DISCOMAT: Distantly Supervised Composition Extraction from Tables in Materials Science Articles

Tanishq Gupta[1], **Mohd Zaki**[2], **Devanshi Khatsuriya**[3], **Kausik Hira**[4], **N. M. Anoop Krishnan**[4,2], **Mausam**[4,3]
[1]Department of Mathematics, [2]Department of Civil Engineering
[3]Department of Computer Science and Engineering, [4]Yardi School of Artificial Intelligence
Indian Institute of Technology Delhi
{tanishqg2406, mohdzaki1995, devanshikhatsuriya18, kausikhira}@gmail.com
{krishnan, mausam}@iitd.ac.in

Abstract

A crucial component in the curation of KB for a scientific domain (e.g., materials science, foods & nutrition, fuels) is information extraction from tables in the domain's published research articles. To facilitate research in this direction, we define a novel NLP task of extracting compositions of materials (e.g., glasses) from tables in material science papers. The task involves solving several challenges in concert, such as tables that mention compositions have highly varying structures; text in captions and full paper needs to be incorporated along with data in tables; and regular languages for numbers, chemical compounds and composition expressions must be integrated into the model.

We release a training dataset comprising 4,408 distantly supervised tables, along with 1,475 manually annotated dev and test tables. We also present DISCOMAT, a strong baseline that combines multiple graph neural networks with several task-specific regular expressions, features, and constraints. We show that DISCOMAT outperforms recent table processing architectures by significant margins. We release our code and data for further research on this challenging IE task from scientific tables.

1 Introduction

Advanced knowledge of a science or engineering domain is typically found in domain-specific research papers. Information extraction (IE) from scientific articles develops ML methods to automatically extract this knowledge for curating large-scale domain-specific KBs (e.g., (Ernst et al., 2015; Hope et al., 2021)). These KBs have a variety of uses: they lead to ease of information access by domain researchers (Tsatsaronis et al., 2015; Hamon et al., 2017), provide data for developing domain-specific ML models (Nadkarni et al., 2021), and potentially help in accelerating scientific discoveries (Jain et al., 2013; Venugopal et al., 2021).

Significant research exists on IE from *text* of research papers (see Nasar et al. (2018) for a survey), but less attention is given to IE (often, numeric) from *tables*. Tables may report the performance of algorithms on a dataset, quantitative results of clinical trials, or other important information. Of special interest to us are tables that mention the composition and properties of an entity. Such tables are ubiquitous in various fields such as food and nutrition (tables of food items with nutritional values, see Tables 1-4 in de Holanda Cavalcanti et al. (2021) and Table 2 in Stokvis et al. (2021)), fuels (constituents and calorific values, see Table 2 in Kar et al. (2022) and Beliavskii et al. (2022)), building construction (components and costs, see Table 4 in Aggarwal and Saha (2022)), materials (constituents and properties, see Table 1 and 2 in Kasimuthumaniyan et al. (2020) and Table 4 in Keshri et al. (2022)), medicine (compounds with weights in drugs, see Table 1 in Kalegari et al. (2014)), and more.

In materials science (MatSci) articles, the details on synthesis and characterization are reported in the text (Mysore et al., 2019), while material compositions are mostly reported in tables (Jensen et al., 2019b). A preliminary analysis of MatSci papers reveals that ∼85%[1] of material compositions and their associated properties (e.g., density, stiffness) are reported in tables and not text. Thus, IE from tables is essential for a comprehensive understanding of a given paper, and for increasing the coverage of resulting KBs. To this extent, we define a novel NLP task of extraction of materials (via IDs mentioned in the paper), constituents, and their relative percentages. For instance, Fig.1a should output four materials A1-A4, where ID A1 is associated with three constituents (MoO_3, Fe_2O_3, and P_2O_5) and their respective percentages, 5, 38, and 57. A model for this task necessitates solving several challenges, which are discussed in detail in Sec. 3. While many of these issues have been investigated

[1]estimated by randomly choosing 100 compositions from a MatSci database and checking where they are reported

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 13465–13483
July 9-14, 2023

separately, e.g., numerical IE (Madaan et al., 2016), unit extraction (Sarawagi and Chakrabarti, 2014), chemical compound identification (Weston et al., 2019), NLP for tables (Jensen et al., 2019b; Swain and Cole, 2016a), solving all these in concert creates a challenging testbed for the NLP community.

Here, we harvest a distantly supervised training dataset of 4,408 tables and 38,799 composition-constituent tuples by aligning a MatSci database with tables in papers. We also label 1,475 tables manually for dev and test sets. We build a baseline system DISCOMAT, which uses a pipeline of a domain-specific language model (Gupta et al., 2022), and two graph neural networks (GNNs), along with several hand-coded features and constraints. We evaluate our system on accuracy metrics for various subtasks, including material ID prediction, tuple-level predictions, and material-level complete predictions. We find that DISCOMAT's GNN architecture obtains a 7-15 points increase in accuracy numbers, compared to table processors (Herzig et al., 2020; Yin et al., 2020), which linearize the table for IE. Subsequent analysis reveals common sources of DISCOMAT errors, which will inform future research. We release all our data and code[2] for further research on this challenging task.

2 Related work

Recent works have developed neural models for various NLP tasks based on tabular data, *viz*, tabular natural language inference (Orihuela et al., 2021; Minhas et al., 2022), QA over one or a corpus of tables (Herzig et al., 2020; Yin et al., 2020; Arik and Pfister, 2021; Glass et al., 2021; Pan et al., 2021; Chemmengath et al., 2021), table orientation classification (Habibi et al., 2020; Nishida et al., 2017), and relation extraction from tables (Govindaraju et al., 2013; Macdonald and Barbosa, 2020). Several recent papers study QA models—they all linearize a table and pass it to a pre-trained language model. For example, TAPAS (Herzig et al., 2020) does this for Wikipedia tables to answer natural language questions by selecting table cells and aggregation operators. TABERT (Yin et al., 2020) and RCI (Glass et al., 2021) also use similar ideas alongside some architectural modifications to handle rows and columns better. TABBIE (Iida et al., 2021) consists of two transformers that encode rows and columns independently, whereas TAPEX uses encoder-decoder architecture using BART. TABBIE and TAPEX also introduce pre-training over tables to learn table representations better. Similar to our work, tables have also been modeled as graphs for sequential question answering over tables (Müller et al., 2019). However, all these works generally assume a fixed and known structure of tables with the same orientation, with the top row being the header row in all cases – an assumption violated in our setting.

Orientation and semantic structure classification: DeepTable (Habibi et al., 2020) is a permutation-invariant neural model, which classifies tables into three orientations, while TabNet (Nishida et al., 2017) uses RNNs and CNNs in a hybrid fashion to classify web tables into five different types of orientations. INFOTABS (Gupta et al., 2020) studies natural language inference on tabular data via linearization and language models, which has been extended to the multilingual setting (Minhas et al., 2022), and has been combined with knowledge graphs (Varun et al., 2022). Some earlier works also focused on annotating column types, entity ID cells, and pair of columns with binary relations, based on rule-based and other ML approaches, given a catalog (Limaye et al., 2010).

3 Challenges in composition extraction from tables

We analyze numerous composition tables in MatSci research papers (see Figures 1, 6 and 4 for examples), and find that the task has several facets, with many table styles for similar compositions. We now describe the key challenges involved in the task of composition extraction from tables.

- **Distractor rows and columns**: Additional information such as material properties, molar ratios, and std errors in the same table. E.g., in Figure 1a, the last three rows are distractor rows.
- **Orientation of tables**: Table shown in Figure 4a is a row oriented table—different compositions are written in different rows. The table in Figure 1a is a column-oriented table.
- **Different units**: Compositions can be in different units such as mol%, weight%, mol fraction, weight fraction. Some tables express composition in both molar and mass units.
- **Material IDs**: Authors refer to different materials in their publication by assigning them unique IDs. These material IDs may not be specified every time, (e.g., Fig. 1c).
- **Single-cell compositions (SCC)**: In Fig. 1a, all

[2] https://github.com/M3RG-IITD/DiSCoMaT

Sample Code		A1	A2	A3	A4
Batch Composition (mol %)	MoO_3	5	10	15	20
	Fe_2O_3	38	36	34	32
	P_2O_5	57	54	51	48
Molar ratio	Mo/P	0.04	0.09	0.15	0.21
	O/P	3.6	3.8	3.9	4.1
	Fe/P	0.67	0.67	0.67	0.67

Glass composition in the paper:
Caption: $20La_2S_3 - (80 - x)Ga_2S_3 - xCsCl$

Glass	CsCl (mol%)	n (1.5 μm)
GLSC10	10	2.253 ± 0.001
GLSC20	20	2.265 ± 0.001
GLSC30	30	2.322 ± 0.001
GLSC40	40	2.379 ± 0.001

Composition	$\log \sigma_{298}$ (S cm^{-1})
$(80GeS_2–20Ga_2S_3)_{90}–(LiI)_{10}$	–6.34 (5)
$(80GeS_2–20Ga_2S_3)_{90}–(NaCl)_{10}$	–6.92 (5)
$(80GeS_2–20Ga_2S_3)_{90}–(NaI)_{10}$	–7.24 (3)
$(72GeS_2–28Ga_2S_3)_{90}–(Li_2S)_{10}$	–5.74 (3)

Figure 1: Examples of composition tables (a) Multi-cell complete-info (Moguš-Milanković et al., 2003) (b) Multi-cell partial-info with caption on top (Marmolejo et al., 1999) (c) Single-cell (Brehault et al., 2014)

compositions are present in multiple table cells. Some authors report the entire composition in a single table cell, as shown in Fig. 1c.

• **Percentages exceeding 100:** Sum of coefficients may exceed 100, and re-normalization is needed. A common case is when a dopant is used; its amount is reported in excess.

• **Percentages as variables**: Contributions of constituents may be expressed using variables like x, y. In Fig. 6 (see App. A), x represents the mol% of ($GeBr_4$) and the 2nd row contains its value.

• **Partial-information tables**: It is also common to have percentages of only some constituents in the table; the remaining composition is to be inferred based on paper text or table caption, e.g., Figure 1b. Another example: if the paper is on silicate glasses, then SiO_2 is assumed.

• **Other corner cases**: There are several other corner cases like percentages missing from the table, compounds with variables (e.g., R_2O in the header; the value of R to be inferred from material ID), and highly unusual placement of information (some examples in appendix).

4 Problem formulation

Our goal is automated extraction of material compositions from tables. Formally, given a table T, its caption, and the complete text of publication in which T occurs, we aim to extract compositions expressed in T, in the form $\{(id, c_k^{id}, p_k^{id}, u_k^{id})\}_{k=1}^{K^{id}}$. Here, id represents the material ID, as used in the paper. Material IDs are defined by MatSci researchers to succinctly refer to that composition in text and other tables. c_k^{id} is a constituent element or compound present in the material, K^{id} is the total number of constituents in the material, $p_k^{id} > 0$ denotes the percentage contribution of c_k^{id} in its composition, and u_k^{id} is the unit of p_k^{id} (either mole% or weight%). For instance, the desired output tuples corresponding to ID A1 from Figure 1a are (A1, MoO_3, 5, mol%), (A1, Fe_2O_3, 38, mol%), (A1, P_2O_5, 57, mol%).

5 Dataset construction

We match a MatSci DB of materials and compositions with tables from published papers, to automatically provide distantly-supervised labels for extraction. We first use a commercial DB (NGF, 2019) of glass compositions with the respective references. Then, we extract all tables from the 2,536 references in the DB using text-mining API (els). We use a table parser (Jensen et al., 2019a) for raw XML tables and captions. This results in 5,883 tables of which 2,355 express compositions with 16,729 materials, and 58,481 (material ID, constituent, composition percentage, unit) tuples. We keep tables from 1,880 papers for training, and the rest are split into dev and test (see Table 4b).

The DB does not contain information about the location of a given composition in the paper – in text, images, graphs, or tables. If present in a table, it can appear in any column or row. Since we do not know the exact location of a composition, we use distantly supervised train set construction (Mintz et al., 2009). First, we simply match the chemical compounds and percentages (or equivalent fractions) mentioned in the DB with the text in a table from the associated paper. If all composition percentages are found in multiple cells of the table, it is marked as MCC-CI (multi-cell composition with complete information). However, due to several problems (see Appendix 3), it misses many composition tables. To increase the coverage, we additionally use a rule-based composition parser (described below), but restricted to only those compounds (CPD non-terminal in Figure 2) that appear in the DB for this paper.

Our distant supervision approach obtains table-level annotation (NC, SCC, MCC-PI, MCC-CI), where a table is labeled as non-composition, single/multi cell composition with partial/complete information. It also obtains annotation for each row or column into four labels: ID, composition,

constituent, and other. While training data is created using distant supervision, dev and test sets are hand annotated. We now explain the dataset construction process in further detail.

Rule-based composition parser: The parser helps find names of constituents from MCC tables, and also match full compositions mentioned in SCC tables. Recall that in SCC tables, the full *composition expression* is written in a single cell in the row/column corresponding to each Material ID. Such compositions are close to regular languages and can be parsed via regular expressions.

$CMP = PAT_1 \mid PAT_2 \mid PAT_3$
PAT_i = START CST_i (SEP CST_i)$^+$ END

CST_1 = NUM? W CPD
$CST_t = CST_1$ (SEP CST_1)*
CST_2 = (CST_t | OB CST_t CB) W NUM
CST_3 = NUM W (CST_t | OB CST_t CB)

Figure 2: Regexes in parser

Figure 2 shows the regular expression (simplified, for understandability) used by the parser. Here CMP denotes the matched composition, PATs are the three main patterns for it, CSTs are sub-patterns, CPD is a compound, NUM is a number, and OB and CB are, respectively, open and closed parentheses (or square brackets). W is zero or more whitespace characters, and SEP contains explicit separators like '-' or '+'. START and END are indicators to separate a regular expression from the rest of the text.

The first pattern parses simple number-compound expressions like $40Bi_2O_3$ * $60B_2O_3$. Here each of the two constituents will match with CST_1. The other two patterns handle *nested* compositions, where simple expressions are mixed in a given ratio. The main difference between the second and third patterns is in the placement of outer ratios – after or before the simple composition, respectively. Example match for PAT_2 is ($40Bi_2O_3$+$60B_2O_3$)30 - (AgI+AgCl)70, and for PAT_3 is $40Bi_2O_3$,$40B_2O_3$,20(AgI:2AgCl).

To materialize the rules of the rule-based composition parser, we pre-label compounds. For our dataset, we use a list-based extractor, though other chemical data extractors (Swain and Cole, 2016b) may also be used. After parsing, all coefficients are normalized so that they sum to hundred. For nested expressions, the outer ratio and the inner ones are normalized separately and then multiplied.

The compositions parsed by rule-based composition parser are then matched with entries in the DB. A successful matching leads to a high-quality annotation of composition expressions in these papers. If this matching happens: (i) in a single cell, the table is deemed as SCC, (ii) on caption/paper text that has an algebraic variable (or compound) found in the table, it is marked as MCC-PI (see Figure 1(b)). In case of no matching, the table is marked as NC. This automatic annotation is post-processed into row, column and edge labels.

One further challenge is that material IDs mentioned in papers are not provided in the DB. So, we manually annotate material IDs for all the identified composition tables in the training set. This leads to a train set of 11,207 materials with 38,799 tuples from 4,408 tables. Since the train set is distantly supervised and can be noisy, two authors (one of them is a MatSci expert) of this paper manually annotated the dev and test tables with row/column/edge labels, units, tuples, compositions, and table type, resulting in over 2,500 materials and over 9,500 tuples per set. We used Cohen's Kappa measure for identifying inter-annotator agreement, which was 86.76% for Glass ID, 98.47% for row and column labels, and 94.34% for table types. Conflicts were resolved through mutual discussions. Further statistics and the description of the developed in-house annotation tools used for manual annotations are discussed in A.2.

6 DISCOMAT architecture

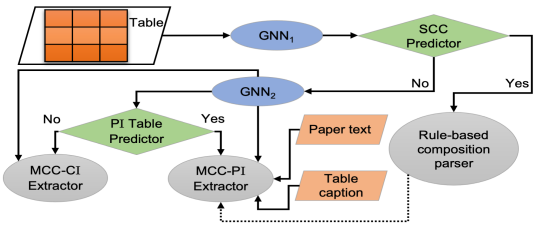

Figure 3: The design of DISCOMAT

Figure 3 illustrates the basic pipeline for extraction in DISCOMAT. We find that the simplest task is to identify whether the table T is an SCC table, owing to the distinctive presence of multiple numbers, and compounds in single cells. DISCOMAT first runs a GNN-based SCC predictor, which classifies T as an SCC table or not. For the SCC table, it uses the rule-based composition parser (described in Sec. 5). For the other category, DISCOMAT runs a second GNN (GNN_2), and labels rows and columns of T as compositions, material IDs, constituents, and others. If no constituents or composition predictions are found, then T is

deemed to be a non-composition (NC) table. Else, it is an MCC table, for which DISCOMAT predicts whether it has all information in T or some information is missing (partial-information predictor). If it is a complete information table, then GNN_2's predictions are post-processed into compositions. If not, the caption and text of the paper are also processed, along with GNN_2's predictions leading to final composition extraction. We note that our system ignores statistically infrequent corner cases, such as single-cell partial information tables – we discuss this further in our error analysis. We now describe each of these components, one by one.

6.1 GNN_1 and GNN_2 for table processing

At the core of DISCOMAT are two GNNs that learn representations for each cell, row, column and the whole table. Let table T has R rows, C columns, and text at $(i,j)^{th}$ cell be t_{ij}, where $1 \leq i \leq R$, and $1 \leq j \leq C$. We construct a directed graph $G_T = (V_T, E_T)$, where V_T has a node for each cell (i,j), one additional node for each row and column, denoted by $(i,0)$ and $(0,j)$, respectively, and one node for the whole table represented by $(0,0)$. There are bidirectional edges between two nodes of the same row or column. All cell nodes have directed edges to the table node and also their corresponding row and column nodes. The table, row, and column embeddings are randomly initialized with a common vector, which gets trained during learning. A node (i,j)'s embedding $\overrightarrow{x_{ij}}$ is initialized by running a language model LM over t_{ij}.

As constructed, G_T is permutation-invariant, i.e., if we permute rows or columns, we get the same graph and embeddings. However, initial rows/columns can be semantically different, since they often represent headings for the subsequent list. For instance, material IDs are generally mentioned in the first one or two rows/columns of the table. So, we additionally define *index embeddings* $\overrightarrow{p_i}$ to represent a row/column numbered i. We use the same index embeddings for rows and columns so that our model stays transpose-invariant. We also observe that while first few indices are different, the semantics is generally uniform for indices higher than 3. Accordingly, to allow DISCOMAT to handle large tables, we simply use $\overrightarrow{p_i} = \overrightarrow{p_3}\ \forall i > 3$. Finally, any manually-defined features added to each node are embedded as $\overrightarrow{f}$ and concatenated to the cell embeddings. Combining all ideas, a cell embedding is initialized as:

$$\overrightarrow{x_{ij}} = \overrightarrow{f_{ij}} \,||\, (LM_{CLS}(\langle CLS\ t_{ij}\ SEP\rangle) + \overrightarrow{p_i} + \overrightarrow{p_j})$$

Here, $1 \leq i \leq R, 1 \leq j \leq C$. $||$ is the concat operation and LM_{CLS} gives the contextual embedding of the CLS token after running a LM over the sentence inside $\langle\rangle$. Message passing is run on the graph G_T using a GNN, which computes a learned feature vector $\overrightarrow{h}$ for every node:

$$\{\overrightarrow{h_{ij}}\}_{i,j=(0,0)}^{(R,C)} = GNN\left(\{\overrightarrow{x_{ij}}\}_{i,j=(0,0)}^{(R,C)}\right).$$

6.2 SCC Predictor

In its pipeline, DISCOMAT first classifies whether T is an SCC table. For that, it runs a GNN (named GNN_1) on T with two manually defined features (see below). It then implements a Multi-layer Perceptron MLP_1 over the table-level feature vector $\overrightarrow{h_{00}}$ to make the prediction. Additionally, GNN_1 also feeds row and column vectors $\overrightarrow{h_{i0}}$ and $\overrightarrow{h_{0j}}$ through another MLP (MLP_2) to predict whether they contain material IDs or not. If T is predicted as an SCC table, then one with the highest MLP_2 probability is deemed as material ID row/column (provided probability > α, where α is a hyper-parameter tuned on dev set), and its contents are extracted as potential material IDs. If all row and column probabilities are less than α, then the table is predicted to not have Material IDs, as in Figure 1c.

For an SCC table, DISCOMAT must parse the full *composition expression* written in a single cell in the row/column corresponding to each Material ID, for which it makes use of the rule-based composition parser (as described in Section 5). The only difference is that at test time there is no DB available and hence extracted compositions cannot be matched with further. Consequently, DISCOMAT retains all extracted composition expresssions from the parser for further processing.

For units, DISCOMAT searches for common unit keywords such as mol, mass, weight, and their abbreviations like wt.%, and at.%. The search is done iteratively with increasing distance from the cell containing the composition. If not found in the table, then the caption is searched. If still not found, mole% is used as default.

Manual Features: GNN_1 uses two hand-coded features. The first feature is set to true if that cell contains a composition that matches our rule-based composition parser. Each value, true or false, is embedded as $\overrightarrow{o}$. The second feature named *max*

frequency feature adds the bias that material IDs are generally unique in a table. We compute q_i^r and q_j^c, which denote the maximum frequency of any non-empty string occurring in the cells of row i and column j, respectively. If these numbers are on the lower side, then that row/column has more unique strings, which should increase the probability that it contains material IDs. The computed q values are embedded in a vector as $\overrightarrow{q}$. The embedded feature $\overrightarrow{f_{ij}}$ for cell (i, j) is initialized as $\overrightarrow{o_{ij}} \mid\mid (\overrightarrow{q_{ij}^r} + \overrightarrow{q_{ij}^c})$.

6.3 MCC-CI and MCC-PI Extractors

If T is predicted to not be an SCC table, DISCOMAT runs it through another GNN (GNN_2). The graph structure is very similar to G_T from Section 6.1, but with two major changes. First, a new *caption node* is created with initial embedding as given by LM processing the caption text. Edges are added from the caption node to all row and column nodes. To propagate the information further to cells, edges are added from row/column nodes to corresponding cell nodes. The caption node especially helps in identifying non-composition (NC) tables. Second, the max frequency feature from Section 6.2 is also included in this GNN.

We use tables in Figure 4 as our running examples. While Figure 4a is a complete-information table, Figure 4b is not, and can only be understood in the context of its caption, which describes the composition as $[(Na_2O)_x(Rb_2O)_{1-x}]_y(B_2O_3)_{1-y}$. Here x and y are variables, which also need to be extracted and matched with the caption. DISCOMAT first decodes the row and column feature vectors $\overrightarrow{h_{i0}}$ and $\overrightarrow{h_{0j}}$, as computed by GNN_2, via an MLP_3 into four classes: composition, constituent, ID, and other (label IDs 1, 2, 3, 0, respectively). The figures illustrate this labelling for our running example. The cell at the intersection of composition row/column and constituent column/row represents the percentage contribution of that constituent in that composition.

Further, to associate the identified percentage contribution with the corresponding constituent (like P_2O_5 in Figure 4a) or variables x and y in Figure 4b), we perform classification at the edge level. For ease of exposition, we describe our method in this Section 6.3 for the setting that the table has been predicted by GNN_2 to have row-wise orientation, i.e., rows are compositions and columns are constituents. A transposed computation is done in the reverse case. Since the constituent/variable

	3	2	2	2	2	0
0	Sample	PbO	B_2O_3	P_2O_5	TeO_2	$\rho \pm 0.02$
0		(mol%)				[g·cm^{-3}]
1	LBPT0	50	10	40	0	4.97
1	LBPT1	45	9	36	10	5.14
1	LBPT2	40	8	32	20	5.16
1	LBPT3	35	7	28	30	5.19
1	LBPT4	30	6	24	40	5.31

	2	2	0	0	0	0
0	y	x	N_{Na}	T_g	$C_Q(BO_{3A})$	$\delta_{CS}(BO_{3A})$
0			(10^{27} m^{-3})	(K)	(MHz)	(ppm)
1	0.2	0	0	707.3	2.49	16.4
1	0.2	0.2	1.35	690.1	2.51	16.3
1	0.2	0.4	2.84	695.3	2.50	16.2
1	0.2	0.6	4.42	719.4	2.49	16.2
1	0.2	0.8	6.03	731.9	2.50	16.5

Figure 4: Multi-cell composition tables (a) Complete information (Koudelka et al., 2014) (b) Partial information (Epping et al., 2005)

will likely occur in the same column or row as the cell containing percentage contribution, our method computes an edge feature vector: for edge $e = (i, j) \rightarrow (i', j')$, s.t. $i = i' \vee j = j'$, the feature vector $\overrightarrow{h_e} = \overrightarrow{h_{ij}} \mid\mid \overrightarrow{h_{i'j'}}$. It then takes all such edges e from cell (i, j), if row i is labeled composition and column j is labeled constituent. Each edge e is classified through an MLP_4, and the edge with the maximum logit value is picked to identify the constituent/variable. This helps connect 36 to P_2O_5 and 0.8 to x in our running examples. GNN_2 also helps in predicting NC tables. In case none of the rows/columns are predicted as 1 or 2, then the table is deemed as NC and discarded.

Partial information table predictor: Next, DISCOMAT distinguishes between complete-information (CI) and partial-information (PI) MCC tables. It uses a logistic regression model with custom input features for this prediction task. Let P and Q be the sets of all row indices with label 1 (composition) and column indices with label 2 (constituent), respectively. Also, assume n_{ij} is the number present in table cell (i, j) or 0 if no number is present. To create the features, we first extract all the constituents (compounds) and variables predicted by MLP_4. We now construct five table-level features (F1-F5). F1 and F2 count the number of unique variables and chemical compounds extracted by MLP_4. The intuition is that if F1 is high, then it is more likely an MCC-PI, and vice-versa if F2 is high. F3 computes the number of rows and columns labeled as 2 (constituent) by MLP_3. The more the value of F3, the more likely it is that the table is MCC-CI. Features F4 (and F5) compute the maximum (average) of the sum of all extracted compositions. The intuition of F4 and F5 is that the

higher these feature values, the higher the chance of the table being an MCC-CI. Formally,

$$\text{F4} = (\max_{i \in P} \sum_{j \in Q} n_{ij}) \qquad \text{F5} = (\frac{1}{|P|} \sum_{i \in P} \sum_{j \in Q} n_{ij}).$$

MCC table extractor: For MCC-CI, MLP_3 and MLP_4 outputs are post-processed, and units are added (similar to SCC tables), to construct final extracted tuples. For MCC-PI, on the other hand, information in the text needs to be combined with the MLP outputs for final extraction. The first step here is to search for the composition expression, which may be present in the table caption, table footer, and if not there, somewhere in the rest of the research paper. Here, DISCOMAT resorts to using our rule-based composition parser from Figure 2, but with one key difference. Now, the composition may contain variables (x, y) and even mathematical expressions like $100 - x$. So the regular grammar is enhanced to replace the non-terminal NUM with a non-terminal EXPR, which represents, numbers, variables, and simple mathematical expressions over them. An added constraint is that if there are variables in set Q, then those variables must be present in the matched composition expression. DISCOMAT completes the composition by substituting the variable values from every composition row into the matched composition. There may be other types of MCC-PI tables where only compounds are identified in tables, such as Figure 1b. For these, DISCOMAT first computes the constituent contributions in terms of variables from the composition expression, and then equates it with the numbers present in rows/columns labeled 1 (composition). In our example, DISCOMAT matches x with the numbers 10, 20, 30, and 40, and the rest of the composition is extracted by processing the composition expression in the caption with these values of x. Units and material IDs are added to the tuples, similar to other tables.

6.4 Constraint-aware loss functions

DISCOMAT needs to train the two GNNs and the PI table predictor. Our data construction provides gold labels for each prediction task (discussed in the next section), so we train them component-wise. The PI table predictor is trained on standard logistic regression loss. GNN_1 is trained on a weighted sum of binary cross entropy loss for SCC table classification and row/column classification for material IDs – weight is a hyper-parameter. Similarly, the GNN_2 loss function consists of the sum of row/column cross-entropy and edge binary cross-entropy losses.

GNN_2 has a more complex prediction problem since it has to perform four-way labeling for each row and column. In initial experiments, we find that the model sometimes makes structural errors like labeling one row as a constituent and another row as a composition in the same table – highly unlikely as per the semantics of composition tables. To encourage GNN_2 to make structurally consistent predictions, we express a set of constraints on the complete labelings, as follows. (1) A row and a column cannot both have compositions or constituents. (2) Composition and material ID must be orthogonally predicted (i.e, if a row has a composition then ID must be predicted in some column, and vice versa). (3) Constituents and material IDs must never be orthogonally predicted (if rows have constituents then another row must have the ID). And, (4) material ID must occur at most once for the entire table. As an example, constraint (1) can be expressed as a hard constraint as:

$$r_i = l \Rightarrow c_j \neq l \quad \forall i \in \{1, \ldots, R\}, j \in \{1, \ldots, C\}, l \in \{1, 2\}.$$

Here, r_i and c_j are predicted labels of row i and column j. We wish to impose these structural constraints at training time so that the model is trained to honor them. We follow prior work by Nandwani et al. (Nandwani et al., 2019), to first convert these hard constraints into a probabilistic statement. For example, constraint (1) gets expressed as:

$$P(r_i = l; \theta) + P(c_j = l; \theta) - 1 \leq 0$$
$$\forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\},\ l \in \{1, 2\}.$$

θ represents GNN_2's parameters. Following the same work, each such constraint gets converted to an auxiliary penalty term, which gets added to the loss function for constraint-aware training. The first constraint gets converted to: $\lambda \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{l=1}^{2} max(0, P(r_i = l; \theta) + P(c_j = l; \theta) - 1)$.
This and similar auxiliary losses for other constraints (App. A.1) get added to the GNN_2's loss function for better training. λ is a hyper-parameter. We also use constraint (4) for GNN_1 training.

7 Experiments

Baseline models: We implement DISCOMAT with LM as MATSCIBERT (Gupta et al., 2022), and the GNNs as Graph Attention Networks

(Veličković et al., 2018). We compare DISCOMAT with six non-GNN baseline models. Our first baseline is TAPAS (Herzig et al., 2020), a state-of-the-art table QA system, which flattens the table, adds row and column index embeddings, and passes as input to a language model. To use TaPas for our task, we use table caption as a proxy for the input question. All the model parameters in this setting are initialized randomly. Next, we use TABERT (Yin et al., 2020), which is a pretrained LM that jointly learns representations for natural (NL) sentences and tables by using pretraining objectives of masked column prediction (MCP) and cell value recovery (CVR). It finds table cell embeddings by passing row linearizations concatenated with the NL sentence into a language model and then applying vertical attention across columns for information propagation. Finally, we use TABBIE, which is pretrained by corrupt cell detection and learns exclusively from tabular data without any associated text, unlike the previous baselines. Additionally, we replace the LM of all models with MATSCIBERT to provide domain-specific embeddings to obtain the respective ADAPTED versions. We also implement a simple rule-based baseline for MCC-CI and NC tables. The baseline identifies constituent names using regex matching and a pre-defined list of compounds, extracts numbers from cells and finds the units using simple heuristics to generate the required tuples. Further details on baselines is provided in App. A.3.

Evaluation metrics: We compute several metrics in our evaluation. (1) *Table-type (TT) prediction accuracy* – it computes table-level accuracy on the 4-way table classification as NC, SCC, MCC-CI and MCC-PI. (2) *ID F_1 score* computes F_1 score for Material ID extraction. (3) *Tuple-level (TL) F_1 score* evaluates performance on the extraction of composition tuples. A gold is considered matching with a predicted 4-tuple if *all* arguments match exactly. (4) *Material-level (MatL) F_1 score* is the strongest metric. It evaluates whether all predicted information related to a material (including its ID, all constituents and their percentages) match exactly with the gold. Finally, (5) *constraint violations (CV)* counts the number of violations of hard constraints in the prediction. We consider all four types of constraints, as discussed in Section 6.4. Implementation details are mentioned in App. A.4.

7.1 Results

How does table linearization compare with a graph-based model for our task? To answer this question, we compare DISCOMAT with four models that use linearization: TAPAS, TABERT, and their adapted versions. TAPAS and TABERT do table level and row level linearizations respectively. Since the baselines do not have the benefit of regular expressions, features, and constraints, we implement a version of our model without these, which we call V-DISCOMAT. We do this comparison, trained and tested only on the subset of MCC-CI and NC tables since other table types require regular expressions for processing. As shown in Table 1 V-DISCOMAT obtain 6-7 pt higher F_1 on TL and MatL scores. Moreover, compared to the RULE BASED SYSTEM, DISCOMAT obtains upto 17 points improvement in the MatL F1 score. This experiment suggests that a graph-based extractor is a better fit for our problem – this led to us choosing a GNN-based approach for DISCOMAT.

How does DISCOMAT *perform on the complete task?* Table 2, reports DISCOMAT performance on the full test set with all table types. Its ID and tuple F_1-scores are 82 and 70, respectively. Since these errors get multiplied, unsurprisingly, its material-level F_1-score is lower (63.5). Table 3 reports DISCOMAT performance for different table types. In this experiment, we assume that the table type is already known and run only the relevant part of DISCOMAT for extraction. We find that MCC-PI is the hardest table type since it requires combining information from text and tables for accurate extraction. A larger standard deviation in ID F_1 for MCC-PI is attributed to the fact that material IDs occur relatively rarely for this table type – the test set for MCC-PI consists of merely 20 material ID rows and columns.

What is the incremental contribution of task-specific features and constraints? Table 2 also presents the ablation experiments. DISCOMAT scores much higher than V-DISCOMAT, which does not have these features and constraints. We also perform additional ablations removing one component at a time. Unsurprisingly constrained training helps with reducing constraint violations. Both constraints and features help with ID prediction, due to constraints (2), (3), (4) and max frequency feature. Removal of caption nodes significantly hurts performance on MCC-PI tables, as these tables require combining caption with table

Model	ID F_1	TL F_1	MatL F_1	CV
TAPAS	80.37 (± 4.78)	71.23 (± 0.77)	49.88 (± 0.10)	543.67
TAPAS-ADAPTED	**89.65** (± 0.46)	70.91 (± 3.79)	57.88 (± 2.73)	490.33
TABERT	79.61 (± 8.25)	58.20 (± 1.79)	47.05 (± 1.50)	1729.67
TABERT-ADAPTED	85.07 (± 6.28)	59.31 (± 0.67)	50.10 (± 2.86)	1195.67
TABBIE	80.99 (± 2.41)	50.90 (± 3.34)	47.03 (± 2.14)	**388.00**
TABBIE-ADAPTED	80.18 (± 5.38)	53.20 (± 5.57)	48.89 (± 2.73)	728.67
RULE BASED SYSTEM	72.64	54.44	47.38	0
V-DISCOMAT	77.38 (± 12.21)	**76.52** (± 2.37)	**64.71** (± 3.45)	626.33

Table 1: Performance of V-DISCOMAT vs baseline models on the subset of data containing only MCC-CI and NC table types.

Model	TT Acc.	ID F_1	TL F_1	MatL F_1	CV
DISCOMAT	88.35 (± 1.20)	84.57 (± 2.16)	**70.04** (± 0.69)	**63.53** (± 1.45)	75.22
DISCOMAT w/o features	**88.84** (± 1.00)	84.15 (± 1.61)	68.31 (± 1.45)	62.47 (± 1.98)	83.11
DISCOMAT w/o constraints	88.47 (± 0.31)	84.07 (± 0.83)	69.68 (± 1.21)	61.44 (± 1.00)	434.44
DISCOMAT w/o captions	87.35 (± 0.71)	**84.76** (± 0.68)	66.82 (± 1.90)	62.68 (± 3.33)	**17.89**
V-DISCOMAT	88.59 (± 0.33)	76.61 (± 6.16)	66.15 (± 2.00)	59.52 (± 3.33)	380.11

Table 2: Contribution of task-specific features and constraints in DISCOMAT on the complete dataset.

Table Type	ID F_1	TL F_1	MatL F_1
SCC	88.81 (± 1.54)	79.89 (± 0.18)	78.21 (± 0.14)
MCC-CI	93.91 (± 1.46)	77.62 (± 1.07)	65.41 (± 4.35)
MCC-PI	70.67 (± 11.58)	50.60 (± 2.59)	51.66 (± 2.21)

Table 3: DISCOMAT performance on the table-types.

cells. Although the ablation study done by removing features, constraints, and captions individually does not show much of a difference on the tuple-level and material-level scores, we observe that on removing all the three factors, the performance of V-DISCOMAT drops significantly. Therefore, we can conclude that even though each component is improving the performance of DISCOMAT marginally, collectively, they help us to achieve significant gains.

What are the typical errors in DISCOMAT*?* The confusion matrix in Figure 5 suggests that most table-type errors are between MCC-PI and NC tables. This could be attributed to the following reasons. (i) DISCOMAT has difficulty identifying rare compounds like Yb_2O_3, $ErS_{3/2}$, Co_3O_4 found in MCC-PI—these aren't present frequently in the training set. (ii) MCC-PI tables specify dopant percentages found in small quantities. (iii) Completion of composition in MCC-PI tables may require other tables from the same paper. (iv) Finally, MCC-PI composition may contain additional information such as properties that may bias the model to classify it as NC. Some corner cases are given in App. A.6.

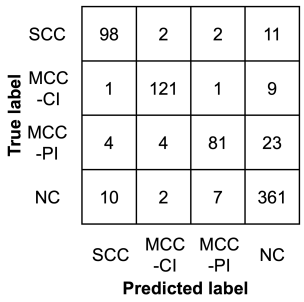

Figure 5: Confusion matrix for all table types

8 Conclusions

We define the novel and challenging task of extracting material compositions from tables in scientific papers. This task has importance beyond material science, since many other scientific disciplines use tables to express compositions in their domains. We harvest a dataset using distant supervision, combining information from a MatSci DB with tables in respective papers. We present a strong baseline system DISCOMAT, for this task. It encodes tables as graphs and trains GNNs for table-type classification. Further, to handle incomplete information in PI tables, it includes the text associated with the tables from respective papers. To handle domain-specific regular languages, a rule-based composition parser helps the model by extracting chemical compounds, numbers, units, and composition expressions. We find that our DISCOMAT baseline outperforms other architectures that linearize the tables by huge margins. In the future, our work can be extended to extract material properties that are also often found in tables. The code and data are made available in the GitHub repository of this work.

Acknowledgements

N. M. Anoop Krishnan acknowledges the funding support received from SERB (ECR/2018/002228), DST (DST/INSPIRE/04/2016/002774), BRNS YSRA (53/20/01/2021-BRNS), ISRO RESPOND as part of the STC at IIT Delhi. Mohd Zaki acknowledges the funding received from the PMRF award by Government of India. Mausam acknowledges grants by Google, IBM, Verisk, and a Jai Gupta chair fellowship. He also acknowledges travel support from Google and Yardi School of AI travel grants. The authors thank the High Performance Computing (HPC) facility at IIT Delhi for computational and storage resources.

Limitations and outlook

DISCOMAT is a pipelined solution trained component-wise. This raises a research question: can we train one end-to-end trained ML model that not only analyzes a wide variety of table structures but also combines the understanding of regular expressions, extraction of chemical compounds and scientific units, textual understanding and some mathematical processing? This defines a challenging ML research question and one that can have a direct impact on the scientific MatSci community. Indeed, automating parts of scientific discovery through such NLP-based approaches has the potential for biases and errors. Note that wrong and biased results can lead to erroneous information about materials. To a great extent, this issue is addressed as we rely only on published literature. The issue could be further addressed by considering larger datasets covering a wider range of materials.

References

Elsevier Developer Portal.

Yati Aggarwal and Sandip Kumar Saha. 2022. Component repair cost functions in indian context for seismic loss estimation of reinforced concrete buildings. *Structures*.

Sercan Ö. Arik and Tomas Pfister. 2021. TabNet: Attentive interpretable tabular learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):6679–6687.

Sergei V. Beliavskii, N B Anikin, Sirajo Alhassan, S. Kudeev, and V. O. Nesterov. 2022. Effect of fuel nuclide composition on the fuel lifetime of the ritm-200 reactor unit. *Annals of Nuclear Energy*.

Antoine Brehault, Solenn Cozic, Rémi Boidin, Laurent Calvez, Eugène Bychkov, Pascal Masselin, Xianghua Zhang, and David Le Coq. 2014. Influence of nax (x= i or cl) additions on ges2–ga2s3 based glasses. *Journal of Solid State Chemistry*, 220:238–244.

Saneem A. Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Jaydeep Sen, Mustafa Canim, Soumen Chakrabarti, Alfio Gliozzo, and Karthik Sankaranarayanan. 2021. Topic transferable table question answering. 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*, pages 4159–4172.

Natália Sufiatti de Holanda Cavalcanti, Tatiana Colombo Pimentel, Marciane Magnani, Maria Teresa Bertoldo Pacheco, Susana Paula Almeida Alves, Rui José Branquinho Bessa, Amanda Marília da Silva Sant'Ana, and Rita de Cássia Ramos do Egypto Queiroga. 2021. Donkey milk and fermented donkey milk: are there differences in the nutritional value and physicochemical characteristics? *Lwt - Food Science and Technology*, 144:111239.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

J.R. Duclère, A.A. Lipovskii, A.P. Mirgorodsky, Ph. Thomas, D.K. Tagantsev, and V.V. Zhurikhina. 2009. Kerr studies of several tellurite glasses. *Journal of Non-Crystalline Solids*, 355(43):2195–2198.

Epam. Epam/sciglass: The database contains a vast set of data on the properties of glass materials.

Jan Dirk Epping, Hellmut Eckert, Árpád W Imre, and Helmut Mehrer. 2005. Structural manifestations of the mixed-alkali effect: Nmr studies of sodium rubidium borate glasses. *Journal of non-crystalline solids*, 351(43-45):3521–3529.

Patrick Ernst, Amy Siu, and Gerhard Weikum. 2015. Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences. *BMC Bioinform.*, 16:157:1–157:13.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Michael R. Glass, Mustafa Canim, Alfio Gliozzo, Saneem A. Chemmengath, Vishwajeet Kumar, Rishav Chakravarti, Avi Sil, Feifei Pan, Samarth Bharadwaj, and Nicolas Rodolfo Fauceglia. 2021. Capturing row and column semantics in transformer based question answering over tables. In *NAACL-HLT*, pages 1212–1224. Association for Computational Linguistics.

Vidhya Govindaraju, Ce Zhang, and Christopher Ré. 2013. Understanding tables in context using standard nlp toolkits. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 658–664.

Tanishq Gupta, Mohd Zaki, N. M. Anoop Krishnan, and Mausam. 2022. MatSciBERT: A materials domain language model for text mining and information extraction. *npj Computational Materials*, 8(1):102.

Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: inference on tables as semi-structured data. In *ACL*, pages 2309–2324. Association for Computational Linguistics.

Maryam Habibi, Johannes Starlinger, and Ulf Leser. 2020. Deeptable: a permutation invariant neural network for table orientation classification. *Data Mining and Knowledge Discovery*, 34(6):1963–1983.

Thierry Hamon, Natalia Grabar, and Fleur Mougin. 2017. Querying biomedical linked data with natural language questions. *Semantic Web*, 8(4):581–599.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Tom Hope, Aida Amini, David Wadden, Madeleine van Zuylen, Sravanthi Parasa, Eric Horvitz, Daniel S. Weld, Roy Schwartz, and Hannaneh Hajishirzi. 2021. Extracting a knowledge base of mechanisms from COVID-19 papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4489–4503. Association for Computational Linguistics.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. TABBIE: pretrained representations of tabular data. In *NAACL-HLT*, pages 3446–3456. Association for Computational Linguistics.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR.

Anubhav Jain, Shyue Ping Ong, Geoffroy Hautier, Wei Chen, William Davidson Richards, Stephen Dacek, Shreyas Cholia, Dan Gunter, David Skinner, Gerbrand Ceder, et al. 2013. Commentary: The materials project: A materials genome approach to accelerating materials innovation. *APL materials*, 1(1):011002.

Zach Jensen, Edward Kim, Soonhyoung Kwon, Terry Z. H. Gani, Yuriy Roman-Leshkov, Manuel Moliner, Avelino Corma, and Elsa Olivetti. 2019a. A machine learning approach to zeolite synthesis enabled by automatic literature data extraction. *ACS Central Science*, 5(5):892–899.

Zach Jensen, Edward Kim, Soonhyoung Kwon, Terry ZH Gani, Yuriy Roman-Leshkov, Manuel Moliner, Avelino Corma, and Elsa Olivetti. 2019b. A machine learning approach to zeolite synthesis enabled by automatic literature data extraction. *ACS central science*, 5(5):892–899.

Milena Kalegari, Murilo Luiz Cerutti, Sérgio José Macedo-Júnior, Franciane Bobinski, Marilis Dallarmi Miguel, Véronique Eparvier, Adair Roberto Soares Santos, Didier Stien, and Obdulio Gomes Miguel. 2014. Chemical composition and antinociceptive effect of aqueous extract from rourea induta planch. leaves in acute and chronic pain models. *Journal of Ethnopharmacology*, 153(3):801–809.

Tanmay Kar, Toluwalase Fosudo, Anthony J. Marchese, Bret C. Windom, and Daniel Olsen. 2022. Effect of fuel composition and egr on spark-ignited engine combustion with lpg fueling: Experimental and numerical investigation. *Fuel*.

S Kasimuthumaniyan, Allu Amarnath Reddy, NM Anoop Krishnan, and Nitya Nand Gosvami. 2020. Understanding the role of post-indentation recovery on the hardness of glasses: Case of silica, borate, and borosilicate glasses. *Journal of Non-Crystalline Solids*, 534:119955.

Nirmal Kaur, Atul Khanna, Marina Gónzález-Barriuso, Fernando González, and Banghao Chen. 2015. Effects of al3+, w6+, nb5+ and pb2+ on the structure and properties of borotellurite glasses. *Journal of Non-Crystalline Solids*, 429:153–163.

Shweta R Keshri, Indrajeet Mandal, Sudheer Ganisetti, S Kasimuthumaniyan, Rajesh Kumar, Anuraag Gaddam, Ankita Shelke, Thalasseril G Ajithkumar, Nitya Nand Gosvami, NM Anoop Krishnan, et al. 2022. Elucidating the influence of structure and ag+-na+ ion-exchange on crack-resistance and ionic conductivity of na3al1. 8si1. 65p1. 8o12 glass electrolyte. *Acta Materialia*, 227:117745.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Ladislav Koudelka, Ivana Rösslerová, Zdeněk Černošek, Petr Mošner, Lionel Montagne, and Bertrand Revel. 2014. The structural role of tellurium dioxide in lead borophosphate glasses. *Journal of non-crystalline solids*, 401:124–128.

Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*, 3(1):1338–1347.

Erin Macdonald and Denilson Barbosa. 2020. Neural relation extraction on wikipedia tables for augmenting knowledge graphs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2133–2136, New York, NY, USA. Association for Computing Machinery.

Aman Madaan, Ashish R. Mittal, Mausam, Ganesh Ramakrishnan, and Sunita Sarawagi. 2016. Numerical relation extraction with minimal supervision. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2764–2771. AAAI Press.

EM Marmolejo, E Granado, OL Alves, CL Cesar, and LC Barbosa. 1999. Spectroscopy and thermal properties of ga2s3 based glasses. *Journal of non-crystalline solids*, 247(1-3):189–195.

David A McKeown, Wing K Kot, and Ian L Pegg. 2003. X-ray absorption studies of the local strontium environments in borosilicate waste glasses. *Journal of Non-Crystalline Solids*, 317(3):290–300.

Bhavnick Minhas, Anant Shankhdhar, Vivek Gupta, Divyanshu Aggrawal, and Shuo Zhang. 2022. XInfoTabS: evaluating multilingual tabular natural language inference.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

A Moguš-Milanković, A Šantić, A Gajović, and DE Day. 2003. Spectroscopic investigation of moo3–fe2o3–p2o5 and sro–fe2o3–p2o5 glasses. part i. *Journal of non-crystalline solids*, 325(1-3):76–84.

Thomas Müller, Francesco Piccinno, Peter Shaw, Massimo Nicosia, and Yasemin Altun. 2019. Answering conversational questions on structured data without logical forms. In *EMNLP/IJCNLP (1)*, pages 5901–5909. Association for Computational Linguistics.

T. Murata, M. Sato, H. Yoshida, and K. Morinaga. 2005. Compositional dependence of ultraviolet fluorescence intensity of ce3+ in silicate, borate, and phosphate glasses. *Journal of Non-Crystalline Solids*, 351(4):312–316.

Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop*, pages 56–64, Florence, Italy. Association for Computational Linguistics.

Rahul Nadkarni, David Wadden, Iz Beltagy, Noah A. Smith, Hannaneh Hajishirzi, and Tom Hope. 2021. Scientific language models for biomedical knowledge base completion: An empirical study. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. 2019. A primal dual formulation for deep learning with constraints. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zara Nasar, Syed Waqar Jaffry, and Muhammad Kamran Malik. 2018. Information extraction from scientific articles: a survey. *Scientometrics*, 117(3):1931–1990.

Japan NGF. 2019. International glass database system.

Kyosuke Nishida, Kugatsu Sadamitsu, Ryuichiro Higashinaka, and Yoshihiro Matsuo. 2017. Understanding the semantic structures of tables with a hybrid deep neural network architecture. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Refka Oueslati Omrani, Saida Krimi, Jean Jacques Videau, Ismail Khattech, Abdelaziz El Jazouli, and Mohamed Jemal. 2014. Structural investigations and calorimetric dissolution of manganese phosphate glasses. *Journal of Non-Crystalline Solids*, 389:66–71.

Mario Alberto Ramirez Orihuela, Alex Bogatu, Norman Paton, and André Freitas. 2021. Natural language inference over tables: Enabling explainable data exploration on data lakes. In *Eighteenth Extended Semantic Web Conference - Research Track*.

Feifei Pan, Mustafa Canim, Michael Glass, Alfio Gliozzo, and Peter Fox. 2021. CLTR: An end-to-end, transformer-based system for cell-level table retrieval and table question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 202–209, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Sunita Sarawagi and Soumen Chakrabarti. 2014. Open-domain quantity queries on web tables: annotation, response, and consensus models. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 711–720. ACM.

Yong Beom Shin, Chang Kuk Yang, and Jong Heo. 2002. Optimization of dy3+-doped ge–ga–as–s–csbr glass composition and its 1.31 μm emission properties. *Journal of Non-Crystalline Solids*, 298(2):153–159.

Leslie N. Smith. 2017. Cyclical learning rates for training neural networks. In *WACV*, pages 464–472. IEEE Computer Society.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958.

L. Stokvis, Marinus van Krimpen, René P. Kwakkel, and Paul Bikker. 2021. Evaluation of the nutritional value of seaweed products for broiler chickens' nutrition. *Animal Feed Science and Technology*, 280:115061.

Matthew C Swain and Jacqueline M Cole. 2016a. Chemdataextractor: a toolkit for automated extraction of chemical information from the scientific literature. *Journal of chemical information and modeling*, 56(10):1894–1904.

Matthew C. Swain and Jacqueline M. Cole. 2016b. Chemdataextractor: A toolkit for automated extraction of chemical information from the scientific literature. *Journal of Chemical Information and Modeling*, 56(10):1894–1904.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel-Cyrille Ngonga Ngomo, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC Bioinform.*, 16:138:1–138:28.

Osamu Uemura, Takeshi Usuki, Masanori Inoue, Keigo Abe, Yasuo Kameda, and Masaki Sakurai. 2001. Local atomic order of ge–se–br glasses. *Journal of non-crystalline solids*, 293:792–798.

Yerram Varun, Aayush Sharma, and Vivek Gupta. 2022. Trans-KBLSTM: an external knowledge enhanced transformer BiLSTM model for tabular reasoning.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*.

Vineeth Venugopal, Suresh Bishnoi, Sourabh Singh, Mohd Zaki, Hargun Singh Grover, Mathieu Bauchy, Manish Agarwal, and NM Anoop Krishnan. 2021. Artificial intelligence and machine learning in glass science and technology: 21 challenges for the 21st century. *International journal of applied glass science*, 12(3):277–292.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

Leigh Weston, Vahe Tshitoyan, John Dagdelen, Olga Kononova, Amalie Trewartha, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. 2019. Named entity recognition and normalization applied to large-scale information extraction from the materials science literature. *Journal of chemical information and modeling*, 59(9):3692–3702.

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. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

A Appendix

	$(GeSe_2)_{1-x}(GeBr_4)_x$			$(GeSe_2)_{1-x}Br_x$	
x	0.079	0.167	0.265	0.250	0.429
p_{Se}(%)	94.1	86.0	76.5	84.1	78.0
	(92.1)	(83.3)	(73.5)	(91.7)	(81.2)
p_{Br}(%)	5.8	14.0	23.5	15.9	22.0
	(7.9)	(16.7)	(26.5)	(8.3)	(18.8)

Figure 6: Percentages as variables (Uemura et al., 2001)

A.1 Constraint-aware training

As discussed in Section 6.4, to encourage GNN_2 to make structurally consistent predictions, we express a set of constraints on the complete labeling as follows. (1) A row and a column cannot both have compositions or constituents. (2) Composition and material ID must be orthogonally predicted (i.e., if a row has a composition, then the ID must be predicted in some column, and vice versa). (3) Constituents and material IDs must never be orthogonally predicted (that is, if rows have constituents, then another row in the table must have the ID). And, (4) material ID must occur at most once for the entire table. Let r_i and c_j be the predicted labels of row i and column j. Further, let θ represent GNN_2's parameters.

Constraint (1) is expressed as a hard constraint by:

$$r_i = l \Rightarrow c_j \neq l$$
$$\forall i \in \{1, \ldots, R\}, j \in \{1, \ldots, C\},\ l \in \{1, 2\}.$$

The equivalent probabilistic statement is:

$$P(r_i = l; \theta) + P(c_j = l; \theta) - 1 \leq 0$$
$$\forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\},\ l \in \{1, 2\}.$$

Constraint (2) can be written in the form of hard constraints as:

$$r_{i_1} = 1 \Rightarrow r_{i_2} \neq 3\ \ \forall i_1, i_2 \in \{1, \ldots, R\}, i_1 \neq i_2.$$

$$c_{j_1} = 1 \Rightarrow c_{j_2} \neq 3\ \forall j_1, j_2 \in \{1, \ldots, C\}, j_1 \neq j_2.$$

Equivalent probabilistic statements are:

$$P(r_{i_1} = 1; \theta) + P(r_{i_2} = 3; \theta) - 1 \leq 0$$
$$\forall i_1,\ i_2 \in \{1, \ldots, R\},\ i_1 \neq i_2.$$

$$P(c_{j_1} = 1; \theta) + P(c_{j_2} = 3; \theta) - 1 \leq 0$$
$$\forall j_1,\ j_2 \in \{1, \ldots, C\},\ j_1 \neq j_2.$$

We write constraint (3) in a hard constraint form as:

$$r_i = l \Rightarrow c_j \neq 5 - l$$
$$\forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\},\ l \in \{2, 3\}.$$

The equivalent probabilistic statement is:

$$P(r_i = l; \theta) + P(c_j = 5 - l; \theta) - 1 \leq 0$$
$$\forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\},\ l \in \{2, 3\}.$$

Finally, hard versions of constraint (4) can be stated as:

$$r_{i_1} = 3 \Rightarrow r_{i_2} \neq 3 \quad 1 \leq i_1 < i_2 \leq R.$$

$$c_{j_1} = 3 \Rightarrow c_{j_2} \neq 3 \quad 1 \leq j_1 < j_2 \leq C.$$

$$r_i = 3 \Rightarrow c_j \neq 3 \quad \forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\}.$$

Equivalent probabilistic statements are:

$$P(r_{i_1} = 3; \theta) + P(r_{i_2} = 3; \theta) - 1 \leq 0$$
$$1 \leq i_1 < i_2 \leq R.$$

$$P(c_{j_1} = 3; \theta) + P(c_{j_2} = 3; \theta) - 1 \leq 0$$
$$1 \leq j_1 < j_2 \leq C.$$

$$P(r_i = 3; \theta) + P(c_j = 3; \theta) - 1 \leq 0$$
$$\forall i \in \{1, \ldots, R\},\ j \in \{1, \ldots, C\}.$$

As explained in Section 6.4, we convert all these probabilistic statements to an auxiliary penalty term, which gets added to the loss function.

A.2 Dataset details

We use the INTERGLAD V7.0 (Interglad) database (NGF, 2019) for annotating our training set as described in Section 7. Since the Interglad database is not publicly available, we use SciGlass (Epam) database (released under Open Database License) as a proxy for Interglad in the shared code. Interglad contains 12634 compositions corresponding to the publications in our training set. However, SciGlass contains only 2347 compositions of these publications. Hence, the code provided by us can annotate a subset of the training data only. However, we do provide training data annotated using the Interglad database for reproducing the results of DISCOMAT for training and evaluation. Also, anyone with Elsevier and Interglad subscriptions can replicate our training set (by replacing SciGlass database files with Interglad database files).

We have manually annotated the val and test set, due to the fact that distantly supervised annotations can have noise and are not always 100% accurate. The inter-annotator agreement has already been discussed in 5. Along with the provision of manual annotation, the in-house annotation tools also contained several checks on conditions that shouldn't arise such as: whether the annotator has missed annotating any table, or the annotator has annotated with out-of-range labels or a row/column having both composition and constituent or vice-versa i.e. composition/constituent present in both row and

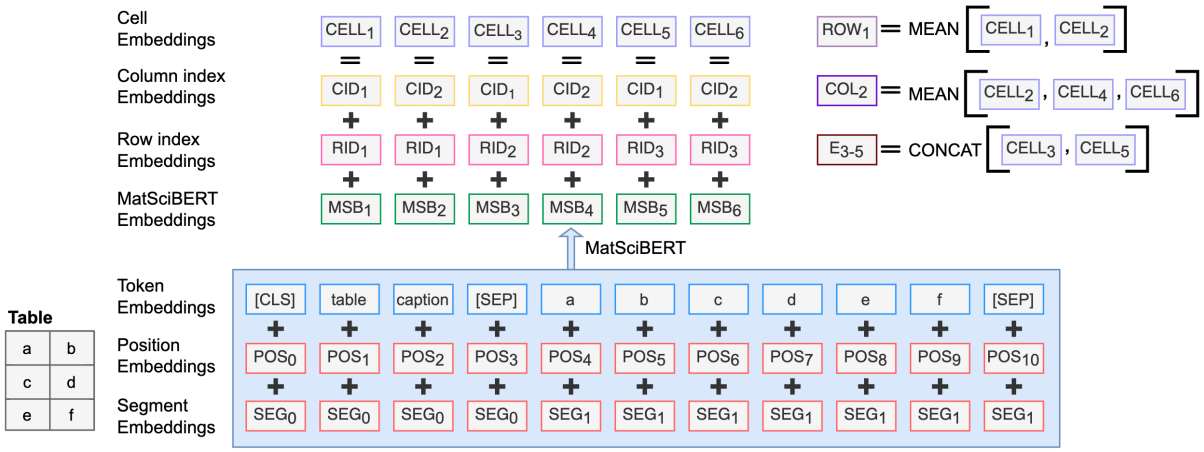

Figure 7: Schematic of TAPAS-ADAPTED baseline model

	Splits		
Table Type	**Train**	**Dev**	**Test**
SCC	704	110	113
MCC-CI	626	132	132
MCC-PI	317	109	112
NC	2761	387	380
Total	4408	738	737

(a)

	Splits		
	Train	**Dev**	**Test**
Publications	1880	330	326
Materials	11207	2873	2649
Tuples	38799	10168	9514

(b)

Table 4: Number of (a) each of the table types and (b) journals from which the tables are obtained, materials in the tables, and the tuples for the three splits.

column of a table. With the help of these self-checks and mutual discussions on disagreements, we annotated our val and test dataset.

Table 4 presents some statistics about our dataset. Table 4a shows the number of tables in our dataset belonging to different table types. Further, Table 4b shows the total number of publications, materials, and tuples in all three splits. We release our code and data under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 (CC BY-NC-SA 4.0) International Public License.

A.3 Baseline models

In this section, we describe the details of our baseline models: TAPAS, TAPAS-ADAPTED, TABERT and TABERT-ADAPTED. Since, the TAPAS (Herzig et al., 2020) architecture has been used for QA over tables and we do not have any questions in the composition extraction task, we use table caption as a proxy for the question. We replace the empty table cells with a special [EMPTY] token. The table caption and text in table cells are converted to word-pieces using the LM tokenizer. Then, we concatenate the word-pieces of the caption and row-wise flattened table. Note that it is possible to obtain more than one word-piece for some table cells. Since the input length after tokenization can be greater than 512, we truncate the minimum possible rows from the end so that the length becomes less than or equal to 512. To avoid a large number of rows getting truncated due to long captions, we truncate the caption so that it only contributes $\leq$ 100 word-pieces. To differentiate between the table cells belonging to different rows and/or columns, row and column index embeddings are added to the word-piece embeddings in the TAPAS architecture. Position and Segment embeddings are the same as in BERT (Devlin et al., 2019), except that position indexes are incremented when the table cell changes. Original TAPAS architecture also involves adding different Rank embeddings to the input in order to answer rank-based questions. We use the same rank embeddings for every table cell since there is no rank relation among the table cells for our case.
All these different types of embeddings are added together and passed through the LM. We take the contextual embedding of the first word-piece of every table cell to be representative of it. Since we do not have row and column nodes here, row and column embeddings are computed by taking the

(a)

Glass composition (mol%)	
$20Li_2O$–$20RO$–$60SiO_2$	R = Mg, Ca, Sr, Ba
$xBaO$–$(100 - x)B_2O_3$	x = 20, 30, 40
$50RO$–$50P_2O_5$	R = Ca, Sr, Ba
xNa_2O–$(30 - x)CaO$–$60P_2O_5$–$10Al_2O_3$	x = 0, 15, 30

(b)

Sample	Composition
1	$0.6TeO_2$–$0.3Tl_{O_{0.5}}$–$0.1ZnO$
4	$0.85TeO_2$–$0.15WO_3$ + 0.1 wt%Ag_2O + 0.076 wt%CeO_2
5	$0.85TeO_2$–$0.15WO_3$ + 0.1 wt% Ag_2O + 0.056 wt%CeO_2
10	$0.85TeO_2$–$0.15WO_3$

(c)

	$0.91[Ge_{0.25}As_{0.1}S_{0.65}]$–$0.05GaS_{3/2}$–$0.04CsBr$	$0.9[Ge_{0.25}As_{0.1}S_{0.65}]$–$0.05GaS_{3/2}$–$0.05CsBr$
τ_R (μs)	504	1170
τ_m (μs)	98	937

Figure 8: Examples of corner case composition tables (a) (Murata et al., 2005) (b) (Duclère et al., 2009) (c) (Shin et al., 2002)

(a)

Sample code	Composition				Molar mass, M (g mol^{-1})	Density, d (g cm^{-3})	Molar Volume, V_m (cm^3 mol^{-1})
	M_xO_y	Al_2O_3	B_2O_3	TeO_2			
5W20BTe	5	–	20	75	145.216	4.999 ± 0.001	29.05
5Nb20BTe	5	–	20	75	146.915	4.877 ± 0.001	30.12
5Pb20BTe	5	–	20	75	144.785	5.212 ± 0.001	27.78

(b)

Glass	Na_2O	Network modifiers		Fe_2O_3	Network formers		MnO	SrO	Other oxides
HLWMS-8 (target)	3.00	CaO	0.00	0.00	Al_2O_3	0.00	13.60	29.90	2.00
		Li_2O	6.00		B_2O_3	12.00			
		K_2O	0.00		SiO_2	33.50			
HLWMS-11 (target)	1.00	CaO	0.00	0.00	Al_2O_3	5.00	13.60	29.90	7.00
		Li_2O	3.50		B_2O_3	10.00			
		K_2O	0.00		SiO_2	30.00			

(c)

MnO Nominal/analyzed	Na_2O Nominal/analyzed	P_2O_5 Nominal/analyzed
0/0	50/46.7	50/53.3
5/4.6	47.5/46.0	47.5/49.6
10/9.6	45/43.2	45/47.2
15/14.7	42.5/41.3	42.5/44.1

Figure 9: Some more examples of corner case composition tables (a) (Kaur et al., 2015) (b) (McKeown et al., 2003) (c) (Omrani et al., 2014)

average of the first word-piece contextual embeddings of cells occurring in that row/column, which are then fed to an MLP for row/column classification. Edge embeddings are computed by concatenating the first workpiece contextual embeddings of source and destination cells.

Figure 7 shows the schematic of TAPAS-ADAPTED model. Here, we initialize LM weights with that of MATSCIBERT (Gupta et al., 2022). All other details are the same as in the TAPAS model, except that here we add row and column index embeddings to MATSCIBERT output, instead of input.

For TABERT also, we use the table caption as the proxy for the NL sentence, concatenate it with linearized rows and feed into the TABERT model which generates cell embedding by passing through BERT and applying vertical attention to propagate information across columns. Following the kind of linearization used by TABERT, we linearize each cell as a concatenation of cell type and cell value for each cell, where cell type is divided into numeric, alphanumeric or text. Since DISCOMAT does not use pretraining, we do not use TABERT'S pretrained weights but instead train from initial weights on our row, column and edge-level prediction tasks. We also implement another baseline called TABERT-ADAPTED, which replaces the BERT encoder in TABERT with MATSCIBERT (Gupta et al., 2022) to provide materials science domain's information to the model.

In TABBIE, as opposed to TAPAS and TABERT, table cells are passed independently into the LM, instead of being linearized/flattened into a single long sequence. Similar to TABERT, we don't initialize TABBIE's architecture with its pretrained weights for a fair comparison. TABBIE-ADAPTED again replaces the BERT encoder in TABERT with MATSCIBERT (Gupta et al., 2022).

The complete code and data is available at https://github.com/M3RG-IITD/DiSCoMaT.

A.4 Implementation details

For Graph Attention Networks (GATs) (Veličković et al., 2018), we use the GAT implementation of Deep Graph Library (Wang et al., 2019). For LMs, TAPAS, we use the implementation by Transformers library (Wolf et al., 2020). We use TABERT's source code from their GitHub repository. We implement and train all models using PyTorch (Paszke et al., 2019) and AllenNLP (Gardner et al., 2017). We optimize the model parameters using Adam (Kingma and Ba, 2015) and a triangular learning rate (Smith, 2017). We further use different learning rates for LM and non-LM parameters (GNNs, MLPs) (App. A.5). To deal with imbalanced labels, we scale loss for all labels by weights inversely proportional to their frequency in the training set. All experiments were run on a machine with one 32 GB V100 GPU. Each model is run with three seeds and the mean and std. deviation is reported.

A.5 Hyper-parameter details

Now, we describe the hyper-parameters of DISCOMAT. Both GNN_1 and GNN_2 can have multiple hidden layers with different numbers of attention heads. We experiment with hidden layer sizes of 256, 128, and 64 and the number of attention heads as 6, 4, and 2. We include residual connections in

GAT, exponential linear unit (ELU) non-linearity after hidden layers, and LeakyRELU non-linearity (with slope α = 0.2) to compute attention weights as done in (Veličković et al., 2018). Training is performed using 8 tables in a batch and we select the checkpoint with the maximum dev MatL F_1 score. We use a triangular learning rate and choose the peak learning rate for LM to be among 1e-5, 2e-5, and 3e-5 and the peak learning rate for non-LM parameters to be among 3e-4 and 1e-3. A warmup ratio of 0.1 is used for all parameters. We further use batch normalization (Ioffe and Szegedy, 2015) and dropout (Srivastava et al., 2014) probability of 0.2 in all MLPs. We use the same λ for every constraint penalty term. Embedding sizes for features are chosen from 128 and 256 and edge loss weight is selected among 0.3 and 1.0.

Hyper-parameter	**GNN$_1$**	**GNN$_2$**
GAT Hidden Layer Sizes	[256, 128, 64]	[128, 128, 64]
GAT Attention Heads	[4, 4, 4]	[6, 4, 4]
Peak LR for LM	1e-5	2e-5
Peak LR for non-LM	3e-4	3e-4
RegEx feature emb size	256	NA
Max-frequency feature emb size	256	128
Constraint penalty (λ)	50.0	30.0
Edge loss weight	NA	1.0

Table 5: Hyper-parameters for DISCOMAT.

A.6 Corner cases

Figure 8 shows examples of some corner case tables. In Figure 8a, elements are being used as variables. Moreover, the values that variables can take are present in a single cell only. Figure 8b shows a table where units occur within the composition itself. Also, mixed units are being used to express the composition. Figure 8c comprises compositions having both elements and compounds. Whereas, we made different REs for element compositions and different REs for compound compositions. Hence our REs are unable to match these.

Figure 9 shows some more examples of corner cases. In Figure 9a, the first compound has to be inferred using the Material IDs. For example, W corresponds to WO_3 and Nb corresponds to Nb_2O_5. DISCOMAT makes the assumption that composition is present in a single row/column. Figure 9b refutes this assumption as compositions are present in multiple rows. Sometimes researchers report both theoretical (nominal) and experimental (analyzed) compositions for the same material. The table in Figure 9c lists both types of compositions in the same cell and hence can't be extracted using DISCOMAT.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Left blank.

☑ A2. Did you discuss any potential risks of your work?
After Conclusion and Acknowledgements section. This is the last paragraph after main text of the paper.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☒ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
We did not use them.

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Section 5 and Appendix A2

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Section 5 and Appendix A2

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Section 5 and Appendix A2

☒ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
The annotators are the co-authors of this paper and it is mentioned in Section 5.

☒ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. ☐

☒ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. ☐