Tabular Transformers for Modeling Multivariate Time Series

Inkit Padhi, Yair Schiff, Igor Melnyk, Mattia Rigotti, Youssef Mroueh, Pierre Dognin Jerret Ross, Ravi Nair, Erik Altman
Correspondence to inkpad@ibm.com, mroueh@us.ibm.com
IBM Research & MIT-IBM Watson AI Lab

Abstract

Tabular datasets are ubiquitous in data science applications. Given their importance, it seems natural to apply state-of-the-art deep learning algorithms in order to fully unlock their potential. Here we propose neural network models that represent *tabular time series* that can optionally leverage their hierarchical structure. This results in two architectures for tabular time series: one for learning representations that is analogous to BERT and can be pre-trained end-to-end and used in downstream tasks, and one that is akin to GPT and can be used for generation of realistic synthetic tabular sequences. We demonstrate our models on two datasets: a synthetic credit card transaction dataset, where the learned representations are used for fraud detection and synthetic data generation, and on a real pollution dataset, where the learned encodings are used to predict atmospheric pollutant concentrations. Code and data are available at `https://github.com/IBM/TabFormer`.

1 Introduction

Tabular datasets are ubiquitous across many industries, especially in vital sectors such as healthcare and finance. Such industrial datasets often contain sensitive information, raising privacy and confidentiality issues that preclude their public release and limit their analysis to methods that are compatible with an appropriate anonymization process. We can distinguish between two types of tabular data: *static* tabular data that corresponds to independent rows in a table, and *dynamic* tabular data that corresponds to tabular time series, also referred to also as multivariate time series. The machine learning and deep learning communities have devoted considerable effort to learning from static tabular data, as well as generating synthetic static tabular data that can be released as a privacy compliant surrogate of the original data. On the other hand, less effort has been devoted to the more challenging dynamic case, where it is important to also account for the temporal component of the data. The purpose of this paper is to remedy this gap by proposing deep learning techniques to: 1) learn useful representation of tabular time series that can be used in downstream tasks such as classification or regression and 2) generate realistic synthetic tabular time series.

Tabular time series represent a hierarchical structure that we leverage by endowing transformer-based language models with field-level transformers, which encode individual rows into embeddings that are in turn treated as embedded tokens that are passed to BERT [5]. This results in an alternative architectures for tabular time series encoding that can be pre-trained end-to-end for representation learning that we call Tabular BERT (TabBERT). Another important contribution is adapting state-of-the-art (SOTA) language generative models GPT [15] to produce realistic synthetic tabular data that we call Tabular GPT (TabGPT). A key ingredient of our *language metaphor* in modeling tabular time series is the quantization of continuous fields, so that each field is defined on a finite vocabulary, as in language modeling.

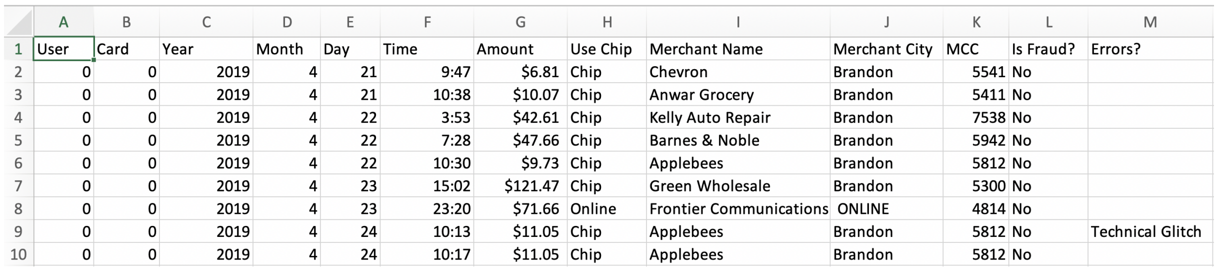
As mentioned, static tabular data have been widely analyzed in the past, typically with feature engineering and classical learning schemes such as gradient boosting or random forests. Recently, [2] introduced TabNet, which uses attention to perform feature selection across fields and shows the advantages of deep learning over classical approaches. A more recent line of work [8, 22] concurrent to ours, deals with the joint processing of static tabular and textual data using transformer architectures, such as BERT, with the goal of querying tables with natural language. These works consider the static case, and to the best of our knowledge, our work is the first to address tabular time series using transformers.

On the synthetic generation side, a plethora of work [4, 11, 19–21] are dedicated to generating static tabular data using Generative Adversarial Networks (GANs), conditional GANs, and variational Auto-Encoders. [3, 6] argue for the importance of synthetic generation on financial tabular data in order to preserve user privacy and to allow for training on cloud-based solutions without compromising real users' sensitive information. Nevertheless, their generation scheme falls short of modeling the temporal dependency in the data. Our work addresses this crucial aspect in particular. In summary, the main contributions of our paper are:

• We propose Hierarchical Tabular BERT to learn representations of tabular time series that can be used in downstream tasks such as classification or regression.
• We propose TabGPT to synthesize realistic tabular time series data.
• We train our proposed models on a synthetic credit card transaction dataset, where the learned encodings are used for a downstream fraud detection task and for synthetic data generation. We also showcase our method on a public real-world pollution dataset, where the learned encodings are used to predict the concentration of pollutant.
• We open-source our synthetic credit-card transactions dataset to encourage future research on this type of data. The code to reproduce all experiments in this paper is available at https://github.com/IBM/TabFormer. Our code is built within HuggingFace's framework [18].

2 TabBERT: Unsupervised Learning of Multivariate time series Representation

2.1 From Language Modeling to Tabular Time Series



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	User	Card	Year	Month	Day	Time	Amount	Use Chip	Merchant Name	Merchant City	MCC	Is Fraud?	Errors?
2	0	0	2019	4	21	9:47	$6.81	Chip	Chevron	Brandon	5541	No	
3	0	0	2019	4	21	10:38	$10.07	Chip	Anwar Grocery	Brandon	5411	No	
4	0	0	2019	4	22	3:53	$42.61	Chip	Kelly Auto Repair	Brandon	7538	No	
5	0	0	2019	4	22	7:28	$47.66	Chip	Barnes & Noble	Brandon	5942	No	
6	0	0	2019	4	22	10:30	$9.73	Chip	Applebees	Brandon	5812	No	
7	0	0	2019	4	23	15:02	$121.47	Chip	Green Wholesale	Brandon	5300	No	
8	0	0	2019	4	23	23:20	$71.66	Online	Frontier Communications	ONLINE	4814	No	
9	0	0	2019	4	24	10:13	$11.05	Chip	Applebees	Brandon	5812	No	Technical Glitch
10	0	0	2019	4	24	10:17	$11.05	Chip	Applebees	Brandon	5812	No	

Figure 1: An example of sequential tabular data, where each row is a transaction. A to M are the fields of the transactions. Some of the fields are categorical, others are continuous, but through quantization we convert all fields into categorical. Each field is then processed to build its own local vocabulary. A single sample is defined as some number of contiguous transactions, for example rows 1 through 10, as shown in this figure.

In Fig. 1, we give an example of tabular time series, that is a sequence of card transactions for a particular user. Each row consists of fields that can be continuous or categorical. In order to unlock the potential of language modeling techniques for tabular data, we quantize continuous fields so that each field is defined on its own local finite vocabulary. We define a sample as a sequence of rows (transactions in this case). The main difference with NLP is that we have a sequence of structured rows consisting each of fields defined on local vocabulary. As introduced in previous sections, unsupervised learning for multivariate time series representations require modeling both inter- and intra-transaction dependencies. In the next section, we show how to exploit this hierarchy in learning unsupervised representations for tabular time series.

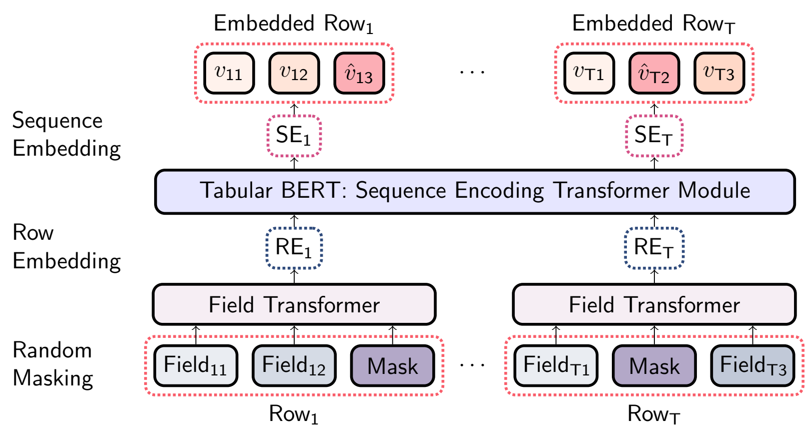


Figure 2: TabBERT: Field level masking and cross entropy.

2.2 Hierarchical Tabular BERT

In order to learn representations for multivariate tabular data, we use a recipe similar to the one employed for training language representation using BERT. The encoder is trained through masked language modeling (MLM), i.e by predicting masked tokens. More formally, given a table with M rows and N columns (or fields), an input t_j to TabBERT is represented as a windowed series of time-dependent rows,

$$t_j = [t_{i+1}, t_{i+2}, ..., t_{i+T}], \tag{1}$$

where T ($\ll M$) is the number of contiguous rows, and i is a window offset (or stride). TabBERT is a variant of BERT, which accommodates the temporal nature of rows in the tabular input. As shown in Fig. 2, TabBERT encodes the series of transactions in a hierarchical fashion. The field transformer processes rows individually, creating transaction/row embeddings. These transaction embeddings are then fed to a second-level transformer to create sequence embeddings. In other words, the field transformer tries to capture intra-transaction relationships (local), whereas the sequence transformer encodes inter-transaction relationships (global), capturing the temporal component of the data. Note that hierarchical transformer has already been proposed in NLP in order to hierarchically model documents [7, 14, 23].

Many pre-trained transformer models on domain specific data have recently been successfully applied in various downstream tasks. More specifically, BioBERT [12], VideoBERT [17], ERNIE [24], ClinicalBERT [10] are pre-trained efficiently on various domains, such as biomedical, YouTube videos, knowledge graph, and clinical electronic health record, respectively. Representations from these BERT variants have shown to achieve SOTA results for different tasks ranging from video captioning to hospital readmission. In order to ascertain the richness of learned TabBERT representation, we study two downstream tasks: classification (Section 2.3) and regression (Section 2.4).

2.3 TabBERT Features for Classification

Transaction Dataset One of the contributions of this work is in introducing a new synthetic corpus for credit card transactions. The transactions are created using a rule-based generator where values are produced by stochastic sampling techniques, similar to a method followed by [1]. Due to privacy concerns, most of the existing public transaction datasets are either heavily anonymized or preserved through PCA-based transformations, and thus distort the real data distributions. The proposed dataset has 24 million transactions from 20,000 users. Each transaction (row) has 12 fields (columns) consisting of

Features	Prediction Head	**Fraud** F1	**PRSA** RMSE
Raw	MLP	0.74	38.5
	LSTM	0.83	43.3
TabBERT	MLP	0.76	34.2
	LSTM	**0.86**	**32.8**

Table 1: Performance comparison on the classification task of fraud detection (Fraud), and the regression task of pollution prediction (PRSA) for two approaches: one based on TabBERT features and the baseline using raw data. We compare two architectures: MLP and LSTM for the downstream tasks.

both continuous and discrete nominal attributes, such as merchant name, merchant address, transaction amount, etc.

Training TabBERT For training TabBERT on our transaction dataset, we create samples as sliding windows of 10 transactions, with a stride of 5. The continuous and categorical values are quantized, and a vocabulary is created, as described in Section 2.1. Note that during training we exclude the label column `isFraud?`, to avoid biasing the learned representation for the downstream fraud detection task. Similar to strategies used by [5], we mask 15% of a sample's fields, replacing them with the [Mask] token, and predict the original field token with cross entropy loss.

Fraud Detection Task For fraud detection, we create samples by combining 10 contiguous rows (with a stride of 10) in a time-dependent manner for each user. In total, there are 2.4M samples with 29,342 labeled as fraudulent. Since in real world the fraudulent transactions are very rare events, a similar trend is observed in our synthetic data, resulting in an imbalanced, non-uniform distribution of fraudulent and non-fraudulent class labels. To account for this imbalance, during training, we upsample the fraudulent class to roughly equalize the frequencies of both classes. We evaluate performance of different methods using F1 binary score, on a test set consisting of 480K samples. As a baseline, we use a multi-layer perceptron (MLP) trained directly on the embeddings of the raw features. In order to model temporal dependencies, we also use an LSTM network baseline on the raw embedded features. In both cases, we pool the encoder outputs at individual row level to create E_i (see Fig. 2) before doing classification. In Tab. 1, we compare the baselines and the methods based on TabBERT features while using the same architectures for the prediction head. During MLP and LSTM networks training, with TabBERT as the feature extractor, we freeze the TabBERT network foregoing any update of its weights. As can be seen from the table, the inclusion of TabBERT features boosts the F1 for the fraud detection task.

2.4 TabBERT Features for Regression Tasks

Pollution Dataset For the regression task, we use a public UCI dataset (Beijing PM2.5 Data) for predicting both PM2.5 and PM10 air concentration for 12 monitoring sites, each containing around 35k entries (rows). Every row has 11 fields with a mix of both continuous and discrete values. For a detailed description of the data, please refer to [13]. Similar to the pre-processing steps for our transaction dataset, we quantize the continuous features, remove the targets (*PM2.5* and *PM10*), and create samples by combining 10 time-dependent rows with a stride of 5. We use 45K samples for training and report a combined RMSE for both targets from the test set of 15K samples. As reported in Tab. 1, using TabBERT features shows significant improvement in terms of RMSE over the case of using simple raw embedded features. This consistent performance gain when using TabBERT features for both classification and regression tasks underlines the richness of representations learned from TabBERT.

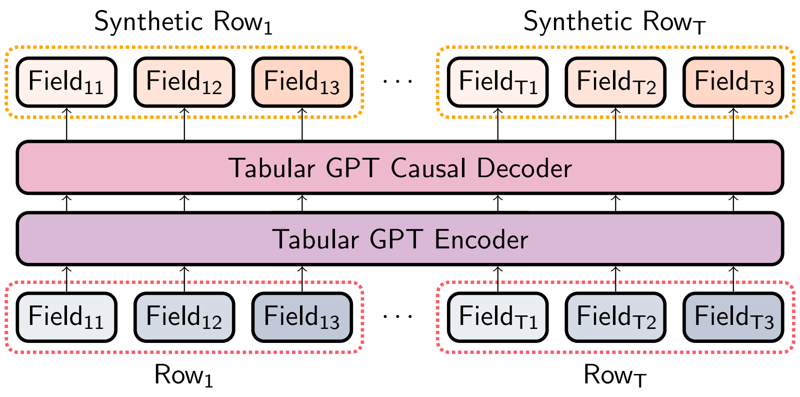


Figure 3: TabGPT: Synthetic Transaction GPT Generator.

3 TabGPT: Generative modeling of Multivariate Time Series Tabular Data

Another useful application of language modeling in the context of tabular, time-series data is the preservation of data privacy. GPT models trained on large corpora have demonstrated human-level capabilities in the domain of text generation. In this work, we apply the generative capabilities of GPT as a proof of concept for creating synthetic tabular data that is close in distribution to the true data, with the advantage of not exposing any sensitive information. Specifically, we train a GPT model (referred to throughout as TabGPT) on user-level data from the credit card dataset in order to generate synthetic transactions that mimic a user's purchasing behavior. This synthetic data can subsequently be used in downstream tasks without the precautions that would typically be necessary when handling private information.

We begin, as with TabBERT, by quantizing the data to create a finite vocabulary for each field. To train the TabGPT model, we select specific users from the dataset. By ordering a user's transactions chronologically and segmenting them into sequences of ten transactions, the model learns to predict future behavior from past transactions, similar to how GPT language models are trained on text data to predict future tokens from past context. We apply this approach to two of the users that have a relatively high volume of transactions, each with $\sim$60k transactions. For each user, we train a separate TabGPT model, which is depicted in Fig. 3. Unlike with TabBERT, we do not employ the hierarchical structure of passing each field into a field-level transformer, but rather we pass sequences of transactions separated by a special [SEP] token directly to the GPT encoder network.

After training, synthetic data is generated by again segmenting a user's transaction data into sequences of ten transactions, passing the first transaction of each group of ten to the model, and predicting the remaining nine. To evaluate a model's generative capabilities we examine how it captures both the aggregate and time-dependent features of the data.

The quantization of non-categorical data, which enables the use of a finite vocabulary for each field, renders field level evaluation of the fidelity of TabGPT to the real data more straightforward. Namely, for each field, we compute and compare histograms for both ground truth and generated data on an aggregate level over all timestamps. To measure proximity of the true and synthetic distributions we calculate the χ^2 distance between them, defined as: $\chi^2(\mathcal{X}, \mathcal{X}') = \frac{1}{2}\sum_{i=1}^{n} \frac{(x_i - x_i')^2}{(x_i + x_i')}$ where x_i, x_i' are columns from the corresponding transactions ($i = 1..n$) from the true ($\mathcal{X}$) and generated ($\mathcal{X}'$) distributions, respectively. In Fig. 4, we plot results of this evaluation for the two selected users. Overall, we see that for both users, their respective TabGPT models are able to generate synthetic distributions that are similar to the ground truth for each feature of the data, even for columns with high entropy, such as

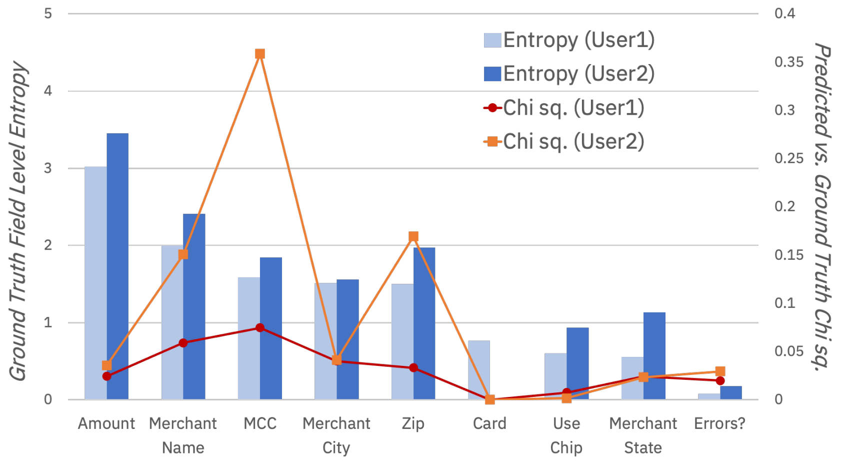


Figure 4: For each column in the tabular data, we compare the generated and ground truth distributions for the user's data rows. The entropy of each feature is represented by the bars and displayed on the left vertical axis and χ^2 distance between real and synthetic data distributions is represented by the line and displayed on right vertical axis.

`Amount`. The TabGPT model for *user 1* produces distributions that are generally closer to ground truth, but for *user 2*, most column distributions also align closely.

While field distribution matching evaluates fidelity of generated data to real data on an aggregate level, this analysis does not capture the sequential nature of the generation. Hence we use an additional metric that compares two datasets of time series $(t^a_{1,i}, \dots t^a_{T,i})_{i=1\dots N}$ and $(t^b_{1,i}, \dots t^b_{T,i})_{i=1\dots N}$. Inspired by the *Fréchet Inception Distance* (FID) [9] used in computer vision and *Fréchet InferSent Distance* (FD) [16] in NLP, we use our TabBERT model to embed real and generated sequence to a fixed length vector for each instance $v^a_i = \text{TabBERT}((t^a_{1,i}, \dots t^a_{T,i}))$, and $v^b_i = \text{TabBERT}((t^b_{1,i}, \dots t^b_{T,i}))$. v^a_i is obtained by mean pooling all time-wise embeddings $\texttt{SE}_t$ in TabBERT. Then we compute mean and covariance for each dataset (μ_a, Σ_a) and (μ_b, Σ_b), respectively. The FID score is defined as follows:

$$\text{FID}_{a,b} = ||\mu_a - \mu_b||_2^2 + Tr(\Sigma_a + \Sigma_b - 2(\Sigma_a \Sigma_b)^{\frac{1}{2}}) \qquad (2)$$

FID		Real	
		User 1	*User 2*
Real	*User 1*	-	492.92
GPT-Gen	*User 1*	22.90	497.68
	User 2	515.94	49.08

Table 2: FID between real and GPT-generated transactions.

FID scores between the transaction datasets for *user 1* and *user 2* are presented in Tab. 2. For the real user data, we see that they have different behaviors, with FID of 492.95. In contrast, the TabGPT generated data (GPT-Gen) for *user 1* matches the real *user 1* more closely, as can be seen from the relatively low FID score. The same conclusion holds for GPT-Gen *user 2*. Interestingly the cross distances between the generated user and the other real user are also maintained. The combination of the aggregate histogram and FID analyses indicates that TabGPT is able to learn the behavior of each user and to generate realistic synthetic transactions.

4 Conclusion

In this paper, we introduce Hierarchical Tabular BERT and Tabular GPT for modeling multivariate times series. We also open-source a synthetic card transactions dataset and the code to reproduce our experiments. This type of modeling for sequential tabular data via transformers is made possible thanks to the quantization of the continuous fields of the tabular data. We show that the representations learned by TabBERT provide consistent performance gains in different downstream tasks. TabBERT features can be used in fraud detection *in lieu* of hand-engineered features as they better capture the intra-dependencies between the fields as well as the temporal dependencies between rows. Finally, we show that TabGPT can reliably synthesise card transactions that can replace real data and alleviate the privacy issues encountered when training off premise or with cloud based solutions [3, 6].

Acknowledgements

Authors acknowlege the MIT-IBM Watson AI lab and Wells Fargo's Sherif Boutros and Vanio Markov for fruitful discussions and feedback on this work.

References

[1] Erik R Altman. Synthesizing credit card transactions. *arXiv preprint arXiv:1910.03033*, 2019.

[2] Sercan O Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. *arXiv preprint arXiv:1908.07442*, 2019.

[3] Samuel Assefa, Danial Dervovic, Mahmoud Mahfouz, Tucker Balch, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: opportunities, challenges and pitfalls. *Challenges and Pitfalls (June 23, 2020)*, 2020.

[4] Ramiro Daniel Camino, Christian Hammerschmidt, et al. Working with deep generative models and tabular data imputation. 2020.

[5] 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.

[6] Dmitry Efimov, Di Xu, Luyang Kong, Alexey Nefedov, and Archana Anandakrishnan. Using generative adversarial networks to synthesize artificial financial datasets, 2020.

[7] Vikas K. Garg, Inderjit S. Dhillon, and Hsiang-Fu Yu. Multiresolution transformer networks: Recurrence is not essential for modeling hierarchical structure, 2019.

[8] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th ACL*, pages 4320–4333. ACL, July 2020.

[9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[10] Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*, 2019.

[11] Anton Karlsson and Torbjörn Sjöberg. Synthesis of tabular financial data using generative adversarial networks, 2020.

[12] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[13] Xuan Liang, Tao Zou, Bin Guo, Shuo Li, Haozhe Zhang, Shuyi Zhang, Hui Huang, and Song Xi Chen. Assessing beijing's pm2. 5 pollution: severity, weather impact, apec and winter heating. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2182):20150257, 2015.

[14] Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. Hierarchical transformers for long document classification. In *2019 IEEE ASRU Workshop*, pages 838–844. IEEE, 2019.

[15] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018.

[16] Stanislau Semeniuta, Aliaksei Severyn, and Sylvain Gelly. On accurate evaluation of gans for language generation. *arXiv preprint arXiv:1806.04936*, 2018.

[17] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473, 2019.

[18] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

[19] Lei Xu. *Synthesizing Tabular Data using Conditional GAN*. PhD thesis, Massachusetts Institute of Technology, 2020.

[20] Lei Xu and Kalyan Veeramachaneni. Synthesizing tabular data using generative adversarial networks.(2018). *arXiv preprint arXiv:1811.11264*, 2018.

[21] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional gan. In *Advances in Neural Information Processing Systems*, pages 7335–7345, 2019.

[22] Pengcheng Yin, Graham Neubig, Wen tau Yih, and Sebastian Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Annual Conference of the Association for Computational Linguistics (ACL)*, July 2020.

[23] Xingxing Zhang, Furu Wei, and Ming Zhou. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5059–5069, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1499. URL `https://www.aclweb.org/anthology/P19-1499`.

[24] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*, 2019.