Adversarial Attacks on Tables with Entity Swap

Aneta Koleva¹,², Martin Ringsquandl¹ and Volker Tresp²

¹Siemens AG
²Ludwig Maximilian University of Munich

Abstract
The capabilities of large language models (LLMs) have been successfully applied in the context of table representation learning. The recently proposed tabular language models (TaLMs) have reported state-of-the-art results across various tasks for table interpretation. However, a closer look into the datasets commonly used for evaluation reveals an entity leakage from the train set into the test set. Motivated by this observation, we explore adversarial attacks that represent a more realistic inference setup. Adversarial attacks on text have been shown to greatly affect the performance of LLMs, but currently, there are no attacks targeting TaLMs. In this paper, we propose an evasive entity-swap attack for the column type annotation (CTA) task. Our CTA attack is the first black-box attack on tables, where we employ a similarity-based sampling strategy to generate adversarial examples. The experimental results show that the proposed attack generates up to a 70% drop in performance.

Keywords
Column Type Annotation, Adversarial Attack, Table Representation Learning

1. Introduction
Following the advancements of large language models (LLMs) in NLP, tabular language models (TaLMs) have emerged as state-of-the-art approaches to solve table interpretation (TI) tasks, such as table-to-class annotation [1], entity linking [2], and column type annotation (CTA) [3, 4, 5]. Similar to other deep neural networks, LLMs are sensitive to small input perturbation, which as adversarial examples can further be optimized to be imperceptible to humans [6]. Several works have studied adversarial attacks on LLMs, and it is becoming an increasingly important topic as LLMs are vastly being integrated into applications [7]. For the table modality, so far, sensitivity to perturbations has not been investigated in TaLMs for TI tasks. Hence, it is unclear which perturbation operations should be considered when attacking tables and how to make them imperceptible.

Using CTA as an example task, we phrase the novel problem of generating adversarial examples in entity tables. The existing models already report very high F1 scores on this task, and it is hard to judge by the performance of the model, how well it can generalize to unseen novel entities. In this direction, we design an evasive entity-swap attack that is motivated by a problem we observed. Namely, in two datasets commonly used for evaluation of the CTA task, WikiTables [3] and VizNet [4], there is a data leakage from entities from the training set into the test set.

In Table 1 we show the percentage of overlapped entities between the train and test set in the WikiTables dataset for the top 5 classes. The last 15 types in this dataset have 100 overlap among entities. In Figure 1, both of the first two tables have a column named Player which contains the exact same set of entities, and is annotated with the same semantic types Athlete and Person. The third table shows an example of an adversarial table with an entity swap. In this table, the entities of the column Player are swapped with new, unseen entities of the same semantic type.

In the evaluation, we gradually increase the percentage of entities that we swap in the targeted columns, ranging from 20 % up to 100 % percent of the number of entities in the column. For choosing the adversarial entities, we propose a similarity-based strategy and compare it to sampling at random. Our evaluation demonstrates that swapping entities with the most dissimilar entity of the same type results in a substantial drop in performance (6% drop when replacing 20% of the entities per column up to 70% drop when replacing all of the entities).

Table 1
Overlap of entities per type in the WikiTables dataset.

<table>
<thead>
<tr>
<th>type</th>
<th>total</th>
<th>overlap</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>people.person</td>
<td>47852</td>
<td>29215</td>
<td>61.0</td>
</tr>
<tr>
<td>location.location</td>
<td>34073</td>
<td>21327</td>
<td>62.6</td>
</tr>
<tr>
<td>sports.pro_athlete</td>
<td>17588</td>
<td>10948</td>
<td>62.2</td>
</tr>
<tr>
<td>organization.organization</td>
<td>9904</td>
<td>7122</td>
<td>71.9</td>
</tr>
<tr>
<td>sports.sports_team</td>
<td>8207</td>
<td>6640</td>
<td>80.9</td>
</tr>
</tbody>
</table>

2. Related Work
With the growing popularity of LLMs, the concern over their vulnerability to adversarial attacks also increased.
We define a table as a tuple $(E, H)$, where $E = \{e_{1,1}, e_{2,1}, \ldots, e_{p,1}, \ldots, e_{m,1}\}$ is the set of table body entities for $n$ rows and $m$ columns. The table header $H = \{h_1, h_2, \ldots, h_m\}$ is the set of corresponding $m$ column header cells. We use $T_{i, j}$ to refer to the $i$-th row, e.g., $H = T_{i, 1}$ and $T_{i, j} = \{h_j, e_{1,j}, \ldots, e_{n,j}\}$ to refer to the $j$-th column of $T$.

### CTA Adversarial Attack

We define a table as a tuple $T = (E, H)$, where $E = \{e_{1,1}, e_{2,1}, \ldots, e_{p,1}, \ldots, e_{m,1}\}$ is the set of table body entities for $n$ rows and $m$ columns. The table header $H = \{h_1, h_2, \ldots, h_m\}$ is the set of corresponding $m$ column header cells. We use $T_{i, j}$ to refer to the $i$-th row, e.g., $H = T_{i, 1}$ and $T_{i, j} = \{h_j, e_{1,j}, \ldots, e_{n,j}\}$ to refer to the $j$-th column of $T$.

#### CTA Model

Let $\mathcal{T}$ be the input space of tables and let $f$ be the space of all possible column indices, i.e., $f \subseteq \mathbb{N}$. Let $\mathcal{C}$ be the output space, denoting the set of semantic types. A CTA model is a multilabel classification function $h : \mathcal{T} \times f \to P(\mathcal{C})$, i.e., given a table $T \in \mathcal{T}$ and a column index $j \in f$ the CTA task is to assign a subset of classes from the power set of $\mathcal{C}$ to the corresponding column $T_{1,j}$.

#### CTA Attack

Given classification model $h$, the goal of a CTA attack is to transform a (correctly classified) test input $(T, j) \in \mathcal{T} \times f$ into an (untargeted) adversarial sample $(T', j)$ such that $h(T, j) \cap h(T', j) = \emptyset$. In addition to fooling the classification model, the transformation from $T$ to $T'$ should also be imperceptible for a human observer. In the CTA setting we define the imperceptibility condition to be met if all entities in column $T'_{1,j}$ are of the same class as the unmodified column. Formally, $\forall e' \in T'_{1,j} \exists e \in T_{1,j} : c(e') = c(e)$, where $c \in \mathcal{C}$ represents the most specific class assigned to the column $T_{1,j}$.

### 3.1. Entity Swap Attack

In principle, a CTA attack can apply transformations to the full table $T$; however, most importantly, it should focus on $T_{1,j}$. Our attack, called entity-swap, follows a two-step approach inspired by adversarial attacks on LLMs \cite{Nature-Attack, ZhangAttacks}. First, it picks a set of key entities $\{e_i \in T_{1,j}\}$. The number of key entities can be controlled as a percentage of the entities in the original column. In a second step, every key entity $e_i$ is swapped with an adversarial entity $e'_i$ that most likely changes the predicted class from the ground truth. The proposed attack is a black-box attack, meaning we only have access to the predictions scores of the classifier.

### 3.2. Key Entities

Finding which are the key entities to swap can increase the rate of success of the attack. In the case of the CTA task, the most informative entities are those which, when replaced, the model will misclassify the column. To find those entities, we calculate an importance score for every entity in the attacked column.

The output from the classification model $h$ for a column $T_{1,j}$ is the logit vector $o_h(T_{1,j}) \in \mathbb{R}^k$, where $k$ is the number of ground-truth classes assigned to $(T, j)$. We calculate the importance score for entity $e_i \in T_{1,j}$ as the...
difference between the logit output of the model for the
ground truth classes when the entity is in the column,
denoted as \( o_h \), and the logit output of the model when
the entity is replaced with the [MASK] token, denoted as
\( o_{h'}/e_i \). Since, the CTA task is evaluated under the multi-
label setting, we always take the maximum importance
score for an entity.

\[
\text{score}(e_i) = \max(o_h - o_{h'}/e_i)
\]

Figure 2 shows an example of how the importance
score is calculated. We calculate \( o_h \) as the logit output
of the model without any perturbation, while \( o_{h'}/e_i \)
represents the logit output of the model when the entity
Rafael Nadal is masked. After calculating the importance
score for every entity in the column, we select the top
\( p \) percent of entities (\( p \in 20, 40, 60, 80, 100 \)) based on their
importance scores and substitute them with adversarial
entities. By sorting the entities according to their im-
portance scores, we ensure that the attack consistently
targets the key entities within the targeted column.

3.3. Adversarial Entities

After identifying the key entities, the next step involves
sampling adversarial entities for swapping. In order to
adhere to the perceptibility assumption, we constrain the
search space to include only entities belonging to the
same class as the attacked column. Subsequently, we use
a similarity-based strategy to sample adversarial entities.

Let \( e_i \in T_{[\ldots]} \) be the key entity from the attacked
column, and let \( c \in \mathbb{C} \) be the most specific class of this
column. We use an embedding model to generate a con-
textualized representation for both the original entity,
\( e_i \), and all entities of the same class \( A_c = \{e'_1, e'_2, \ldots, e'_k\} \),
such that \( c(e_i) = c(e'_1) \) where \( e'_i \in A_c \). Next, we calcu-
late the cosine similarity between the original entity and
each entity from the set \( A_c \). As an adversarial exam-
ple, we take the most dissimilar entity from the original
entity, such that \( e'_i = \argmax_{e'_j} \text{CosineSimilarity}(e_i, e'_j) \).
We then swap the original entity \( e_i \) with the adversarial
dentity \( e'_i \).

As we describe in the introduction, there is a substan-
tial overlap of entities between the train and test set.
Therefore, we propose two different sampling sets for
adversarial entities. The first, is the set of entities per-
class from the WikiTables test dataset [3]; we refer to
this set as \textit{test set}. The second set contains only novel
entities, i.e., entities that also appear in the training set,
are removed from the test set. We refer to this set as the
\textit{filtered set}.

\textbf{Metadata Attack} In addition to the proposed attack
method for column values, we also introduce an attack
specifically targeting column headers, considering that
they often indicate the class of a column. However, in this
case, we use an independent embedding model to identify
similar entities instead of swapping with column names
from the same class. For the generation of adversarial
samples in the column headers, we first generate embed-
dings for the original column names and then substitute
the column names with their synonyms. The library \texttt{TextAttack}
[14] was used to generate the embeddings, and
based on the embeddings to retrieve the synonyms for
the column names.

4. Evaluation

\textbf{Model} We evaluate the performance of the CTA attack
on the TURL model [3], which has been fine-tuned for
the CTA task and uses only entity mentions. We use
the WikiTables dataset for evaluation. We follow their
evaluation procedure and report the achieved F1 score,
precision, and recall.

To evaluate the influence of the proposed strategy for
sampling adversarial samples, we compare it to a random
sampling of adversarial entities. Similarly, to evaluate
the influence of the importance scores, we compare with
random sampling when choosing which entities to swap.

\subsection{Results}

Table 2 shows the results of the CTA attack when swapping
edentities by their importance scores and sampling
adversarial entities using the similarity-based strategy from the filtered set. We notice that as we increase the percentage of swapped entities, the performance of the model drops, even though the perturbed entities are of the same semantic type as the original entities. Another observation is that the drop in the F1 score is attributed to the sharp decline of the recall.

Effect of the importance score Figure 3 shows the benefit of using the importance scores. We notice that the drop in F1 score is around 3% higher when using the importance scores. This is consistent, regardless if we are substituting 20% or 80% of the entities, which suggests that the importance scores consistently identify entities that have a greater influence on the model’s performance.

Effect of the sampling strategy Figure 4 shows the difference in F1 score drop when sampling adversarial entities from the test set versus the filtered set. The original F1 score is represented by the red line. Additionally, here we illustrate the advantages of using the similarity-based strategy over a random-based sampling of adversarial examples. For both cases, when sampling adversarial entities from the test and filtered set, the similarity-based strategy for sampling induces sharper drop of the F1 score. This suggests that this approach is successful in selecting entities that are more likely to cause misclassifications or confusion for the classification model.

Effect of perturbing the table metadata To evaluate the relevance of the column header for the CTA task, we also propose an adversarial attack specific to the TURL model [3], which uses only the table metadata. Table 3 shows the effect of perturbing the table metadata. We observe similar results here, as we increase the percentage of perturbed column names, all the evaluation metrics decline. This indicates that the model’s reliance on specific column names, affects its ability to accurately classify and predict the correct class.

5. Conclusion

In this paper, we introduce the formalization of an adversarial attack targeting TaLMs. Additionally, we identify and highlight an issue concerning the evaluation of the CTA task. The evaluation showed that TaLMs are susceptible to adversarial attacks. Even subtle modifications to the entities, guided by similarity, can lead to significant changes in the model’s predictions and subsequently affect the F1 score. In future, we will extend our evaluation with more sophisticated attacks, targeting also other models used for table interpretation tasks.
References


