimated statistics in addition to the values to be tokenized. This is fundamentally different from natural language processing or computer vision, where the process of tokenization (or patch-wise embeddings in the case of images) naturally transfers to previously unseen data. We argue that understanding and developing suitable *cross-table tokenization* techniques is the key to successfully developing Tabular Foundation Models – not the particular backbone architecture or training objective itself.

One way to perform tabular tokenization that transfers across tables is to use tokenizers from LLMs in combination with serialization of the tabular data. This is the predominant approach in many self-supervised tabular learning architectures, some of which we outlined in Section 2. Different serialization techniques are outlined in more detail in a recent review paper (Badaro et al. 2023) and experimentally compared in the context of zero- and few-shot classification within the TabLLM approach (Hegselmann et al. 2023). However, we believe that there are numerous problems with this seemingly straightforward idea:

First, table serialization is not token-efficient. That is, every cell value is serialized into at least two, more commonly three or more, samples (e.g. "`The <Column> is <Value>`") which is further tokenized into a multitude of tokens, depending on the subword vocabulary of the specific text tokenizer used. Effectively, this results in a suboptimal scaling constant of the quadratic scaling behavior of the Transformer architecture with respect to the number of columns processed and drastically limits the number of columns to be encoded within the maximum context length of the backbone model.

Second, and more crucially, text tokenizers are not particularly well suited to encode numerical features. For example, conventional tokenizers tokenize numerical values into multiple tokens by splitting decimals at the decimal point and larger numbers into multiple subword tokens. Categorical values are typically directly treated as text which and split into multiple tokens depending on the used tokenizer and vocabulary. The relationship between these tokens associated with a single cell value then needs to be recovered by the backbone. While the effect of text tokenization on numerical feature representations has not been investigated in detail, recent works show that LLMs perform comparably poorly at tasks involving numerical understanding such as simple arithmetic tasks (Pi et al. 2022; Yuan et al. 2023) or financial reasoning (Wu et al. 2023; Li et al. 2023) – an observation likely rooted in unsuited tokenization and lack of specific training contexts. While workarounds, such as using character-level tokenization for numerical values, have been proposed and utilized (Pi et al. 2022; T. Zhang et al. 2023; Golkar et al. 2023), they do not resolve the core issue at hand and specific solutions have only recently emerged (Golkar et al. 2023). Furthermore, tokenizing cell values into multiple tokens makes an additional decoder architecture necessary, introducing needless technical overhead. In particular, how to best design a cross-table decoder is not straightforward and poses similar challenges to tokenization. However, also in the context of natural language processing, tokenization of numerical values has gained some interest. For example, two recent preprints explore first ideas of explicit number tokenization to patch pretrained LLMs (Han et al. 2023; Golkar et al. 2023). In addition, a recent survey discusses the problems and approaches associated with number tokenization in more detail (Thawani et al. 2021). However, note that current works do not take the tabular structure (e.g. the joint probability distributions of a collection of numeric features) into account and treat all numbers identically.

In analogy to tokenizers in natural language, tokenization is a nuanced, domain-specific problem that needs dedicated attention. Despite the immense amount of research efforts in natural lan-

guage processing, there exists no universal standard for text tokenization and the differences in behavior are not fully understood (Mielke et al. 2021). However, note that text tokenizers in natural language processing are pretrained on large text corpora and have contributed significantly to the increased performance of Transformer-based language models as they resolve many of the linguistic and engineering challenges involved. Similarly, tokenization in the tabular context requires dedicated efforts and meticulous experimentation in our opinion.

Permutation invariance Unlike text or images, tabular data does not inherently possess a particular ordering of the columns and rows. Hence, a tabular backbone architecture should provide representations that are invariant against column permutations in particular. Fittingly, the Transformer is naturally invariant against permutations of the input token sequence due to the bidirectional self-attention mechanism and the parallel processing of tokens. In LLMs, this invariance is explicitly broken by incorporating positional encodings. In tabular Transformer models, the positional encoding should be avoided and is replaced in some works by column encodings, i.e. a column-specific bias term added to the embeddings of a cell (Gorishniy, Rubachev, Khrulkov, et al. 2021; Zhu et al. 2023). Utilizing pretrained LLMs via row serialization is hence suffering from spurious spatial relations that are not inherent to the tabular data structure.

Notably, using tokenizers that split cell values into multiple tokens such as those used in LLMs, the desired invariance in the table input space does not naturally transfer to invariance in the token sequence, as the representations should not be invariant against permutations of tokens stemming from a single cell, nor the tokens indicating the corresponding column. Realizing permutation invariance with respect to the input table adds architectural complexity in this case and would require partial positional encodings, for example. For this reason, we believe that tokenizers that tokenize single cell values into a single token to be the most straightforward approach and that table schematics should be injected separately.

Further, the permutation invariance has consequences in the context of autoregressive pretraining strategies commonly used in generative language modeling (such as GPT-like models). For example, when using a table serialization approach, predicting the next column name is strictly an ill-defined problem. Great care should be taken to solely predict value-associated tokens in the autoregressive setup as opposed to tokens originating from the table schema. Some works use a permutation augmentation approach (Pietruszka et al. 2022; Borisov, Seßler, et al. 2023; T. Zhang et al. 2023) which, however, does not resolve the core issue at hand in our opinion. Despite generative models not being the focus of this paper, we believe that autoregressive generative modeling is ill-suited in the context of tabular data. In the tabular context, generative approaches such as Gibbs sampling from a model trained via masked autoencoding interpreted as a Markov random field (A. Wang and Cho 2019) or diffusion models (Ho et al. 2020; Kotelnikov et al. 2023) are likely more suited to the tabular structure in our opinion.

Data Finally, in order to systematically tackle the aforementioned challenges to develop cross-table representation learning approaches towards a Tabular Foundation Model, suitable high-quality data – training data as well as benchmarks – is of crucial importance. In order to be able to scale a corresponding model, a pretrained corpus of corresponding tabular data across various domains is required. As the actual scaling behavior has not yet been investigated, it is difficult to estimate the minimal amount needed. As a guiding principle, we transfer findings in the natural language processing community. For example, GPT-3 was trained on a corpus of 500 billion tokens from five different data sources of varying quality (Brown et al. 2020).

In addition to a large training corpus, a diverse benchmarking suite is necessary. While there exist a variety of different tabular benchmarks in the community, most notably the OpenML benchmark suites, such as the OpenML CC-18 classification benchmark as well as the recently introduced OpenML CRT23 regression benchmark (Bischl et al. 2021; Fischer et al. 2023), these benchmarks have not yet been thoroughly adopted in the community. The reasons for this may be many.

First, due to a lack of other tabular data sources, benchmark datasets are currently being partially used for both pretraining and evaluation, e.g. using a two-fold cross-validation as performed in the experimental evaluation of XTab (Zhu et al. 2023). This limits training data as well as benchmark capabilities. And even in this case, the amount of data is limited. For example, the whole OpenML CC-18 benchmark contains roughly 450 million individual tokens, i.e. cells, whereas CRT23 contains 12 million tokens in total. The biggest suite in the OpenML collection is a dedicated training holdout set with roughly 1.2 billion tokens in total. While this is enough to perform first scaling investigations, more data is very likely needed to scale architectures

towards a Tabular Foundation Model. Moreover, the OpenML training holdout set contains solely classification tasks which likely hinders the model from generalizing to strongly different downstream applications. A recently published large collection of tabular data assets, dubbed TabLib (Eggert et al. 2023), could be a promising contender. However, the dataset in its current raw, non-preprocessed state is likely too noisy to be used for large model training directly and requires engagement and experimentation of the community. Moreover, the license of the proposed dataset is unclear and usage is currently only recommended for research. It is unlikely that the dataset will be released under a commercially friendly license required for product development.

Secondly, the suitability of these benchmarks in the deep learning context remains an open question. For example, the OpenML benchmark suites contain mostly small- to mid-sized datasets that are often easy to tackle with established machine learning approaches. Extending the benchmarks by distinguishing between the difficulty of tasks or the width of the considered tables might yield additional insights into the difference in performance of conventional machine learning approaches versus deep learning ones. This is not to say that well-thought-out design choices are absent in these benchmarks. Notably, the OpenML benchmarks do not include tasks that are considered too easily solvable by a linear model. Nevertheless, the performance of simple linear models is often notably high (Borisov, Seßler, et al. 2023; McElfresh et al. 2023). It is unclear to what extent current benchmarks are truly suited to evaluate deep learning approaches.

Finally, the available downstream tasks are not very diverse. Currently, solid benchmarks only exist for classification and regression tasks. In comparison, in natural language processing, it is established practice to evaluate models across a multitude of benchmarks covering a wide range of downstream applications. Towards a Tabular Foundation Model, a more diverse set of downstream benchmark tasks, for example covering aspects of table question answering, missing value imputation, and others, is necessary.

4 Proposed approach

In order to systematically tackle the aforementioned challenges, a solid baseline architecture to serve as a starting point for further ablation studies and extensions is required. To this end, we propose a simple Transformer-based architecture for self-supervised cross-table pretraining that can be finetuned to arbitrary tabular downstream tasks. Our architecture is based on the FT-Transformer (Gorishniy, Rubachev, Khrulkov, et al. 2021) and conceptually similar to the XTab framework (Zhu et al. 2023) with some notable differences. As with XTab, the idea is that a shared backbone learns general relations between the tabular samples regardless of the table specifics. Using the proposed architecture, we investigate the scaling behavior in the single-table case, in which the architecture corresponds to a self-supervised variant of the FT-Transformer, and in the generalized cross-table case where we are able to scale the size of the backbone further. In the following, we will outline our proposed approach and results in a condensed form and refer to our technical preprint for more technical details as well as experimental evaluation (Schambach et al. 2023).

4.1 Architecture

An overview of our proposed architecture is given in Figure 2. Similar to the FT-Transformer and its cross-table generalization XTab, our approach uses table-specific tokenization and a cross-table shared plain Transformer backbone. Unlike XTab, however, we process samples from multiple tables by interleaving them in each batch and do not require a federated setup. This way, we are able to scale the used compute hardware independently of the amount of training dataset contained in the pretraining corpus and are able to instead scale the compute via common distributed training strategies.

That is, we perform stratified sampling from the pretraining corpus, such that samples from each dataset occur in the batch with the same probability. Each row is subsequently tokenized by a table-specific tokenizer. Unlike XTab, we do not process numerical features by quantile transformation but instead use a quantile encoder. That is, instead of performing the actual transform, we encode each feature using its corresponding quantile index. This has the advantage that numerical features are effectively treated as categoricals and all features can be embedded via look-up embeddings simplifying the implementation and its vectorization as well as the final projection layer used for prediction. In order to be able to stack the tokenized sequences across multiple datasets, we perform padding using a learnable Pad token, analogous to the case in natural language processing.

For self-supervised pretraining, we use a Masked Cell Recovery objective corresponding to a missing value imputation. That is, before padding, tokens in each sequence are randomly masked according to a specified masking fraction using a learnable

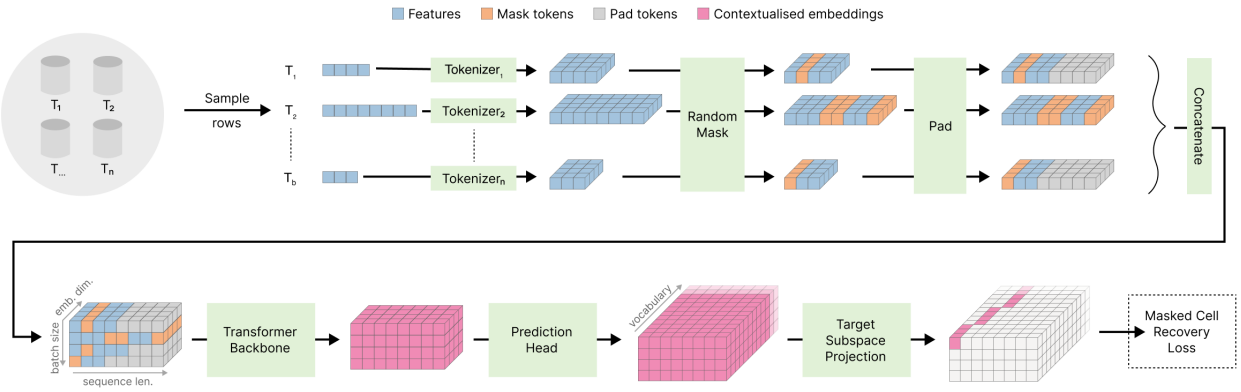

Figure 2: Overview of the proposed cross-table pretraining architecture.

Mask token. The training objective is then to predict the original value from the contextualized embeddings of the masked tokens. Masked recovery has been shown to be superior to both supervised as well as contrastive self-supervised approaches in the cross-table regime (Zhu et al. 2023) and is also a well-established learning objective in natural language where it originated. Having encoded all tabular features as categoricals effectively, the training objective reduces to a classification problem for all tokens which we optimize by minimizing the corresponding cross-entropy loss.

To minimize inductive biases, we do not use any additive learnable encodings, such as table or column-specific ones. As previously discussed, we also do not include positional encodings in order to retain the permutation invariance of the architecture. Implicitly, table and column characteristics need to be learned by the individual embedding layers. Investigating additional encodings is a worthwhile prospective experiment and could be beneficial in terms of an inductive bias in order to increase the pretraining efficiency but we deliberately do not include this in our baseline experiments.

4.2 Datasets and data pipeline

As previously discussed, sufficient pretraining data is required in order to scale the backbone architecture in size. Hence, we create a large curated heterogenous corpus of datasets from multiple OpenML benchmarks suites, including OpenML CRT, OpenML Training, and multiple regression benchmark suites. This corpus is used for the self-supervised pretraining of the architecture. We filter the datasets and keep only those with more than 1000 rows and between 10 to 50 columns. In total, our curated corpus consists of 76 datasets, including 30 binary and 26 multiclass classification as well as 20 regression task datasets of different sizes, summing to ca. 135 M tokens in total.

To stream from a large corpus of individual tabular datasets hosted on remote storage buckets in a multi-process fashion, we use the Squirrel library (Sohofi et al. 2022). The table-specific encoders are fitted by estimating the sample statistics over a subset with a fixed size before training. That is, in particular, the sample quantiles are estimated for the individual numerical features.

For evaluation, we use a small curated set of tabular tasks following a recent survey on the topic (Borisov, Leemann, et al. 2022), including the HELOC, Adult Income, California Housing, Cover Type, and HIGGS datasets. Together, these datasets cover different tasks, covering binary and multiclass classification as well as regression, and sizes, ranging from roughly 8 k samples to over 9 M rows. Nevertheless, it should be noted that these five benchmark datasets are well-established datasets for which conventional methods perform extraordinarily well.

4.3 Experimental details

We perform scaling experiments for both the single- as well the cross-table pretraining scenario. In the single-table case, we train a separate architecture on a holdout portion of each evaluation dataset via the self-supervised Masked Cell Recovery loss objective. We evaluate the performance of the obtained representations via linear probing using the respective table-specific downstream tasks. Here, linear probing corresponds to fitting a linear model on the contextualized representations, effectively measuring the linear separability of the representations with respect to the corresponding downstream task (which the backbone model has not been explicitly trained on).

Model	Embedding dimension	Number of heads	Number of layers	Parameter count
XTab (Zhu et al. 2023)	192	8	3	740 k
S	16	4	4	13 k
M	64	8	4	200 k
L	128	8	8	1.6 M
XL	192	16	36	16 M

Table 1: Investigated model configurations. The configuration of XTab is shown as reference. Stated number of parameters correspond to backbone parameters only and does not include embedding and projection layers for prediction or linear probing.

In the cross-table setup, pretraining is performed on the large pretraining corpus. To evaluate the model on the used benchmark datasets, the backbone can directly be transferred. However, since the architecture utilizes table-specific tokenizers, the corresponding new tokenizers need be trained. To this end, we again utilize a self-supervised training of the tokenizers, either with a frozen or a trainable transferred backbone, via Masked Cell Recovery and evaluate the representations via linear probing. That is, again, we do not perform any task-specific supervised finetuning of the architecture but evaluate the transferability of the representations learned via Masked Cell Recovery including a cross-table pretraining of the backbone.

For our experiments, we investigate four different backbone model configurations covering four orders of magnitude in the number of learnable parameters. The details are given in Table 1.

4.4 Preliminary results

The main scaling results are depicted in Figure 3. First, we note that the self-supervised pretraining strategy via Masked Cell Recovery *does* result in tabular features that show good linear separability with respect to the respective downstream tasks despite never being explicitly trained on them. In most cases, the performance is on par with a linear model and slightly worse than a (non-optimized) XGBoost. However, note that the used benchmark datasets are historically biased towards conventional methods, and already linear models perform arguably well. With respect to scaling, we initially see an increase with larger scales across all datasets, while scaling beyond the M model leads to either saturation or performance degradation. This is expected as the models are highly overparametrized with respect to the dataset sizes. For example, while the HIGGS dataset contains over 8 M rows, HELOC consists of less than 8 k samples. With increasing scale but limited data, it is not uncommon that model performance degrades (Kaplan et al. 2020).

In the cases where the backbone was first trained in a self-supervised manner across the pretraining corpus and then transferred to the individual benchmark datasets, we see a slight boost in most downstream tasks. Further updating the backbone model via additional self-supervised training performs in higher performance, as expected. Notably, the cross-table pretrained models scale slightly better with an increased number of parameters, but not as predictably as anticipated. In some instances, performance degradates with larger models. There are a number of potential reasons for this. First, in particular the larger models are not trained until fully saturated. That is, further training is likely to result in a higher downstream performance. Nevertheless, the amount of pretraining data, while sufficient for the considered backbone model size, is likely too small, in particular with respect to the large effective vocabulary of the union of tokenizers due to the table-specific embeddings. In fact, the resulting vocabulary is roughly of the same size as the one used to train GPT-2, while our backbone model is orders of magnitude smaller than the one utilized in GPT-2. We discuss this in more detail in our technical report.

4.5 Open questions and future work

As discussed in Section 3, there are a number of unaddressed problems and questions for future work that we believe our approach to be a suitable baseline for. In particular, the tokenization opens multiple interesting opportunities for further research that can now be compared with a standard baseline. For example:

- How can one reduce the parameter overhead introduced by table-specific encodings? Are low-dimensional embeddings combined with a shared up-projection layer a viable mitigation?
- How does the quantile encoding of the numerical features depend on the number of quan-

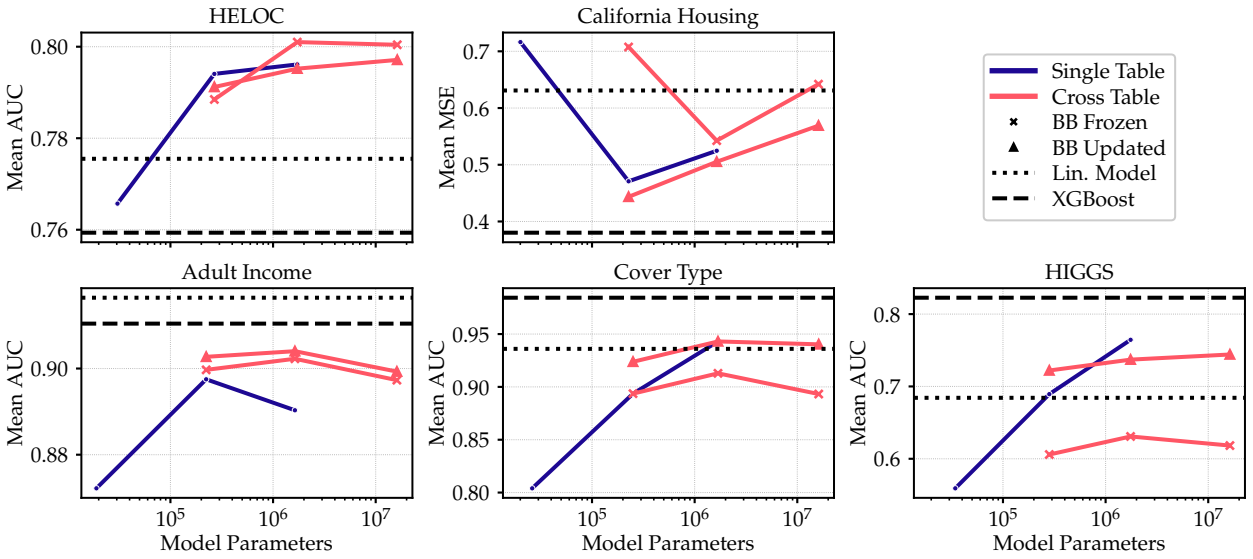

Figure 3: Mean 5-fold cross-validation linear probing results for the considered benchmark datasets in the case of single-table as well as cross-table pretraining with frozen and updated backbones (BB). The (supervised) performance of a linear model as well as XGBoost are shown for comparison.

tiles chosen? Can they be chosen adaptively to reduce unnecessary complexity depending on the specific columns?

- How does quantile encoding compare to other encoding methods for numerical features such as simple linear layers or more complex variants using piece-wise linear functions or periodic activation functions which have been proposed in the literature? Can we validate our intuition that quantile encodings are more stable as well as easy and efficient to implement?

- Can one incorporate the table schema explicitly in tokenization to reduce the complexity of the table-specific embeddings learned by the tokenizer?

- How can tokenization be extended to free-text inputs, as well as dates or timestamps?

- How can the tokenization be transferred to unseen data more efficiently, i.e. in a zero- or few-shot fashion?

- To this end, can we make use of in-context learning or retrieval-augmented modelling similar to previous works in the natural language, graph, and tabular domain (Brown et al. 2020; Q. Huang et al. 2023; Gorishniy, Rubachev, Kartashev, et al. 2023; Hollmann et al. 2023)?

5 Conclusions

In this whitepaper, we have discussed the topic of tabular representation learning at scale towards designing and training general-purpose Tabular Foundation Models. We discussed the current literature on the topic and highlighted opportunities as well as challenges that we consider crucial to address. As discussed, we believe that current approaches, in particular those based on Large Language Models, are limited and that dedicated attention towards table-specific architectures and evaluation benchmarks is required. At the same time, scalable architectures for tabular data could unlock a huge potential for a wide variety of industry and business applications abundant in practice. Whereas recent progress in natural language processing, computer vision, and other media-centric modalities has been staggering, deep learning approaches for tabular data remain particularly underdeveloped and under-researched.

Throughout, our main stances on the matter are:

- We believe that Tabular Foundation Models have the potential to unlock tremendous business value through their ability to capture decisive synergies between structured data assets, thereby developing an inherent understanding for business objects and processes. This would allow for solutions scaling across use cases and customers, enabling better predictive capabilities, including in settings where data is scarce, and greatly simplifying devel-

opment and roll-out, both on the developer and client side.

- We argue that pretrained Large Language Models are ill-suited to be used with tabular data, in particular for complex tabular tasks, due mainly to their text-specific tokenization approaches and lack of permutation invariance. We believe that Tabular Foundation Models can be realized more efficiently by designing and training them from scratch – tailored to the specifics of tabular data.
- We believe the Transformer to be a well-suited architecture with high scaling potential, at the core of which tabular tokenizers need to be better understood and further developed. Using a general Transformer architecture as opposed to table-specific modifications also allows to transfer all technical recent developments mostly present in the context of natural language and computer vision, such as efficient Transformer implementations and variants as well as optimized transfer and finetuning approaches.
- We think that additional public tabular benchmarks as well as pretraining datasets are of utmost importance to bring the tabular domain into the deep learning era as well as to foster and streamline community-wide progress.

5.1 Acknowledgements

This project was funded by SAP SE. We thank Alma Lindborg, Dominique Paul, Paul Rupprecht, Sam Thelin, Markus Kohler, Jonas Kolk, Johannes Hoffart for their valuable contributions in the creation of this manuscript.

References

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, pp. 785–794.

Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin (2017). "CatBoost: Gradient Boosting with Categorical Features Support". In: *NeurIPS ML Systems Workshop*.

Ke, Guolin et al. (2017). "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems*.

Vaswani, Ashish et al. (2017). "Attention Is All You Need". In: *Advances in Neural Information Processing System*.

Chui, Michael et al. (2018). *Notes From the AI Frontier: Insights from Hundreds of Use Cases*. Tech. rep. McKinsey Global Institute.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Houlsby, Neil et al. (2019). "Parameter-Efficient Transfer Learning for NLP". In: *International Conference on Machine Learning*. PMLR, pp. 2790–2799.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals (2019). "Representation Learning with Contrastive Predictive Coding". In: *arXiv:1807.03748*.

Wang, Alex and Kyunghyun Cho (2019). "BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model". In: *NAACL Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pp. 30–36.

Brown, Tom B et al. (2020). "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems*.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (2020). "A simple framework for contrastive learning of visual representations". In: *International Conference on Machine Learning*.

Deng, Xiang et al. (2020). "TURL: Table Understanding through Representation Learning". In: *arXiv:2006.14806*.

Grill, Jean-Bastien et al. (2020). "Bootstrap your own latent: A new approach to self-supervised Learning". In: *Advances in Neural Information Processing Systems*.

Herzig, Jonathan et al. (2020). "TAPAS: Weakly Supervised Table Parsing via Pre-training". In: *Annual Meeting of the Association for Computational Linguistics*.

Ho, Jonathan, Ajay Jain, and Pieter Abbeel (2020). "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems*.

Huang, Xin, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin (2020). "TabTransformer: Tabular Data Modeling Using Contextual Embeddings". In: *arXiv:2012.06678*.
Kaplan, Jared et al. (2020). "Scaling Laws for Neural Language Models". In: *arxiv:2001.08361*.
Yin, Pengcheng, Graham Neubig, Wen-tau Yih, and Sebastian Riedel (2020). "TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data". In: *Annual Meeting of the Association for Computational Linguistics*, pp. 8413–8426.
Yoon, Jinsung, Yao Zhang, James Jordon, and Mihaela van der Schaar (2020). "VIME: Extending the Success of Self- and Semi-supervised Learning to Tabular Domain". In: *Advances in Neural Information Processing Systems*.
Arora, Siddhant, Vinayak Gupta, Garima Gaur, and Srikanta Bedathur (2021). "BERT Meets Relational DB: Contextual Representations of Relational Databases". In: *arXiv:2104.14914*.
Bischl, Bernd et al. (2021). "OpenML Benchmarking Suites". In: *Advances in Neural Information Processing Systems*.
Bommasani, Rishi et al. (2021). "On the Opportunities and Risks of Foundation Models". In: *arXiv:2108.07258*.
Caron, Mathilde et al. (2021). "Emerging properties in self-supervised vision transformers". In: *International Conference on Computer Vision*.
Chen, Sanxing et al. (2021). "HittER: Hierarchical Transformers for Knowledge Graph Embeddings". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10395–10407.
Darabi, Sajad et al. (2021). "Contrastive Mixup: Self- and Semi-Supervised learning for Tabular Domain". In: *arXiv:2108.12296*.
Dosovitskiy, Alexey et al. (2021). "An Image is worth 16x16 words: Transformers for image recognition at scale". In: *International Conference on Learning Representations*.
Du, Nan et al. (2021). "GLaM: Efficient Scaling of Language Models with Mixture-of-Experts". In: *arXiv:2112.06905*.
Gorishniy, Yury, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko (2021). "Revisiting Deep Learning Models for Tabular Data". In: *Advances in Neural Information Processing Systems*.
Mielke, Sabrina J. et al. (2021). "Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP". In: *arXiv:2112.10508*.
Radford, Alec et al. (2021). "Learning Transferable Visual Models From Natural Language Supervision". In: *International Conference on Machine Learning*.
Ramesh, Aditya et al. (2021). "Zero-Shot Text-to-Image Generation". In: *International Conference on Machine Learning*.
Shwartz-Ziv, Ravid and Amitai Armon (2021). "Tabular Data: Deep Learning is Not All You Need". In: *ICML Workshop on Automated Machine Learning*.
Thawani, Avijit, Jay Pujara, Filip Ilievski, and Pedro Szekely (2021). "Representing Numbers in NLP: a Survey and a Vision". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, pp. 644–656.
Ucar, Talip, Ehsan Hajiramezanali, and Lindsay Edwards (2021). "SubTab: Subsetting Features of Tabular Data for Self-Supervised Representation Learning". In: *arXiv:2110.04361*.
Zbontar, Jure et al. (2021). "Barlow Twins: Self-Supervised Learning via Redundancy Reduction". In: *International Conference on Machine Learning*. PMLR, pp. 12310–12320.
Bachmann, Roman, David Mizrahi, Andrei Atanov, and Amir Zamir (2022). "MultiMAE: Multi-modal Multi-task Masked Autoencoders". In: *European Conference on Computer Vision*.
Bahri, Dara, Heinrich Jiang, Yi Tay, and Donald Metzler (2022). "SCARF: Self-Supervised Contrastive Learning using Random Feature Corruption". In: *International Conference on Learning Representations*.
Bardes, Adrien, Jean Ponce, and Yann LeCun (2022). "VICReg: Variance-invariance-covariance regularization for self-supervised learning". In: *International Conference on Learning Representations*.
Bi, Zhen et al. (2022). "Relphormer: Relational Graph Transformer for Knowledge Graph Representations". In: *arXiv.2205.10852*.
Borisov, Vadim, Tobias Leemann, et al. (2022). "Deep Neural Networks and Tabular Data: A Survey". In: *IEEE Transactions on Neural Networks and Learning Systems* 99, pp. 1–21.
Chowdhery, Aakanksha et al. (2022). "PaLM: Scaling Language Modeling with Pathways". In: *arXiv:2204.02311*.
Gorishniy, Yury, Ivan Rubachev, and Artem Babenko (2022). "On Embeddings for Numerical Features in Tabular Deep Learning". In: *Advances in Neural Information Processing Systems*.

Grinsztajn, Leo, Edouard Oyallon, and Gael Varoquaux (2022). "Why Do Tree-Based Models Still Outperform Deep Learning on Typical Tabular Data?" In: *Advances in Neural Information Processing Systems*.
He, Kaiming et al. (2022). "Masked Autoencoders Are Scalable Vision Learners". In: *Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009.
Hu, Edward J. et al. (2022). "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations*.
Liu, Qian et al. (2022). "TAPEX: Table Pre-training via Learning a Neural SQL Executor". In: *International Conference on Learning Representations*.
Majmundar, Kushal, Sachin Goyal, Praneeth Netrapalli, and Prateek Jain (2022). "MET: Masked Encoding for Tabular Data". In: *NeurIPS Table Representation Learning Workshop*.
Narayan, Avanika, Ines Chami, Laurel Orr, and Christopher Ré (2022). "Can Foundation Models Wrangle Your Data?" In: *International Conference on Very Large Data Bases* 16.4, pp. 738–746.
Pi, Xinyu et al. (2022). "Reasoning Like Program Executors". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 761–779.
Pietruszka, Michał et al. (2022). "STable: Table Generation Framework for Encoder-Decoder Models". In: *NeurIPS Table Representation Learning Workshop*.
Rombach, Robin et al. (2022). "High-Resolution Image Synthesis with Latent Diffusion Models". In: *Conference on Computer Vision and Pattern Recognition*, pp. 10674–10685.
Rubachev, Ivan, Artem Alekberov, Yury Gorishniy, and Artem Babenko (2022). "Revisiting Pretraining Objectives for Tabular Deep Learning". In: *arXiv:2207.03208*.
Schaeffer, Rylan, Brando Miranda, and Sanmi Koyejo (2022). "Are Emergent Abilities of Large Language Models a Mirage?" In: *ICML Workshop on Deployable Generative AI*.
Sohofi, Alireza et al. (2022). "Squirrel: A Python library that enables ML teams to share, load, and transform data in a collaborative, flexible, and efficient way". In: *GitHub. https://github. com/merantix-momentum/squirrel-core*.
Somepalli, Gowthami et al. (2022). "SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training". In: *NeurIPS Table Representation Learning Workshop*.
Vogel, Liane, Benjamin Hilprecht, and Carsten Binnig (2022). "Towards Foundation Models for Relational Databases". In: *NeurIPS Table Representation Learning Workshop*.
Wang, Zifeng and Jimeng Sun (2022). "TransTab: Learning Transferable Tabular Transformers Across Tables". In: *Advances in Neural Information Processing Systems* 35, pp. 2902–2915.
Wei, Jason et al. (2022). "Emergent Abilities of Large Language Models". In: *arXiv.2206.07682*.
Yang, Jingfeng et al. (2022). "TableFormer: Robust Transformer Modeling for Table-Text Encoding". In: *Annual Meeting of the Association for Computational Linguistics*, pp. 528–537.
Anil, Rohan et al. (2023). "Gemini: A Family of Highly Capable Multimodal Models". In: *arXiv:2312.11805*.
Badaro, Gilbert, Mohammed Saeed, and Paolo Papotti (2023). "Transformers for Tabular Data Representation: A Survey of Models and Applications". In: *Transactions of the Association for Computational Linguistics* 11, pp. 227–249.
Balestriero, Randall et al. (2023). "A Cookbook of Self-Supervised Learning". In: *arXiv:2304.12210*.
Borisov, Vadim, Kathrin Seßler, et al. (2023). "Language Models are Realistic Tabular Data Generators". In: *International Conference on Learning Representations*.
Bubeck, Sébastien et al. (2023). "Sparks of Artificial General Intelligence: Early experiments with GPT-4". In: *arXiv:2303.12712*.
Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer (2023). "QLoRA: Efficient Finetuning of Quantized LLMs". In: *arXiv:2305.14314* [*cs*].
Diao, Cameron and Ricky Loynd (2023). "Relational Attention: Generalizing Transformers for Graph-Structured Tasks". In: *International Conference on Learning Representations*.
Eggert, Gus, Kevin Huo, Mike Biven, and Justin Waugh (2023). "TabLib: A Dataset of 627M Tables with Context". In: *arXiv:2310.07875*.
Fischer, Sebastian Felix, Matthias Feurer, and Bernd Bischl (2023). "OpenML-CTR23 – A Curated Tabular Regression Benchmarking Suite". In: *AutoML Conference Workshop*.
Gao, Peng et al. (2023). "LLaMA-Adapter V2: Parameter-Efficient Visual Instruction Model". In: *arXiv:2304.15010*.
Girdhar, Rohit et al. (2023). "ImageBind: One Embedding Space To Bind Them All". In: *Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190.

Golkar, Siavash et al. (2023). "xVal: A Continuous Number Encoding for Large Language Models". In: *arXiv:2310.02989*.
Gorishniy, Yury, Ivan Rubachev, Nikolay Kartashev, et al. (2023). "TabR: Tabular Deep Learning Meets Nearest Neighbors in 2023". In: *arXiv:2307.14338* [*cs*].
Han, Hongwei et al. (2023). "LUNA: Language Understanding with Number Augmentations on Transformers via Number Plugins and Pre-training". In: *arXiv:2212.02691* [*cs*].
Hegselmann, Stefan et al. (2023). "TabLLM: Few-shot Classification of Tabular Data with Large Language Models". In: *International Conference on Artificial Intelligence and Statistics*.
Hollmann, Noah, Samuel Müller, Katharina Eggensperger, and Frank Hutter (2023). "TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second". In: *International Conference on Learning Representations*.
Huang, Qian et al. (2023). "PRODIGY: Enabling In-context Learning Over Graphs". In: *ICML Workshop on Structured Probabilistic Inference & Generative Modeling*. arXiv:2305.12600 [cs]. arXiv.
Ioannidis, Vassilis N. et al. (2023). "Efficient and Effective Training of Language and Graph Neural Network Models". In: *AAAI Workshop on Knowledge Augmented Methods for Natural Language Processing*.
Kotelnikov, Akim, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko (2023). "TabDDPM: Modelling Tabular Data with Diffusion Models". In: *International Conference on Machine Learning*. PMLR, pp. 17564–17579.
Levin, Roman et al. (2023). "Transfer Learning with Deep Tabular Models". In: *International Conference on Learning Representations*.
Li, Xianzhi et al. (2023). "Are ChatGPT and GPT-4 General-Purpose Solvers for Financial Text Analytics? An Examination on Several Typical Tasks". In: *arXiv:2305.05862*.
Lu, Sheng et al. (2023). "Are Emergent Abilities in Large Language Models just In-Context Learning?" In: *arXiv:2309.01809*.
McElfresh, Duncan et al. (2023). "When Do Neural Nets Outperform Boosted Trees on Tabular Data?" In: *Advances in Neural Information Processing Systems*.
Onishi, Soma, Kenta Oono, and Kohei Hayashi (2023). "TabRet: Pre-training Transformer-based Tabular Models for Unseen Columns". In: *ICLR Workshop on Understanding Foundation Models*.
Pan, Shirui et al. (2023). "Unifying Large Language Models and Knowledge Graphs: A Roadmap". In: *arXiv:2306.08302*.
Schambach, Maximilian, Dominique Paul, and Johannes S. Otterbach (2023). "Scaling Experiments in Self-Supervised Cross-Table Representation Learning". In: *NeurIPS Table Representation Learning Workshop*.
Touvron, Hugo et al. (2023). "LLaMA: Open and Efficient Foundation Language Models". In: *arXiv*. arXiv:2302.13971.
Wang, Zifeng, Chufan Gao, Cao Xiao, and Jimeng Sun (2023). "AnyPredict: Foundation Model for Tabular Prediction". In: *arXiv:2305.12081*.
Wu, Shijie et al. (2023). "BloombergGPT: A Large Language Model for Finance". In: *arXiv:2303.17564*.
Yang, Yazheng et al. (2023). "UniTabE: Pretraining a Unified Tabular Encoder for Heterogeneous Tabular Data". In: *arXiv:2307.09249*.
Yuan, Zheng et al. (2023). "How Well Do Large Language Models Perform in Arithmetic Tasks?" In: *arXiv:2304.02015*.
Zha, Liangyu et al. (2023). "TableGPT: Towards Unifying Tables, Nature Language and Commands into One GPT". In: *arXiv:2307.08674*.
Zhang, Han et al. (2023). "Towards Foundation Models for Learning on Tabular Data". In: *arXiv:2310.07338*.
Zhang, Tianping et al. (2023). "Generative Table Pre-training Empowers Models for Tabular Prediction". In: *arXiv:2305.09696*.
Zhu, Bingzhao et al. (2023). "XTab: Cross-table Pretraining for Tabular Transformers". In: *International Conference On Machine Learning*.

merantix-momentum.com
momentum@merantix.com

AI Campus Berlin
Max-Urich-Str. 3
13355, Berlin, DE