MIT Sloan School of Management
Machine Learning Under Modern Optimization Lens (Fall 2023)

Context Representation for Tabular Machine Learning

Giorgio Demarchi
giodem@mit.edu

Benjamin Rio
benrio@mit.edu

Abstract

Our work builds upon the TabText framework on the income prediction task from UCI-ML. We propose refinements on the language format template and on the integration of the LLM, first as a way to generate meaningful embeddings to input in a traditional ML model, then as a mean to achieve the prediction task in an end-to-end way. Our experiments explore the trade-off between flexibility and performance when building sentences out of a table's row, finetune the LLMs with different approaches, and test for different training set sizes. We interpret the generated embeddings by reducing dimensionality and providing interpretable plots. We show the substantial improvement brought by fine tuning end-to-end the LLM, which boosts the performance from 0.81 AUC to 0.92 AUC (+0.11). Finally, we discuss the potential and limitations of this approach, while suggesting ideas for further refinement.

(LateX template borrowed from NeurIPS 2019)

Contents

1 Introduction

Standard practice of tabular machine learning often overlooks context. Contextual information emerges from various sources, including metadata such as table headers and the insights that a human data scientist adds during data processing and feature engineering optimization. This context is stored in metadata and the data scientist's knowledge, which can both be converted into textual information. With the advent of Large Language Models, there is now an opportunity to unlock the contextual information by blending it into the machine learning workflow.

TabText is a data processing framework designed to integrate contextual knowledge to a tabular data task [2]. It uses a language construction method to transform the table's metadata and the features values into sentences, and converts it into embeddings by leveraging pre-trained large language models (LLMs). The LLM can be finetuned to these descriptive sentences, in order to abstract the choice of the language construction. The authors of TabText experimented this method and obtained significant improvement in different machine learning tasks.

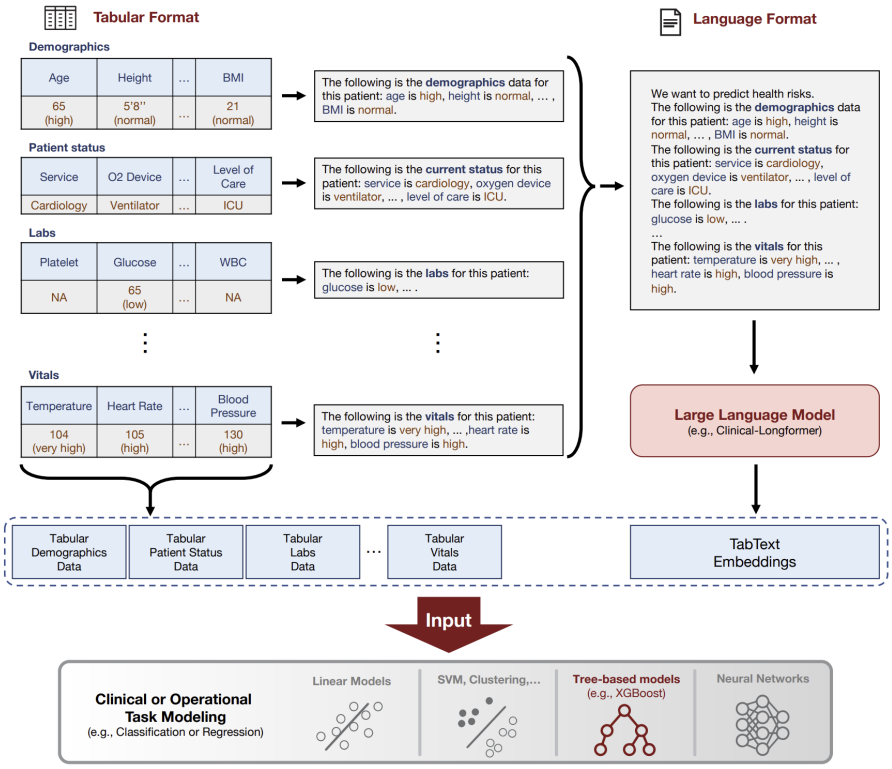

Figure 1: TabText Model

LLMs compress a vast amount of knowledge during their pre-training phase, capturing rich representations of words, including their grammatical function, meaning and real-world abstract reference. This knowledge distillation is encoded in natural language. While LLMs are pretrained in an end-to-end way, we can still execute a forward pass and obtain an informative vectorized representation from one of its last hidden layer. Tabtext authors leverage this output to feed a machine learning to solve the task at hand.

The effectiveness of the capture of the relevant contextual cues depends on the format and the structure of prompts [4]. This critical performance aspect motivates our work to develop a methodology that builds the most relevant sentences and sentence representations. For the purpose of this project, we aimed at:

- Identify datasets that could benefit from contextual encoding and test the TabText framework added value

- Evaluate the performance of different language construction methods and different LLM prompting strategies
- Implement and test the effectiveness of minor improvements across the pipeline
- Carry out an end-to-end fine tuning of the LLM model to boost classification performance

2 Dataset

We consider the Census Income dataset from UCI Machine Learning repository [3]. This dataset contains 48k rows representing characteristics of individuals, which are split between train (32k) and test (16k), and 14 features. The column "income", initially textual information representing whether the income was less than 50k (">=50k" and "<50k") is transformed into a binary variable that will be the target variable of our machine learning models. The prediction task is the classification of people earning a salary that is higher than $50k a year, based on a person's characteristics. The Being the features a mix of numerical, binary and categorical types, the dataset fits well for our purpose of testing the LLM-based processing framework.

3 Methodology

Our methodology is structured around transforming tabular data into natural language sentences, leveraging a fine-tuned LLM to extract meaningful embeddings, and then concatenating these embeddings with traditional features for enhanced predictive modeling.

3.1 Language Construction

We convert every row of the dataset into a meaningful sentence that contains the metadata, the columns names and the values. The values are handled differently depending on their type. We develop different sentence builders methods and we compared the final predicting power. To explain the differences between the language construction methods, consider the following row in the dataset:

age	workclass	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	native.country	income
39	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K

One possibility is to consider *Basic Language Construction* method that embeds the tabular information in a very structured way, where each column is presented in the format "column name: column value" and different columns are separated by a comma. For the example given above, the corresponding sentence built with this method is:

```
[metadata]. age: 39, workclass: State-gov, [...] native.country: United-States.
```

However, TabText paper authors found that formulating the sentence in a descriptive manner while adding context through processing of numerical variables is a value-adding method. Numerical data is hence processed to compute average and standard deviation, and classify values as "high", "normal", "low" based on their distance from the mean value. This information is then added in the sentences.

```
[metadata]. The age is 39 (normal), The workclass is State-gov, [...] The
native.country is United-States.
```

The advantage of this descriptive method is that it is easy to automate across different datasets as it follows a rigorous structure. For the purpose of testing the value added by building better sentences, we consider a third language construction method that sacrifices automation in favor of more expressing, tailor-made sentences. For this method, we consider pronouns and structure the sentence in a way that is more similar to how a human would write it. Furthermore, we create a mapping between the ambiguous categorical values (such as "State-gov" above) to a more meaningful value (such as "State government"). The resulting full sentence is:

```
[metadata]. A Male of age 39 (normal), born in United-States, is currently
not married and works as an Administrative Clerical for the state government.
He holds a Bachelor's degree (advanced studies), which corresponds to 13
(High) years of education. His relationship status within the family is
'not-in-family', and his race is White. He has had a capital gain of $2174
(normal) and a capital loss of $0 (normal). He typically works 40 (normal)
hours per week.
```

The limitation of this approach is that, being designed for this specific task, it is not scalable to multiple datasets and hence it fails in using the method as a black box tabular data processing tool. Acknowledging this limitation, we show that the better the sentences, the higher the added value of the method.

3.2 LLM and Fine-Tuning

We feed the sentences constructed as described above into a sentence transformer to extract meaningful embeddings. A sentence transformer model is a LLM that is specifically designed for generating embeddings of sentences that are able to capture their semantic meanings. The key advantage of sentence transformer models over traditional word embeddings is their ability to capture the context and meaning of a sentence as a whole, rather than just the individual words.

We chose the model "all-MiniLM-L6-v2" [1] as we assessed being the best tradeoff between speed, performance and model size. The model maps sentences and paragraphs to a 384 dimensional dense vector space. We access the open source model through from HuggingFace Model Hub.

3.2.1 Self-supervised masked word prediction

Aligned with the original TabText paper, we attempt to improve the quality of embeddings by fine-tuning the sentence transformer model on the constructed sentences with a self-supervised masked word prediction. This is a typical approach through which the model learns rich contextual representations of words and it understands not just the meaning of individual words but also how they are used in different contexts.

3.2.2 End-to-end fine tuning

We propose an innovative approach for end-to-end fine-tuning an attention-based transformer uncased by integrating the prediction task into the model. We chose bert-base-uncased for its versatility in finetuning on text classification, and added a dense layer for binary classification to proceed to the training phase. An important step of this fine-tuning process is the meticulous optimization of hyperparameters, which greatly vary upon the size of the training dataset.

This approach, while it may reduce the interpretability of the framework, enables the creation of embeddings that are significantly more meaningful in relation to the target variable, as illustrated in Figure 2. Another key advantage of this method is that it avoids the need to use deep learning features as inputs for the XGBoost baseline model. This is a critical improvement, as feeding deep learning features into XGBoost is not advised. By avoiding this, our method ensures a more robust and theoretically consistent approach, enhancing the overall efficacy of the model in practical applications.

3.3 Embeddings

Embeddings are a numerical representation of information that capture the semantic meaning of it. The dimension of the vectorized representation of the sentences has been kept constant and equal to 384. We leverage the statistical method t-SNE (t-Distributed Stochastic Neighbor Embedding) to reduce the dimensionality of embeddings and visually interpret the output for different models and fine-tuning.

Figure 2 shows how an LLM that has not undergone any fine tuning produces embeddings with high variance with respect to the target variable 'income'. The self-supervised mask prediction approach specifically encourages the model to understand and predict words or phrases that are masked out

in the input text. This task might not align well with preserving the original feature correlations, especially if those features were not explicitly modeled during the fine-tuning process.

The end-to-end fine tuning proved to do a much better job at producing meaningful embeddings that are able to capture both tabular and contextual information, and can be clustered according to the target variable.

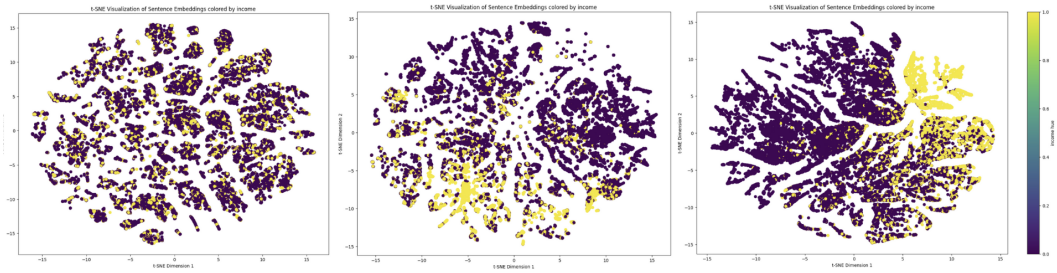

Figure 2: Embeddings visualization for model with no fine tuning (left), masked word prediction fine tuning (center), end-to-end fine tuning (right)

4 Experiments and Results

We have conducted extensive experimentations of multiple versions of the pipeline outlined above. To guarantee consistency in comparing results, we have considered a constant test set consisting of 16k rows, and each result reflects a meticulous manual and automated optimization of hyperparameters. Our objective is to test the framework for multiple language construction methods, and examine different strategies for incorporating embeddings into the prediction task. These strategies include: utilizing the embeddings independently, merging them with other features in a multimodal manner, or fine-tuning the entire model end-to-end. Additionally, we explored the impact of applying Principal Component Analysis (PCA) to the embeddings to address potential overfitting issues. Finally, we have tested the value added by the framework as a function of different training set sizes.

4.1 Baseline

We built baseline models to predict the binary target (1 if a person earns more than $50k a year and 0 if less). While developing these models, we have implemented a fair amount of feature engineering and cross validation to obtain the best possible model. Table 1 summarizes the performance of different models and highlights XGBoost as the best performing model that we choose to reference for the rest of the experiments.

Table 1: Comparison of Baseline Model Performance

Model	AUC
Logistic Regression	0.80
KNN	0.79
CART	0.80
Random Forest	0.81
XGBoost	0.81
Best Model	XGBoost

4.2 Predicting with embeddings only

As a preliminary experiment, we have tested the predicting power of embeddings by building machine learning models with a training set that is exclusively made by them.

We conducted experiments using two distinct language construction techniques: the descriptive sentence builder and the tailor-made builder, each offering unique insights. The first key finding was that the tailor-made sentences resulted in embeddings that had superior predictive power compared to those generated from the descriptive sentences. Furthermore, we discovered that applying Principal Component Analysis (PCA) to these embedded vectors improved the model's performance when dealing with new, unseen data. This improvement is due to the reduction in overfitting, inevitable with the original 384-dimensional input vector.

Lastly, and perhaps most notably, was the unexpectedly high performance of the models that relied solely on embeddings to train the XGBoost model. While these results still fell short of our baseline benchmarks, they were impressively high (refer to Table 2 for detailed figures). This outcome showcases the potential and robustness of the overall framework for processing and prediction accuracy.

Table 2: Comparison of Model Performance

	NO PCA	3D PCA	5D PCA
Descriptive sentences	0.760	0.760	0.762
Tailor-made sentences	0.764	0.768	0.771

4.3 Tabtext replica

By fine-tuning the sentence transformer with a self-supervised masked word prediction using the descriptive sentences as training data, we essentially replicated the best-practices found with the extensive experiments underlying the original TabText paper. We consider the added value of the framework built this way the core benchmark in our experiments, and our implementation has led the AUC out-of-sample performance to an increase of 0.02.

Table 3: Out-of-sample performance for different dimensionality reductions

Model	AUC	vs Baseline
Baseline	0.81	-
TabText Replica No PCA	0.81	- 0.00
TabText Replica 7D PCA	0.82	+ 0.01
TabText Replica 5D PCA	0.82	+ 0.01
TabText Replica 3D PCA	0.83	+ 0.02

4.4 End-to-end fine tuning

The most significant experiment of our effort is the end-to-end fine tuning of the LLM to directly predict the target variable. Such method allowed for the generation of much more information-rich embeddings, that can be clustered with ease into the two target classes. Remarkably, this method yielded an AUC of 0.92, a substantial improvement over our baseline model's AUC of 0.81. This increase is not just numerically significant but also indicates a considerable enhancement in the model's predictive accuracy and reliability.

4.5 Experimenting with different training size

As a final experiment, we have tested the value added by the framework for different training set sizes. The authors of TabText found evidence of a higher added value when the predictive task is harder and/or when the training set is smaller. As described in Table 4, we did not find any significant evidence to confirm this statement on the income dataset. The traditional approach to the TabText pipeline showed little to no benefit when applied on smaller data samples made by 100 to 1000 samples. Even making use of end-to-end fine tuning, which on the larger training set bring an added AUC performance of 0.11, the improvement was minimal (+0.02). This can be attributed to the data size dependency of the deep learning training process that underlies this type of fine tuning. The missing values in Table 2 are due to the computational challenges given by the end-to-end approach.

Table 4: Comparison of Model Performance

Training Set Size	No fine-tuning	Masked fine-tuning	End-to-end fine-tuning
100	+ 0.00	- 0.02	+ 0.02
500	+ 0.00	+ 0.01	n/a
1000	- 0.02	+ 0.00	n/a
10000	- 0.01	+ 0.01	n/a
32561	+ 0.01	+ 0.02	+ 0.11

5 Discussion

We have observed a significant improvement in out of sample performance when the sentences were generated with a tailor made strategy. The downside of such approach is the low generalization capabilities, as it is hard to immediately extend to different datasets and column names. Hence, we highlight a trade-off between flexibility and performance: a less generalizable language format carries more information and predictive power.

The contextual representation framework can both be seen as a low-code data prepossessing tool, and as a performance booster. Under the former perspective, we have shown how a fine-tuned LLM is able to effectively represent both the context and the tabular value, potentially allowing the replacement of code-intensive data engineering with natural language and expert knowledge. Under certain conditions, the framework has also shown the potential to be a performance booster. However, the implementation of TabText in its various form has not always led to an increased in performance when compared to the baseline. In particular, our experiments shows little to no benefit when the LLM is not fine tuned and when the training size is not large enough to make the LLM learn enough. Nonetheless, he out-of-sample performance improvement brought by end-to-end fine tuning an LLM (+0.11 AUC) is a remarkable result.

5.1 Future work

We plan to keep working to extend the progress made so far. First we want to enhance the interpretability of the methodology. We have already improved the visibility by carrying out an exploratory data analysis aimed at visualizing the generated embeddings. This was particularly useful in understanding the underlying reasons of such a large performance delta between different fine tuning methods.

We plan to test the robustification of the pipeline with respect to the sentences format. Specifically, we plan to train the pipeline with sentences generated from several different sentence builders. This method would train the model to focus on the signal rather than the format of the sentences. This would be powerful both in terms of robustness and user experience. An end user could simply provide a sentence in plain english, either in text or audio format, and receive the model's output with no further effort.

6 Authors contribution

Both authors of the report have collectively pursued the initial research of the project. Benjamin first proposed the idea of working on the topic, and we have initiated a conversation with two of the authors of the original TabText paper. Given the fairly theoretical aspect of the project, a significant amount of collective effort has gone into brainstorming and theoretical reasoning. To approach this problem, we drew inspiration from a methodological framework presented in TabText, leveraging a tree to communicate experimental decisions.

We have challenged each other in building the best performing baseline. Benjamin won the challenge by delivering the benchmark made by an AUC of 0.81. Further development of the Machine Learning model were carried out by Giorgio, including the testing of the multimodal implementation and the embeddings predictive power evaluation.

Benjamin leveraged his expertise of the HuggingFace Hub to lead the LLM-facing programming including the embeddings generation and end-to-end fine tuning. Giorgio carried out the self-supervised masked word prediction fine tuning, as well as the multimodal prediction pipeline.

References

[1] *all-MiniLM-L6-v2*. `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2#all-minilm-l6-v2`.

[2] Kimberly Villalobos Carballo et al. *TabText: A Flexible and Contextual Approach to Tabular Data Representation*. 2023. arXiv: `2206.10381 [cs.LG]`.

[3] Ron Kohavi. *Census Income*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5GP7S. 1996.

[4] Wenxuan Zhou et al. *Context-faithful Prompting for Large Language Models*. 2023. arXiv: `2303.11315 [cs.CL]`.