Analysis of the Attention in Tabular Language Models

Aneta Koleva
Siemens AG, LMU,
Munich, Germany
aneta.koleva@siemens.com

Martin Ringsquandl
Siemens AG
Munich, Germany
martin.ringsquandl@siemens.com

Volker Tresp
Siemens AG, LMU,
Munich, Germany
volker.tresp@siemens.com

Abstract

Recent transformer-based models for learning table representation have reported state-of-the-art results for different tasks such as table understanding, question answering and semantic parsing. The various proposed models use different architectures, specifically different attention mechanisms. In this paper, we analyze and compare the attention mechanisms used by two different tabular language models. By visualizing the attention maps of the models, we shed a light on the different patterns that the models exhibit. With our analysis on the aggregate attention over two tabular datasets, we provide insights which might help towards building more efficient models tailored for table representation learning.

1 Introduction

Tables are a commonly used structures for storing and representing information, found in abundance across different databases, industrial spreadsheets and web pages. The availability of large, publicly accessible tabular datasets [2, 9, 16] motivated the research community to investigate and develop models for learning latent table representation, which can be used for variety of tasks such as: table question answering [23], semantic parsing [8, 25], table search [19], semantic annotation of tables [10, 11] and other subtasks towards table interpretation [4, 22].

Several aspects of tabular data distinguish tables as a different type of modality from text; tables have an inherit structure with a header, rows and columns, they often store only numerical values and the information stored in tables does not follow the common rules of grammar, i.e., it is without a logical meaning in contrast to a sentence. The challenge is to leverage the table characteristics and the capabilities of the existing language models (LMs) [5, 15], while designing models tailored for learning table representation. Consequently, in addition to the masked language model (MLM) objective used in BERT, the proposed tabular language models (TaLMs) rely on table-specific pre-training objectives, such as masked column prediction (MCP) [25], masked entity recovery (MER) [4] and masked cell recovery [8, 23]. Two recent surveys provided a detailed overview of the different architectures and pre-training objectives used by the recent tabular language models (TaLMs) [1, 6].

Following the success of the Transformer model [18], the research community is focused towards understanding the attention mechanism that is at the core of this model. Several works on attention in the BERT model share the findings that the deeper layers of the model focus on specific part-of-speech tags (such as nouns, verbs) [14, 21], target longer-distance relationships and therefore require broader context [14, 20]. Visualizing the attention maps of pre-trained LMs provides insights about the

Table Representation Learning Workshop at NeurIPS 2022

different patterns that the models exhibit and the semantic as well as syntactic knowledge that these models capture [3, 17, 20].

Inspired by this line of work, we reuse and adapt the existing methods for analyzing the attention from LMs to the TaLMs. Our analysis focuses on the attention from TaBERT [25] and TURL [4]. These two models use fundamentally different approaches for leveraging the table structure. TaBERT enforces the table structure by adapting the input structure using the so called special token [SEP], while TURL uses *visibility matrix* for restricting the attention mechanism.

Furthermore, we explore the influence of the input data to the attention maps of the models by analyzing the aggregate attention over two essentially different datasets. The first dataset, T2D [16], consists of manually annotated tables with mostly textual data. The second, GitTables [9], consists of tables extracted from online resources with mostly numerical values. Our analysis shows that while the size of the input data does not significantly affect the behaviour of the attention, the content of the data certainly does. Interestingly, when present in the input, the special tokens *steal* all of the attention. With our analysis we aim to answer the following questions:

- What is the effect of the restricted input vs the restricted attention map?
- Does the size of the input affect the behaviour of the attention?
- Is the attention from the header cells to the body cells significant?
- How much attention do the special tokens receive?

2 Preliminaries

In this section we formalize the notion of table, describe the transformer encoder and give an overview of the self-attention mechanisms used in TaLMs.

2.1 Table

We formally define a table as a tuple $T = (C, H)$, where $C = \{c_{1,1}, c_{1,2}, \ldots, c_{i,j}, \ldots, c_{n,m}\}$ is the set of table body cells for n rows and m columns. Every cell $c_{i,j} = \left(t_{c_{i,j},1}, t_{c_{i,j},2}, \ldots, t_{c_{i,j},k}\right)$ is a sequence of tokens of length k. The table header $H = \{h_1, h_2, \ldots, h_m\}$ is the set of corresponding m column header cells, where $h_j = \left(t_{h_j,1}, t_{h_j,2}, \ldots, t_{h_j,l}\right)$ is a sequence of header tokens of length l.

2.2 Transformer Encoder

The encoder block from the Transformer[18], which is used in pre-trained LMs, is also the backbone of the most recent TaLMs. The input to the Transformer encoder is a sequence of vectors $(x_1, x_2, \ldots, x_n)$, corresponding to n tokens from the input sequence. The goal is to learn a contextualized representation for the vectors by passing them through a stack of L identical layers: each with H self-attention heads followed by a position-wise feed forward layer.

Through separate linear transformation, every vector x_i is transformed into a query, key and value vectors, q_i, k_i and v_i. The self-attention mechanism, also known as Scaled Dot-Product Attention, assigns an attention weight of a target token x_i with respect to every other token in the input sequence:

$$\text{Attention}\left(x_i\right) = \left(\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,j}\right) \tag{1}$$

where $\alpha_{i,j}$ is the attention weight that token x_i pays to token x_j and $i, j \in [1, \ldots, n]$, where n is the total number of tokens in the input sequence. The attention weight $\alpha_{i,j}$ is calculated as the dot product of the query vector q_i with the key vectors k_j, followed by the softmax operation. The output of the attention head is the weighted sum of the values:

$$\alpha = \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2}$$

where Q, K and V are the query, key and values matrices respectively, and $\frac{1}{\sqrt{d_k}}$ is a scaling factor with d_k the dimension of K.

Intuitively, this matrix represents how much a token attends to the other tokens from the input sequence, when calculating the next representation for the current token.

By using several attention heads in parallel, the model is able to jointly attend to information at different positions. The multi-head attention, is the concatenation of the independently calculated attention heads, each calculated with Eq. 2 :

$$\begin{aligned} \text{MultiHead}(Q, K, V) &= \text{Concat}(\text{head}_1, \ldots, \text{head}_H)W^O \\ \text{head}_\text{i} &= \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \end{aligned}$$

where the parameter matrices W_i^Q, W_i^K, W_i^V and W^O are learned during the training process.

2.3 Overview of Attention in TaLMs

The attention mechanism is crucial in the computation of the contextual representation of the input. However, the attention mechanism presented in the original Transformer paper [18] is intended to be used on a sequential input (e.g., a sentence). This requires a lot of computations, since every element from the input sequence attends to every other element. A table represents a different type of modality, therefore it can be beneficial to modify the attention mechanism such that it reflects the structure of the table. In the search for a more efficient and more narrow attention mechanism, different adaptations have been proposed.

TaBERT [25] encoded a table row-wise with the self-attention mechanism as presented in the Transformer [18] and then stacked these representations vertically. A column-wise self-attention is then applied to obtain contextualized representation for the columns. TAPAS [8] only used the Transformer self-attention with additional embedding which indicate the row and column identity of the tokens. MATE [7] is an improved and more efficient version of TAPAS with sparse-attention which restricted the attending heads to tokens only from the same row (row heads) or to tokens from the same columns (column heads). TUTA [23] proposed a tree structure to represent cell coordinates and capture the distance between the cells in a table. They used a distance-aware attention with tree-based positional encoding and restricted the attention only to tokens from neighboring cell within a defined distance range. TURL [4] introduced a *visibility matrix* to restrict the attention from the current token to tokens from the same row or the same column. The more recent TableFormer model [24], used a set of predefined attention-biases as learnable scalars which were added as a bias-term to the self-attention module.

In Table 1 we summarize the different attention mechanisms used in the existing TaLMs. We refer to the self-attention mechanism from [18] as the Transformer attention. Additionally, we show the number of attention layers, L, and the number of attention heads, H. For a more detailed overview and in-depth analysis of the complete model architectures and pre-training objectives, we refer the readers to the survey paper by Dong et al. [6].

Table 1: Overview of the different attention mechanisms used in the models.

Model	**Attention**	L	H
TAPAS	Transformer attention	12	12
TaBERT	Transformer attention + vertical on the columns	3	6
TURL	Restricted to tokens in the same row/column + header	4	12
TUTA	Joint bi-tree-based. Focused on spatial and hierarchical info	4	12
MATE	Row/column restricted attention heads	12	3
TableFormer	Added attention-bias to each attention head	12	12

3 Analyzing Attention

In the focus of our analysis are two fundamentally different TaLMs. We first discuss in detail the inputs and the architecture of the attention mechanism in both of the models. We then present a visualization of one attention head for an example input table. After that we explore different aspects of the aggregate attention from the models over two different datasets.

3.1 Models

The models that we analyse were developed and optimized for solving different problems, therefore, it is difficult to make a fair comparison and decide which and what would be better. Instead, our goal is to give an overview of their different inputs and architectures and expose the differences in their attention maps.

TaBERT [25] is one of the first TaLMs which has been pre-trained on tabular data. With the intention to be used as a neural semantic parser, the model gets as input a question and a table with the possible answer, and outputs a contextualized vector representation of the question tokens and the columns of the table. TaBERT lineraizes every table row as a sequence of the cells and passes this as input to the BERT model. This sequence always starts with the [CLS] special token and ends with the [SEP] special token. Every cell is represented as the concatenation of the column name, column type and value, separated from the other cells with [SEP]. For example, the value of the first cell from the example table in Figure 1 is represented as:

$$\text{[CLS][SEP]} \quad \underbrace{\text{name}}_{\text{Column Name}} \;|\; \underbrace{\text{text}}_{\text{Column Type}} \;|\; \underbrace{\text{michael schumacher}}_{\text{Cell Value}} \;\text{[SEP]}$$

Using BERT, a row-wise representation of every row is generated. First, a cell-level pooling is done in order to generate a cell representation. Then, a column-wise vertical self-attention layers are applied and the final representation for the columns are generated. In the TaBERT model, the base BERT model is used as the initial row-encoder, with $L = 12$ and $H = 12$. The vertical self-attention mechanism is with $L = 3$ layers, each with $H = 6$ attention heads. We want to analyse the attention that the different types of tokens receive. Therefore, we extracted the attention weights from the BERT layers instead of the vertical self-attention layers. It is important to note that these layers are part of the tabular pre-trained model.

TURL [4] is a TaLM used for various tasks towards table understanding such as: entity linking, column type annotation, relation extraction, row population, cell filling and schema augmentation.

In TURL, a cell level pooling is done during the initial preprocessing, when the body cells are represented as the mean average of their token embeddings. The embedding representation for a body cell $c_{i,j}$ is referred to as the entity representation: $e^{c_{i,j}} = \text{MEAN}(e^{t_1}, e^{t_2}, \ldots, e^{t_k})$. In contrast, the header tokens are encoded separately and referred to as the token representations. The embedding representation for a header cell h_j is $e^{h_j} = (e^{t_{h_j,1}}, e^{t_{h_j,2}}, \ldots, e^{t_{h_j,l}})$. A sequence of token and entity embeddings, together with an attention mask matrix, are the inputs to self-attention of TURL.

The attention mask that TURL used, is called a *visibility matrix*, and is a mechanism for restricting the attention from the tokens. The visibility matrix allows every body cell to attend only to tokens from the same row or from the same column, with the column header tokens being visible to all cells belonging to the particular column. For example, while generating a representation for the first body cell in the example table from Figure 1, only the tokens from the first header cell are visible, while the rest of the headers are masked. Additionally, TURL generates the contextualized representations for the header tokens, and the body cells separately, i.e., there are 2 attention matrices produced by the self-attention mechanism. In our analysis we used the attention weights for the entity cells.

3.2 Visualizing an individual input

We now present the visualization of the attention weights for a single table. An interactive view of the full attention maps for both models is available online [1]. The visualizations have been created with the tool BertViz [20], which was developed for the analysis of the attention in BERT. Figure 1 shows an example table which we used as input to both TaBERT and TURL. The attention weights between the attending tokens on the left to the attended tokens on the right are represented with lines, where the thickness of each line indicates the strength of the weight.

In this visualization we observe the difference between the inputs of the models. In the left picture, we see the input sequence to TaBERT which includes the special tokens [CLS] and [SEP] and the delimiter sign |. Additionally, the name of the column and the type of the column are added to the

[1] https://github.com/anetakoleva/attention_analysis_TaLMs

Name	Auto Racing Team	Formula1 Race	Year
Michael Schumacher	Scuderia Ferrari	2005 United States Grand Prix	2005

Figure 1: Attention from layer 2, head 9, from TaBERT (left) and TURL (right) for the input table.

content of every body cell. In the right picture, we see the simpler input sequence that is passed to TURL. The header cells are tokenized, while the body cells are observed as one token (an entity view where an entity is the mean representation of its tokens).

We note that the attention weights in TaBERT are much more broad, with stronger weights on the special tokens, while in TURL we notice the effect of the restricted attention, so that every body cell attends only to its column name.

3.3 Analysis of the aggregate attention

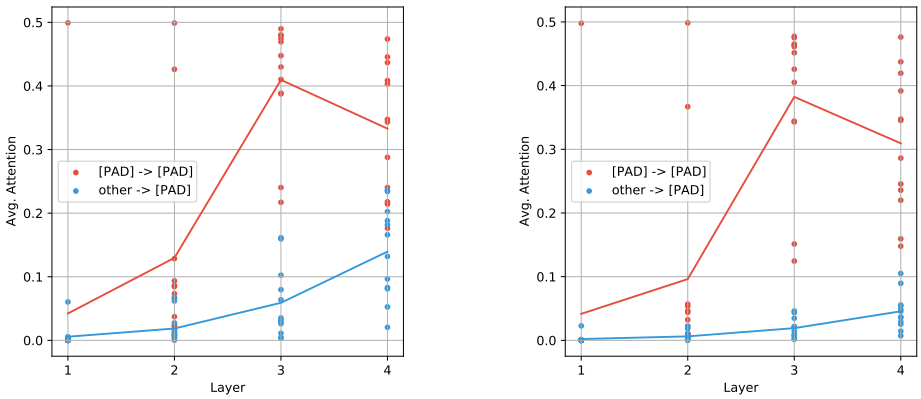
Following the analysis done in [3] for BERT, we explore aspects of the aggregate attention extracted from encoding corpus of tables with the TaLMs. Since for TaBERT we can extract and visualize maps only for tables with 1 row, we report our analysis for both TaBERT and TURL for tables from which we sampled $n = 1$ row. Additionally, to investigate if the size of the dataset would impact the behaviour of the attention, we also conducted our analysis on tables with $n = 3$ and $n = 5$ rows with the TURL model. These findings are presented in the Appendix A of the paper.

To investigate if the content of the input data affects the behaviour of the attention mechanism, we used two different datasets, T2D [16] and GitTables [9]. The first dataset consists of tables extracted from the Web Data Commons [13] with mostly textual data. The second dataset contains tables extracted from Github pages, with many numerical values. From the GitTables dataset we only keep those columns in the tables for which there is a DBpedia [12] column annotation available.

3.3.1 Attention to the special tokens

In BERT, there are special tokens with specific roles; [CLS] - used for classification predictions and always added at the beginning of an input sequence, [SEP] - the separator token is used as a deliminator between 2 input sequences and [PAD] - the padding token. We explore how different attention heads behave with respect to these tokens, i.e., how much attention is payed to these tokens.

The experiments with the two datasets and the sampling of $n \in [1, 3, 5]$ rows for the tables, revealed a clear pattern in the attention of TURL to the [PAD] token. In Figure 2 we show the average attention of each head towards the [PAD] token across the 4 layers. This token receives less than half of the



(a) GitTables (b) T2D

Figure 2: Average attention to the [PAD] token from TURL for the GitTables and T2D datasets.

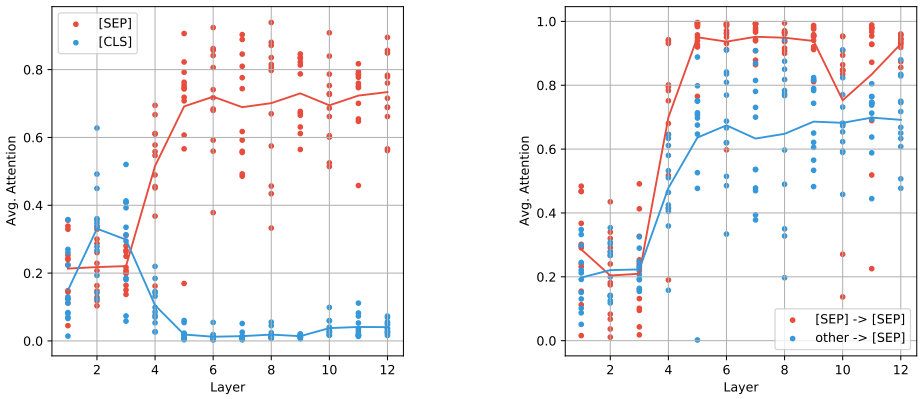


Figure 3: Average attention to the [CLS] and [SEP] tokens from TaBERT for the GitTables dataset. Attention to the [SEP] token is higher when the current token is [SEP].

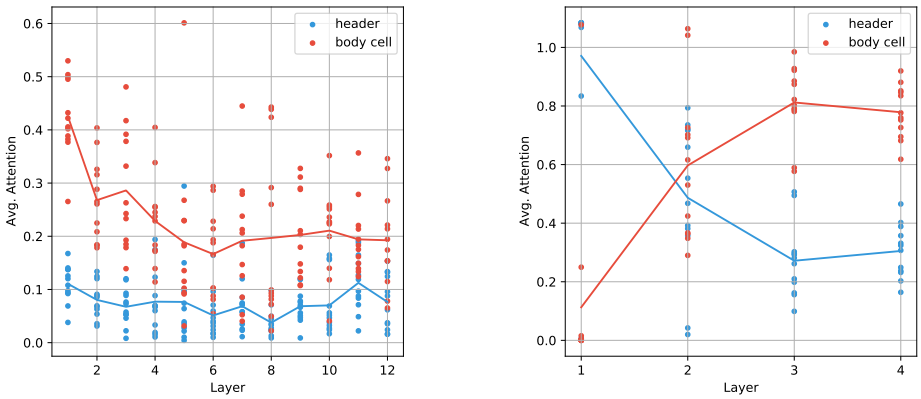
attention, and the highest average attention weight this token receives when the current token is also the token [PAD].

Similarly as in the BERT analysis [3], we see that large amount of the TaBERT's attention focuses only on the [CLS] and [SEP] tokens. While in the first 4 layers, the attention is on the [CLS] token, after the 5th layer substantial amount of attention is directed to the [SEP] token. Moreover, in Figure 3 (on the right), we see that although the attention is stronger when the current token is [SEP], more than half of the attention from the other tokens in the input is also directed to this token. In [3] the authors hypothesize that this token is used as a *no-op* token, which means that it is seen as a signal to the model to ignore this attention head as it does not find a meaningful pattern.

Since in the input sequence, all of the body cells are divided with the [SEP] token, the number of the [SEP] tokens in the input sequence is equal to the number of columns in the table plus 2 ([SEP] is also added at the beginning and at the end of the sequence. Separating all the cells with the [SEP] token might be harming the model's ability to focus to the rest of the tokens.

3.3.2 Attention to the header/body tokens

To analyze the effect that the column names have on the learned representation of the body cells, we compute the average attention that the header tokens and the body cells receive.



(a) TaBERT (b) TURL

Figure 4: Average attention to the header and body tokens for the GitTables dataset.

The average body cell attention of one attention head is calculated as $\frac{1}{k}\sum_i \sum_{j \in c} \alpha_{i,j}$, where $c \in C$ is a body cell, $i \in [0, 1, \ldots, n]$ where n is total number of tokens and k is the number of body tokens. The attention to the body tokens from all of the tokens in the table, $\alpha_{i,j}$, divided by the number of body tokens.

Similarly, the average header attention of one attention head is calculated as $\frac{1}{l}\sum_i \sum_{j \in h} \alpha_{i,j}$ where $h \in H$ is a header token and l is the number of header tokens. The results for both datasets look very similar, therefore here we analyze only the aggregate attention over the GitTables dataset, and we add to the Appendix A the T2D results.

Figure 9 (b) shows that for TURL the average attention to the header is highest in the first layer and then it decreases, while the average attention to the body cells after the first layer increases. Both types of tokens receive equal amount of attention and the model is not fixed only on the header or on the body cell. The body cell tokens attend to the header tokens and vice versa.

Figure 9 (a) shows that for TaBERT the average attention towards the header cells is significantly low (around 0.1) and it does not increase in any of the 12 layers. Intuitively, this means that the header cells have very small impact to the contextualized representations of the body cells. The average attention towards the body cells is higher in the first layer and after that it steadies at around 0.2 which is not a high attention weight.

3.3.3 Attention Entropy

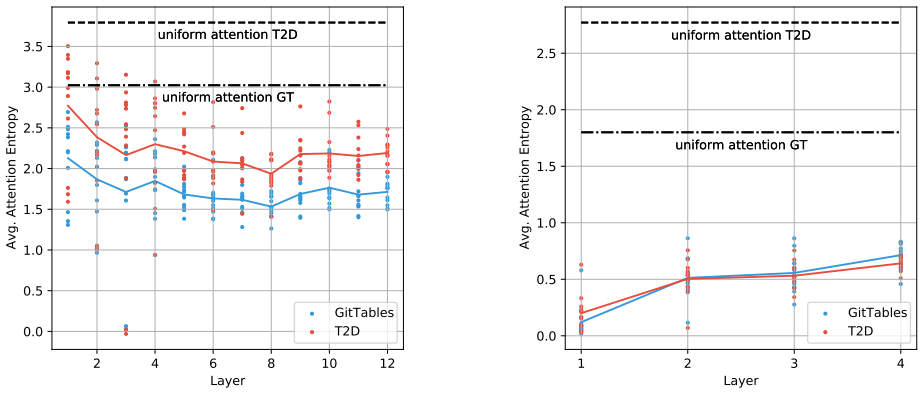
We measure the attention dispersion to see if the attention is more focused on particular words or it attends broadly over the input sequence. We measure the attention dispersion across the layers based on the entropy of the attention distribution.

$$Entropy_{\alpha}\left(x_i\right) = -\sum_{j=1}^{i} \alpha_{i,j} \log\left(\alpha_{i,j}\right)$$

When the pre-defined tokens are part of the input sequence, the attention tends to be more focused on these tokens and previous studies [3, 20] have calculated the attention entropy excluding these tokens. However, these tokens are used to integrate the table structure in the input sequence provided by TaBERT, therefore, we can not exclude them from the calculation.

We observe that the entropy for both datasets is very similar for TURL (Figure 5 (b)). Across the 4 layers the entropy remains very low, under 1, indicating that the attention is focused on several words.

In the case of TaBERT, Figure 5 (a), shows that the GitTables dataset is overall with lower entropy. For both datasets, in the first layer there are particularly high-entropy heads which attend very broadly



(a) TaBERT

(b) TURL

Figure 5: Entropies of the attention distribution for both GitTables and T2D datasets.

and even though the entropy is lower for the middle layers (5-9), it is still notably higher compared to TURL. This reveals that the attention in TaBERT is overall much more dispersed.

4 Conclusion

We have presented a series of analysis for gaining insights into the different self-attention mechanisms used by two TaLMs. Using visualization maps we offer a view of the different inputs of the models and how the attention is divided between the elements of the input sequence. With the analysis on the aggregate attention we reveal the discrepancy in the attention payed to the header and to the body cells between the two models. The higher average attention in TURL towards the header tokens means that the body cells attend to the header tokens, which would be beneficial for solving tasks such as the column type annotation. This observation indicates that the restriction on the attention mechanism used by TURL, compared to the extended input enforced by TaBERT, might be more effective for navigating the attention between the body and the header tokens. Additionally, we showed that considerable amount of attention from TaBERT is payed to the special tokens. Since these tokens may be used as a no-op for the model and more than half of the 12 layers in TaBERT focus exclusively on these tokens, the question of how many attention layers from BERT are needed when building efficient TaLMs raises. To establish if and how much the attention influences the output from the two analysed models, experiments on a commonly defined task are required.

References

[1] Gilbert Badaro and Paolo Papotti. "Transformers for Tabular Data Representation: A Tutorial on Models and Applications". In: *Proc. VLDB Endow.* 15.12 (2022), pp. 3746–3749. URL: https://www.vldb.org/pvldb/vol15/p3746-badaro.pdf.

[2] Michael J. Cafarella et al. "WebTables: exploring the power of tables on the web". In: *VLDB* (2008).

[3] Kevin Clark et al. "What Does BERT Look at? An Analysis of BERT's Attention". In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*. Association for Computational Linguistics, 2019, pp. 276–286. DOI: 10.18653/v1/W19-4828. URL: https://doi.org/10.18653/v1/W19-4828.

[4] Xiang Deng et al. "TURL: Table Understanding through Representation Learning". In: *Proc. VLDB Endow.* 14.3 (2020), 307–319.

[5] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: https://doi.org/10.18653/v1/n19-1423.

[6] Haoyu Dong et al. "Table Pre-training: A Survey on Model Architectures, Pre-training Objectives, and Downstream Tasks". In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*. ijcai.org, 2022, pp. 5426–5435. DOI: 10.24963/ijcai.2022/761. URL: https://doi.org/10.24963/ijcai.2022/761.

[7] Julian Eisenschlos et al. "MATE: Multi-view Attention for Table Transformer Efficiency". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*. Association for Computational Linguistics, 2021, pp. 7606–7619. DOI: 10.18653/v1/2021.emnlp-main.600. URL: https://doi.org/10.18653/v1/2021.emnlp-main.600.

[8] Jonathan Herzig et al. "TaPas: Weakly Supervised Table Parsing via Pre-training". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 2020, pp. 4320–4333. DOI: 10.18653/v1/2020.acl-main.398. URL: https://doi.org/10.18653/v1/2020.acl-main.398.

[9] Madelon Hulsebos, Çağatay Demiralp, and Paul Groth. "GitTables: A Large-Scale Corpus of Relational Tables". In: *arXiv preprint arXiv:2106.07258* (2021). URL: https://arxiv.org/abs/2106.07258.

[10] Viet-Phi Huynh et al. "DAGOBAH: Enhanced Scoring Algorithms for Scalable Annotations of Tabular Data". In: *SemTab@ISWC*. 2020.

[11] Ernesto Jiménez-Ruiz et al. "SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems". In: *ESWC*. 2020.

[12] Jens Lehmann et al. "DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia". In: *Semantic Web Journal* 6.2 (2015), pp. 167–195. URL: http://jens-lehmann.org/files/2015/swj_dbpedia.pdf.

[13] Oliver Lehmberg et al. "A Large Public Corpus of Web Tables Containing Time and Context Metadata". In: *Proceedings of the 25th International Conference Companion on World Wide Web*. WWW '16 Companion. Montréal, Québec, Canada: International World Wide Web Conferences Steering Committee, 2016, 75–76. ISBN: 9781450341448. DOI: 10.1145/2872518.2889386. URL: https://doi.org/10.1145/2872518.2889386.

[14] Nelson F. Liu et al. "Linguistic Knowledge and Transferability of Contextual Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, 2019, pp. 1073–1094. DOI: 10.18653/v1/n19-1112. URL: https://doi.org/10.18653/v1/n19-1112.

[15] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: (2019).

[16] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. "Matching HTML Tables to DBpedia". In: *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS 2015, Larnaca, Cyprus, July 13-15, 2015*. ACM, 2015, 10:1–10:6. DOI: 10.1145/2797115.2797118. URL: https://doi.org/10.1145/2797115.2797118.

[17] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. "A Primer in BERTology: What We Know About How BERT Works". In: *Trans. Assoc. Comput. Linguistics* 8 (2020), pp. 842–866. DOI: 10.1162/tacl_a_00349. URL: https://doi.org/10.1162/tacl_a_00349.

[18] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[19] Petros Venetis et al. "Recovering Semantics of Tables on the Web". In: *VLDB* (2011).

[20] Jesse Vig and Yonatan Belinkov. "Analyzing the Structure of Attention in a Transformer Language Model". In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@ACL 2019, Florence, Italy, August 1, 2019*. Association for Computational Linguistics, 2019, pp. 63–76. DOI: 10.18653/v1/W19-4808. URL: https://doi.org/10.18653/v1/W19-4808.

[21] Elena Voita et al. "Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned". In: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 2019, pp. 5797–5808. DOI: 10.18653/v1/p19-1580. URL: https://doi.org/10.18653/v1/p19-1580.

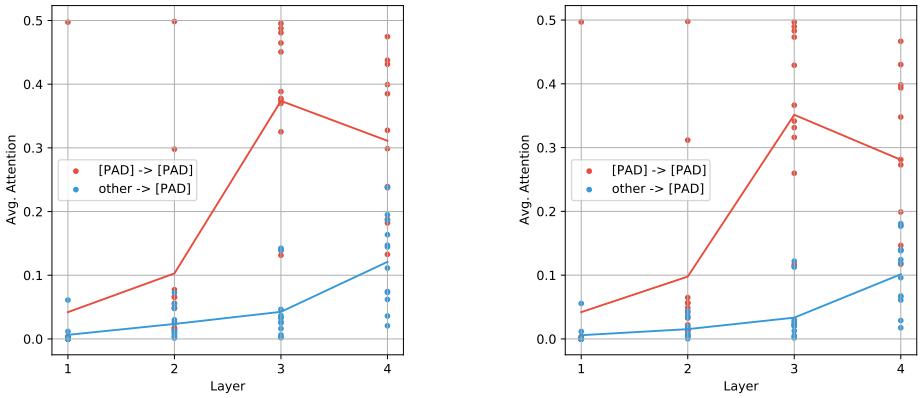
[22] Daheng Wang et al. "TCN: Table Convolutional Network for Web Table Interpretation". In: *WWW*. 2021.

[23] Zhiruo Wang et al. "TUTA: Tree-based Transformers for Generally Structured Table Pre-training". In: *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*. ACM, 2021, pp. 1780–1790. DOI: 10.1145/3447548.3467434. URL: https://doi.org/10.1145/3447548.3467434.

[24] Jingfeng Yang et al. "TableFormer: Robust Transformer Modeling for Table-Text Encoding". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 2022, pp. 528–537. DOI: 10.18653/v1/2022.acl-long.40. URL: https://doi.org/10.18653/v1/2022.acl-long.40.

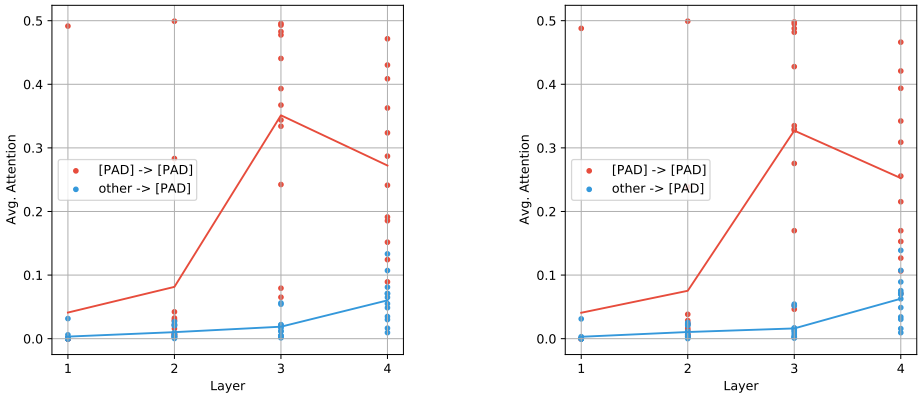
[25] Pengcheng Yin et al. "TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 2020, pp. 8413–8426.

A Appendix



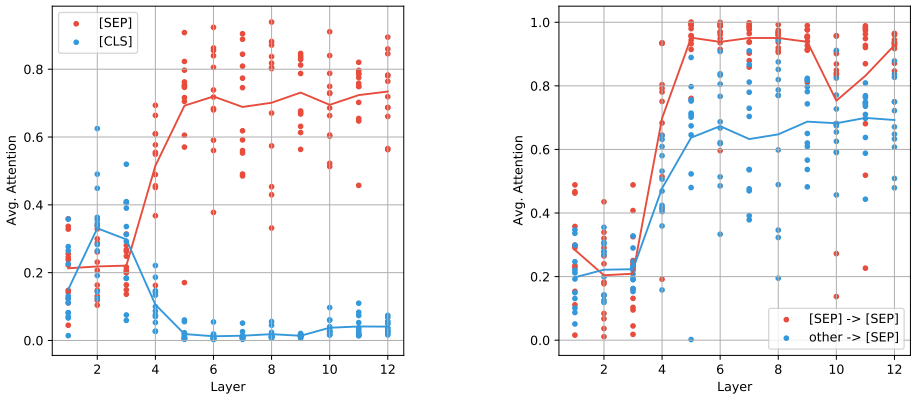
(a) GitTables, n=3 (b) GitTables, n=5

Figure 6: Average attention to the [PAD] token from TURL for the GitTables dataset with n number of rows.



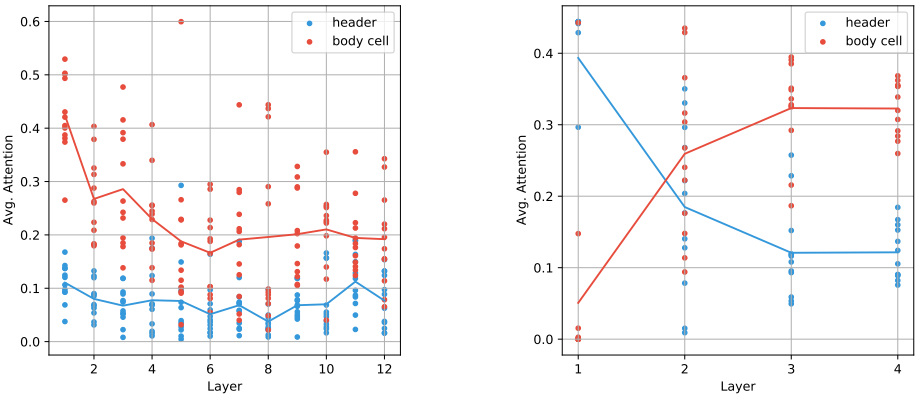
(a) T2D, n=3 (b) T2D, n=5

Figure 7: Average attention to the [PAD] token from TURL for the T2D dataset with n number of rows.



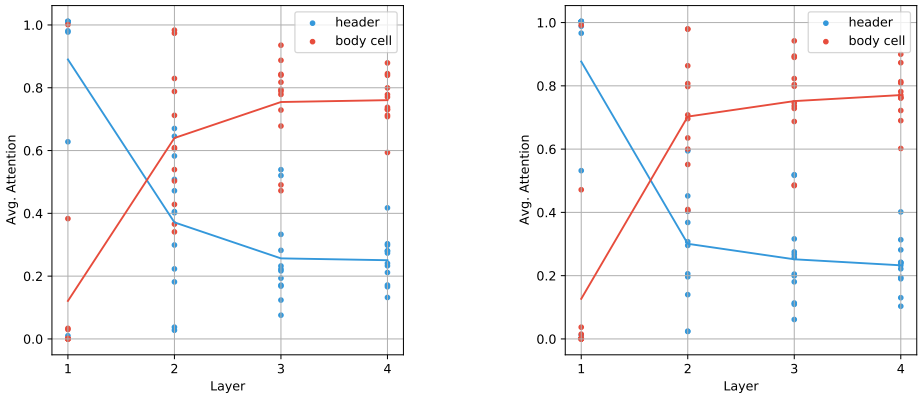
(a) T2D, n=1 (b) T2D, n=1

Figure 8: Average attention to the special tokens from TaBERT for the T2D dataset.



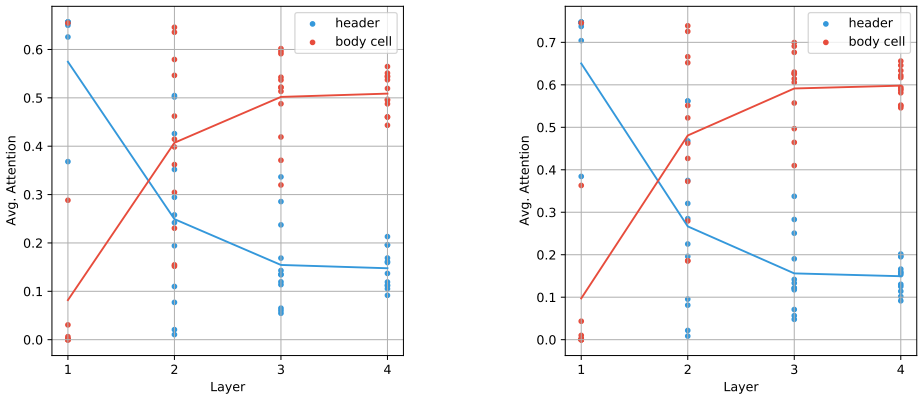
(a) TaBERT (b) TURL

Figure 9: Average attention to the header and body tokens for the T2D dataset.



(a) GitTables, n=3 (b) GitTables, n=5

Figure 10: Average attention to the header/body tokens from TURL for the GitTables dataset with n number of rows.



(a) T2D, n=3 (b) T2D, n=5

Figure 11: Average attention to the header/body tokens from TURL for the T2D dataset with n number of rows.