MatSci-NLP: Evaluating Scientific Language Models on Materials Science Language Tasks Using Text-to-Schema Modeling

Yu Song[1,*] **Santiago Miret**[2,*] **Bang Liu**[1,†]
[1]University of Montreal / Mila - Quebec AI, [2]Intel Labs
{yu.song, bang.liu}@umontreal.ca
{santiago.miret}@intel.com

Abstract

We present MatSci-NLP, a natural language benchmark for evaluating the performance of natural language processing (NLP) models on materials science text. We construct the benchmark from publicly available materials science text data to encompass seven different NLP tasks, including conventional NLP tasks like named entity recognition and relation classification, as well as NLP tasks specific to materials science, such as synthesis action retrieval which relates to creating synthesis procedures for materials. We study various BERT-based models pretrained on different scientific text corpora on MatSci-NLP to understand the impact of pretraining strategies on understanding materials science text. Given the scarcity of high-quality annotated data in the materials science domain, we perform our fine-tuning experiments with limited training data to encourage the generalize across MatSci-NLP tasks. Our experiments in this low-resource training setting show that language models pretrained on scientific text outperform BERT trained on general text. MatBERT, a model pretrained specifically on materials science journals, generally performs best for most tasks. Moreover, we propose a unified text-to-schema for multitask learning on MatSci-NLP and compare its performance with traditional fine-tuning methods. In our analysis of different training methods, we find that our proposed text-to-schema methods inspired by question-answering consistently outperform single and multitask NLP fine-tuning methods. The code and datasets are publicly available[1].

1 Introduction

Materials science comprises an interdisciplinary scientific field that studies the behavior, properties and applications of matter that make up materials systems. As such, materials science often requires deep understanding of a diverse set of scientific disciplines to meaningfully further the state of the art. This interdisciplinary nature, along with the great technological impact of materials advances and growing research work at the intersection of machine learning and materials science (Miret et al.; Pilania, 2021; Choudhary et al., 2022), makes the challenge of developing and evaluating natural language processing (NLP) models on materials science text both interesting and exacting.

The vast amount of materials science knowledge stored in textual format, such as journal articles, patents and technical reports, creates a tremendous opportunity to develop and build NLP tools to create and understand advanced materials. These tools could in turn enable faster discovery, synthesis and deployment of new materials into a wide variety of application, including clean energy, sustainable manufacturing and devices.

Understanding, processing, and training language models for scientific text presents distinctive challenges that have given rise to the creation of specialized models and techniques that we review in Section 2. Additionally, evaluating models on scientific language understanding tasks, especially in materials science, often remains a laborious task given the shortness of high-quality annotated data and the lack of broad model benchmarks. As such, NLP research applied to materials science remains in the early stages with a plethora of ongoing research efforts focused on dataset creation, model training and domain specific applications.

The broader goal of this work is to enable the development of pertinent language models that can be applied to further the discovery of new material systems, and thereby get a better sense of how well language models understand the properties and behavior of existing and new materials. As such, we propose MatSci-NLP, a benchmark of various NLP tasks spanning many applications in the materials science domain described in Section 3. We utilize

[*] Equal contribution.

[†] Corresponding author. Canada CIFAR AI Chair.

[1] https://github.com/BangLab-UdeM-Mila/NLP4MatSci-ACL23

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 3621–3639
July 9-14, 2023 ©2023 Association for Computational Linguistics

this benchmark to analyze the performance of various BERT-based models for MatSci-NLP tasks under distinct textual input schemas described in Section 4. Concretely, through this work we make the following research contributions:

- **MatSci-NLP Benchmark:** We construct the first broad benchmark for NLP in the materials science domain, spanning several different NLP tasks and materials applications. The benchmark contents are described in Section 3 with a general summary and data sources provided in Table 1. The processed datasets and code will be released after acceptance of the paper for reproducibility.

- **Text-to-Schema Multitasking:** We develop a set of textual input schemas inspired by question-answering settings for fine-tuning language models. We analyze the models' performance on MatSci-NLP across those settings and conventional single and multitask fine-tuning methods. In conjunction with this analysis, we propose a new Task-Schema input format for joint multitask training that increases task performance for all fine-tuned language models.

- **MatSci-NLP Analysis:** We analyze the performance of various BERT-based models pretrained on different scientific and non-scientific text corpora on the MatSci-NLP benchmark. This analysis help us better understand how different pretraining strategies affect downstream tasks and find that MatBERT (Walker et al., 2021), a BERT model trained on materials science journals, generally performs best reinforcing the importance of curating high-quality pretraining corpora.

We centered our MatSci-MLP analysis on exploring the following questions:

Q1 *How does in-domain pretraining of language models affect the downstream performance on MatSci-NLP tasks?* We investigate the performance of various models pretrained on different kinds of domain-specific text including materials science, general science and general language (BERT (Devlin et al., 2018)). We find that MatBERT generally performs best and that language models pretrained on diverse scientific texts outperform a general language BERT. Interestingly, SciBERT (Beltagy et al., 2019) often outperforms materials science language models, such as MatSciBERT (Gupta et al., 2022) and BatteryBERT (Huang and Cole, 2022).

Q2 *How do in-context data schema and multitasking affect the learning efficiency in low-resource training settings?* We investigate how several input schemas shown in Figure 1 that contain different kinds of information affect various domain-specific language models and propose a new *Task-Schema* method. Our experiments show that our proposed Task-Schema method mostly performs best across all models and that question-answering inspired schema outperform single task and multitask fine-tuning settings.

2 Background

The advent of powerful NLP models has enabled the analysis and generation of text-based data across a variety of domains. BERT (Devlin et al., 2018) was one of the first large-scale transformer-based models to substantially advance the state-of-the-art by training on large amounts of unlabeled text data in a self-supervised way. The pretraining procedure was followed by task-specific fine-tuning, leading to impressive results on a variety of NLP task, such as named entity recognition (NER), question and answering (QA), and relation classification (Hakala and Pyysalo, 2019; Qu et al., 2019; Wu and He, 2019). A significant collection of large language models spanning millions to billions of parameters followed the success of BERT adopting a similar approach of pretraining on vast corpora of text with task-specific fine-tuning to push the state-of-the-art for in natural language processing and understanding (Raffel et al., 2020; Brown et al., 2020; Scao et al., 2022).

2.1 Scientific Language Models

The success of large language models on general text motivated the development of domain-specific language models pretrained on custom text data, including text in the scientific domain: SciBERT (Beltagy et al., 2019), ScholarBERT (Hong et al., 2022) and Galactica (Taylor et al., 2022) are pretrained on general corpus of scientific articles; BioBERT (Lee et al., 2020), PubMedBERT (Gu et al., 2021), BioMegatron (Shin et al., 2020) and Sci-Five (Phan et al., 2021) are pretrained on various kinds of biomedical corpora; MatBERT (Walker et al., 2021), MatSciBERT (Gupta et al., 2022) are pretrained on materials science specific corpora; and BatteryBERT (Huang and Cole, 2022) is pretrained on a corpus focused on batteries. Concurrently, several domain-specific NLP benchmarks were established to assess language model performance on domain-specific tasks, such as QASPER (Dasigi et al., 2021) and BLURB (Gu et al., 2021) in the scientific domain, as well as PubMedQA (Jin et al., 2019), BioASQ (Balikas et al., 2015), and Biomedical Language Understanding Evaluation (BLUE) (Peng et al., 2019) in the biomedical domain.

2.2 NLP in Materials Science

The availability of openly accessible, high-quality corpora of materials science text data remains highly restricted in large part because data from peer-reviewed journals and scientific documents is usually subject to copyright restrictions, while open-domain data is often only available in difficult-to-process PDF formats (Olivetti et al., 2020; Kononova et al., 2021). Moreover, specialized scientific text, such as materials synthesis procedures containing chemical formulas and reaction notation, require advanced data mining techniques for effective processing (Kuniyoshi et al., 2020; Wang et al., 2022b). Given the specificity, complexity, and diversity of specialized language in scientific text, effective extraction and processing remain an active area of research with the goal of building relevant and sizeable text corpora for pretraining scientific language models (Kononova et al., 2021).

Nonetheless, materials science-specific language models, including MatBERT (Walker et al., 2021), MatSciBERT (Gupta et al., 2022), and BatteryBERT (Huang and Cole, 2022), have been trained on custom-built pretraining dataset curated by different academic research groups. The pretrained models and some of the associated fine-tuning data have been released to the public and have enabled further research, including this work.

The nature of NLP research in materials science to date has also been highly fragmented with many research works focusing on distinct tasks motivated by a given application or methodology. Common ideas among many works include the prediction and construction of synthesis routes for a variety of materials (Mahbub et al., 2020; Karpovich et al., 2021; Kim et al., 2020), as well as the creation of novel materials for a given application (Huang and Cole, 2022; Georgescu et al., 2021; Jensen et al., 2021), both of which relate broader challenges in the field of materials science.

3 MatSci-NLP Benchmark

Through the creation of MatSci-NLP, we aim to bring together some of the fragmented data across multiple research works for a wide-ranging materials science NLP benchmark. As described in Section 2, the availability of sizeable, high-quality and diverse datasets remain a major obstacle in applying modern NLP to advance materials science in meaningful ways. This is primarily driven by a high cost of data labeling and the heterogeneous nature of materials science. Given those challenges, we created MatSci-NLP by unifying various publicly available, high-quality, smaller-scale datasets to form a benchmark for fine-tuning and evaluating modern NLP models for materials science applications. MatSci-NLP consists of seven NLP tasks shown in Table 1, spanning a wide range of materials categories including fuel cells (Friedrich et al., 2020), glasses (Venugopal et al., 2021), inorganic materials (Weston et al., 2019; MatSciRE, 2022), superconductors (Yamaguchi et al., 2020), and synthesis procedures pertaining to various kinds of materials (Mysore et al., 2019; Wang et al., 2022a). Some tasks in MatSci-NLP had multiple source components, meaning that the data was curated from multiple datasets (e.g. NER), while many were obtained from a single source dataset.

The data in MatSci-NLP adheres to a standard JSON-based data format with each of the samples containing relevant text, task definitions, and annotations. These can in turn be refactored into different input schemas, such as the ones shown in Figure 1 consisting of 1) *Input*: primary text jointly with task descriptions and instructions, and 2) *Output*: query and label, which we perform in our text-to-schema modeling described in Section 4. Next, we describe the tasks in MatSci-NLP in greater detail:

- **Named Entity Recognition (NER):** The NER task requires models to extract summary-level information from materials science text and recognize entities including materials, descriptors, material properties, and applications amongst others. The NER task predicts the best entity type label for a given text span

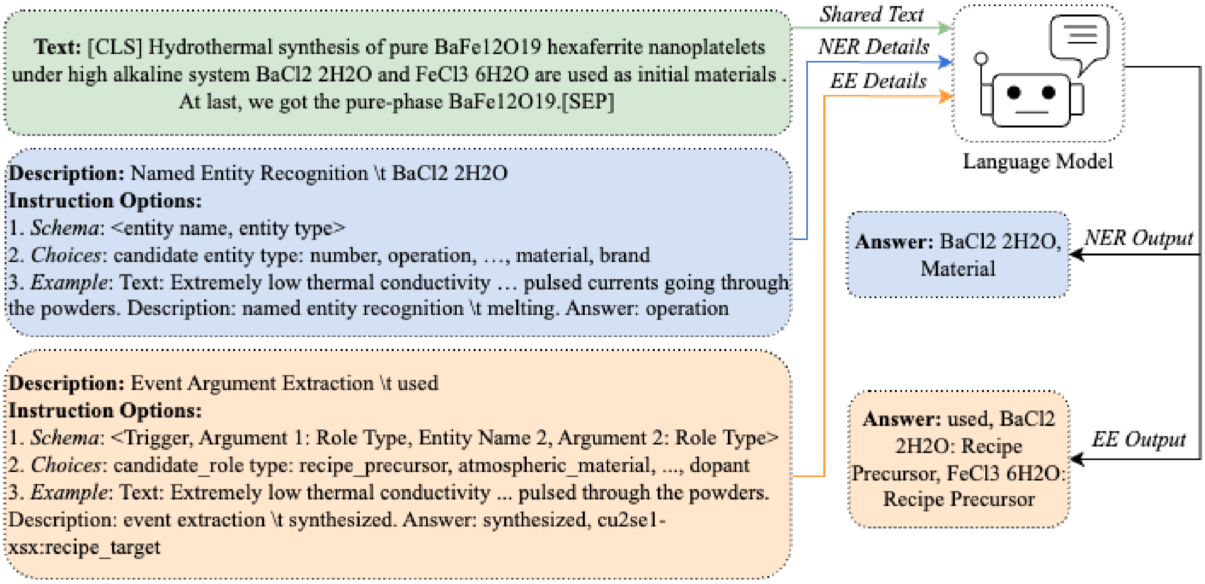
Text: [CLS] Hydrothermal synthesis of pure BaFe12O19 hexaferrite nanoplatelets under high alkaline system BaCl2 2H2O and FeCl3 6H2O are used as initial materials . At last, we got the pure-phase BaFe12O19.[SEP]
Shared Text
NER Details
EE Details
Language Model
Description: Named Entity Recognition \t BaCl2 2H2O
Instruction Options:
1. Schema: <entity name, entity type>
2. Choices: candidate entity type: number, operation, …, material, brand
3. Example: Text: Extremely low thermal conductivity … pulsed currents going through the powders. Description: named entity recognition \t melting. Answer: operation
Answer: BaCl2 2H2O, Material
NER Output
Description: Event Argument Extraction \t used
Instruction Options:
1. Schema: <Trigger, Argument 1: Role Type, Entity Name 2, Argument 2: Role Type>
2. Choices: candidate_role type: recipe_precursor, atmospheric_material, ..., dopant
3. Example: Text: Extremely low thermal conductivity ... pulsed through the powders. Description: event extraction \t synthesized. Answer: synthesized, cu2se1-xsx:recipe_target
Answer: used, BaCl2 2H2O: Recipe Precursor, FeCl3 6H2O: Recipe Precursor
EE Output

Figure 1: Example of different question-answering inspired textual input schemas (Task-Schema , Potential Choices, Example) applied on MatSci-NLP. The input of the language model includes the shared text (green) along with relevant task details (blue for NER and orange for event extraction). The shared text can contain relevant information for multiple tasks and be part of the language model input multiple times.

Task	**Size** (# Samples)	**Meta-Dataset Components**
Named Entity Recognition	112,191	4
Relation Classification	25,674	3
Event Argument Extraction	6,566	2
Paragraph Classification	1,500	1
Synthesis Action Retrieval	5,547	1
Sentence Classification	9,466	1
Slot Filling	8,253	1

Table 1: Collection of NLP tasks in the meta-dataset of the MatSci-NLP Benchmark drawn from Weston et al. (2019); Friedrich et al. (2020); Mysore et al. (2019); Yamaguchi et al. (2020); Venugopal et al. (2021); Wang et al. (2022a); MatSciRE (2022).

s_i with a non-entity span containing a "null" label. MatSci-NLP contains NER task data adapted from Weston et al. (2019); Friedrich et al. (2020); Mysore et al. (2019); Yamaguchi et al. (2020).

- **Relation Classification:** In the relation classification task, the model predicts the most relevant relation type for a given span pair (s_i, s_j). MatSci-NLP contains relation classification task data adapted from Mysore et al. (2019); Yamaguchi et al. (2020); MatSciRE (2022).

- **Event Argument Extraction:** The event argument extraction task involves extracting event arguments and relevant argument roles. As there may be more than a single event for a given text, we specify event triggers and require the language model to extract corresponding arguments and their roles. MatSci-NLP contains event argument extraction task data adapted from Mysore et al. (2019); Yamaguchi et al. (2020).

- **Paragraph Classification:** In the paragraph classification task adapted from Venugopal et al. (2021), the model determines whether a given paragraph pertains to glass science.

- **Synthesis Action Retrieval (SAR):** SAR is a materials science domain-specific task that defines eight action terms that unambiguously identify a type of synthesis action to describe a synthesis procedure. MatSci-NLP adapts SAR data from Wang et al. (2022a) to ask language models to classify word tokens into pre-defined action categories.

- **Sentence Classification:** In the sentence classification task, models identify sentences that describe relevant experimental facts based on data adapted from Friedrich et al. (2020).

- **Slot Filling:** In the slot-filling task, models extract slot fillers from particular sentences based on a predefined set of semantically meaningful entities. In the task data adapted from Friedrich et al. (2020), each sentence describes a single experiment frame for which the model predicts the slots in that frame.

The tasks contained in MatSci-NLP were selected based on publicly available, high-quality annotated materials science textual data, as well as their relevance to applying NLP tools to materials science. Conventional NLP tasks (NER, Relation Classification, Event Argument Extraction, Paragraph Classification, Sentence Classification) enable materials science researchers to better process and understand relevant textual data. Domain specific tasks (SAR, Slot Filling) enable materials science research to solve concrete challenges, such as finding materials synthesis procedures and real-world experimental planning. In the future, we aim to augment to current set of tasks with additional data and introduce novel tasks that address materials science specific challenges with NLP tools.

4 Unified Text-to-Schema Language Modeling

As shown in Figure 1, a given piece of text can include multiple labels across different tasks. Given this multitask nature of the MatSci-NLP benchmark, we propose a new and unified *Task-Schema* multitask modeling method illustrated in Figure 2 that covers all the tasks in the MatSci-NLP dataset. Our approach centers on a unified text-to-schema modeling approach that can predict multiple tasks simultaneously through a unified format. The underlying language model architecture is made up of modular components, including a domain-specific encoder model (e.g. MatBERT, MatSciBERT, SciBERT), and a generic transformer-based decoder, each of which can be easily exchanged with different pretrained domain-specific NLP models. We fine-tune these pretrained language models and the decoder with collected tasks in MatSci-NLP using the procedure described in Section 4.3.

The unified text-to-schema provides a more structured format to training and evaluating language model outputs compared to seq2seq and text-to-text approaches (Raffel et al., 2020; Luong et al., 2015). This is particularly helpful for the tasks in MatSci-NLP given that many tasks can be reformulated as classification problems. NER and Slot Filling, for example, are classifications at the token-level, while event arguments extraction entails the classification of roles of certain arguments. Without a predefined schema, the model relies entirely on unstructured natural language to provide the answer in a seq2seq manner, which significantly increases the complexity of the task and also makes it harder to evaluate performance. The structure imposed by text-to-schema method also simplifies complex tasks, such as event extraction, by enabling the language model to leverage the structure of the schema to predict the correct answer. We utilize the structure of the schema in decoding and evaluating the output of the language models, as described further in Section 4.3 in greater detail.

Moreover, our unified text-to-schema approach alleviates error propagation commonly found in multitask scenarios (Van Nguyen et al., 2022; Lu et al., 2021), enables knowledge sharing across multiple tasks and encourages the fine-tuned language model to generalize across a broader set of text-based instruction scenarios. This is supported by our results shown in Section 5.2 showing text-to-schema outperforming conventional methods.

4.1 Language Model Formulation

The general purpose of our model is to achieve multitask learning by a mapping function (f) between input (x), output (y), and schema (s), i.e., $f(x, s) = y$. Due to the multitasking nature of our setting, both inputs and outputs can originate from different tasks n, i.e. $x = [x_{t1}, x_{t2}, ...x_{tn}]$ and $y = [y_{t1}, y_{t2}, ...y_{tn}]$, all of which fit under a common schema (s). Given the presence of domain-specific materials science language, our model architecture includes a domain-specific BERT encoder and a transformer decoder. All BERT encoders and transformer decoders share the same general architecture, which relies on a self-attention mechanism: Given an input sequence of length N, we compute a set of attention scores, $A = \mathrm{softmax}(\mathrm{QT^K}/(\sqrt{\mathrm{d_k}}))$. Next, we compute the weighted sum of the value vectors, $O = AV$, where Q, K, and V are the query, key, and value matrices, and d_k is the dimensionality of the key vectors.

Additionally, the transformer based decoder dif-

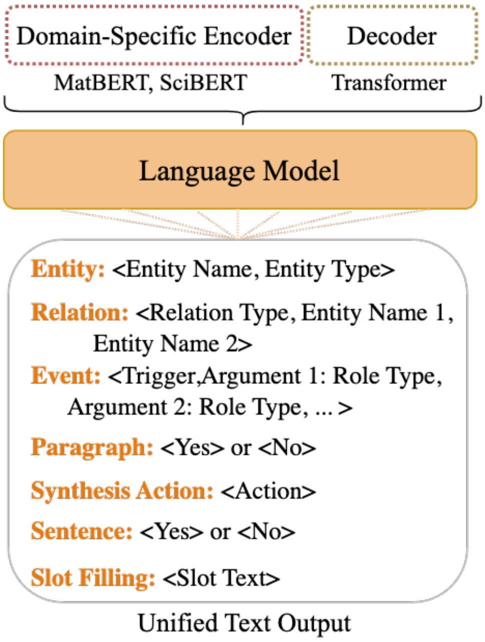
Domain-Specific Encoder
Decoder
MatBERT, SciBERT
Transformer
Language Model
Entity: <Entity Name, Entity Type>
Relation: <Relation Type, Entity Name 1, Entity Name 2>
Event: <Trigger,Argument 1: Role Type, Argument 2: Role Type, ... >
Paragraph: <Yes> or <No>
Synthesis Action: <Action>
Sentence: <Yes> or <No>
Slot Filling: <Slot Text>
Unified Text Output

Figure 2: Unified text-to-schema method for MatSci-NLP text understanding applied across the seven tasks. The language model includes a domain specific encoder, which can be exchanged in a modular manner, as well as a general language pretrained transformer decoder.

fer from the domain specific encoder by: 1) Applying masking based on the schema applied to ensure that it does not attend to future positions in the output sequence. 2) Applying both self-attention and encoder-decoder attention to compute attention scores that weigh the importance of different parts of the output sequence and input sequence. The output of the self-attention mechanism (O_1) and the output of the encoder-decoder attention mechanism (O_2) are concatenated and linearly transformed to obtain a new hidden state, $H = \tanh(\mathrm{W_o}[\mathrm{O_1}; \mathrm{O_2}] + \mathrm{b_o})$ with W_o and b_o being the weight and biases respectively. The model then applies a *softmax* to H to generate the next element in the output sequence $P = \mathrm{softmax}(\mathrm{W_p H} + \mathrm{b_p})$, where P is a probability distribution over the output vocabulary.

4.2 Text-To-Schema Modeling

As shown in Figure 1, our schema structures the text data based on four general components: text, description, instruction options, and the predefined answer schema.

- **Text** specifies raw text from the literature that is given as input to the language model.

- **Description** describes the task for a given text according to a predefined schema containing the task name and the task arguments.

- **Instruction Options** contains the core explanation related to the task with emphasis on three different types: 1) Potential choices of answers; 2) Example of an input/output pair corresponding to the task; 3) Task-Schema : our predefined answer schema illustrated in Figure 2.

- **Answer** describes the correct label of each task formatted as a predefined answer schema that can be automatically generated based on the data structure of the task.

4.3 Language Decoding & Evaluation

Evaluating the performance of the language model on MatSci-NLP requires determining if the text generated by the decoder is valid and meaningful in the context of a given task. To ensure consistency in evaluation, we apply a constrained decoding procedure consisting of two steps: 1) Filtering out invalid answers through the predefined answer schema shown in Figure 2 based on the structure of the model's output; 2) Match the model's prediction with the most similar valid class given by the annotation for the particular task. For example, if for the NER task shown in Figure 1 the model's predicted token is "BaCl2 2H2O materials", it will be matched with the NER label of "material", which is then used as the final prediction for computing losses and evaluating performance. This approach essentially reformulates each task as a classification problem where the classes are provided based on the labels from the tasks in MatSci-NLP. We then apply a cross-entropy loss for model fine-tuning based on the matched label from the model output. The matching procedure simplifies the language modeling challenge by not requiring an exact match of the predicted tokens with the task labels. This in turns leads to a more comprehensible signal in the fine-tuning loss function.

5 Evaluation and Results

Our analysis focuses on the questions outlined in Section 1: 1) Studying the effectiveness of domain-specific language models as encoders, and 2) Analyzing the effect of different input schemas in resolving MatSci-NLP tasks. Concretely, we study the performance of the language models and language schema in a *low resource* setting where we

NLP Model	Named Entity Recognition	Relation Classification	Event Argument Extraction	Paragraph Classification	Synthesis Action Retrieval	Sentence Classification	Slot Filling	Overall (All Tasks)
MatSciBERT (Gupta et al., 2022)	$0.707_{\pm 0.076}$ $0.470_{\pm 0.092}$	$0.791_{\pm 0.046}$ $0.507_{\pm 0.073}$	$0.436_{\pm 0.066}$ $0.251_{\pm 0.075}$	$0.719_{\pm 0.116}$ $0.623_{\pm 0.183}$	$0.692_{\pm 0.179}$ $0.484_{\pm 0.254}$	$0.914_{\pm 0.008}$ $0.660_{\pm 0.079}$	$0.436_{\pm 0.142}$ $0.194_{\pm 0.062}$	$0.671_{\pm 0.060}$ $0.456_{\pm 0.042}$
MatBERT (Walker et al., 2021)	$0.875_{\pm 0.015}$ $0.630_{\pm 0.047}$	$0.804_{\pm 0.071}$ $0.513_{\pm 0.138}$	$0.451_{\pm 0.091}$ $0.288_{\pm 0.066}$	$0.756_{\pm 0.073}$ $0.691_{\pm 0.188}$	$0.717_{\pm 0.040}$ $0.549_{\pm 0.091}$	$0.909_{\pm 0.009}$ $0.614_{\pm 0.134}$	$0.548_{\pm 0.058}$ $0.273_{\pm 0.051}$	$0.722_{\pm 0.023}$ $0.517_{\pm 0.041}$
BatteryBERT (Huang and Cole, 2022)	$0.786_{\pm 0.113}$ $0.472_{\pm 0.150}$	$0.801_{\pm 0.081}$ $0.466_{\pm 0.111}$	$0.457_{\pm 0.024}$ $0.277_{\pm 0.034}$	$0.633_{\pm 0.075}$ $0.610_{\pm 0.046}$	$0.614_{\pm 0.128}$ $0.419_{\pm 0.149}$	$0.912_{\pm 0.015}$ $0.684_{\pm 0.095}$	$0.520_{\pm 0.057}$ $0.224_{\pm 0.073}$	$0.663_{\pm 0.038}$ $0.456_{\pm 0.048}$
SciBERT (Beltagy et al., 2019)	$0.734_{\pm 0.079}$ $0.497_{\pm 0.091}$	$0.819_{\pm 0.067}$ $0.545_{\pm 0.119}$	$0.451_{\pm 0.077}$ $0.276_{\pm 0.080}$	$0.696_{\pm 0.094}$ $0.546_{\pm 0.243}$	$0.701_{\pm 0.138}$ $0.516_{\pm 0.217}$	$0.911_{\pm 0.017}$ $0.617_{\pm 0.143}$	$0.481_{\pm 0.144}$ $0.224_{\pm 0.010}$	$0.685_{\pm 0.056}$ $0.460_{\pm 0.044}$
ScholarBERT (Hong et al., 2022)	$0.168_{\pm 0.067}$ $0.101_{\pm 0.034}$	$0.428_{\pm 0.148}$ $0.274_{\pm 0.110}$	$0.489_{\pm 0.083}$ $0.356_{\pm 0.109}$	$0.663_{\pm 0.032}$ $0.433_{\pm 0.122}$	$0.322_{\pm 0.260}$ $0.178_{\pm 0.051}$	$0.906_{\pm 0.007}$ $0.478_{\pm 0.008}$	$0.296_{\pm 0.085}$ $0.109_{\pm 0.044}$	$0.468_{\pm 0.028}$ $0.276_{\pm 0.024}$
BioBERT (Wada et al., 2020)	$0.715_{\pm 0.031}$ $0.459_{\pm 0.055}$	$0.797_{\pm 0.092}$ $0.465_{\pm 0.134}$	$0.488_{\pm 0.036}$ $0.274_{\pm 0.049}$	$0.675_{\pm 0.144}$ $0.578_{\pm 0.102}$	$0.647_{\pm 0.140}$ $0.446_{\pm 0.231}$	$0.915_{\pm 0.021}$ $0.686_{\pm 0.098}$	$0.452_{\pm 0.114}$ $0.191_{\pm 0.045}$	$0.670_{\pm 0.061}$ $0.442_{\pm 0.057}$
BERT (Devlin et al., 2018)	$0.657_{\pm 0.077}$ $0.461_{\pm 0.058}$	$0.782_{\pm 0.056}$ $0.494_{\pm 0.061}$	$0.418_{\pm 0.053}$ $0.225_{\pm 0.091}$	$0.665_{\pm 0.057}$ $0.532_{\pm 0.194}$	$0.656_{\pm 0.099}$ $0.515_{\pm 0.067}$	$0.910_{\pm 0.017}$ $0.633_{\pm 0.133}$	$0.520_{\pm 0.019}$ $0.257_{\pm 0.022}$	$0.658_{\pm 0.030}$ $0.439_{\pm 0.021}$

Table 2: Low-resource fine-tuning results applying unified Task-Schema setting for various BERT-based encoder models pretrained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We denote the best performing encoder model and those that outperform the general language BERT according to the micro-f1 with orange shading with MatBERT and SciBERT performing best on most tasks and ScholarBERT and general language BERT generally performing worst.

perform fine-tuning on different pretrained BERT models with limited data from the MatSci-NLP benchmark. This low-resource setting makes the learning problem harder given that the model has to generalize on little amount of data. Moreover, this setting approximates model training with very limited annotated data, which is commonly found in materials science as discussed in Section 2. In our experiments, we split the data in MatSci-NLP into 1% training subset and a 99% testing subset for evaluation. None of the evaluated encoder models were exposed to the fine-tuning data in advance of our experiments and therefore have to rely on the knowledge acquired during their respective pretraining processes. We evaluate the results of our experiments using micro-F1 and macro-F1 scores of the language model predictions on the test split of the MatSci-NLP that were not exposed during fine-tuning.

5.1 How does in-domain pretraining of language models affect the downstream performance on MatSci-NLP tasks? (Q1)

Based on the results shown in Table 2, we can gather the following insights:

First, domain-specific pretraining affects model performance. We perform fine-tuning on various models pretrained on domain-specific corpora in a low-resource setting and observe that: i) MatBert, which was pretrained on textual data from materials science journals, generally performs best for most tasks in the MatSci-NLP benchmark with SciBERT generally performing second best. The high performance of MatBERT suggests that materials science specific pretraining does help the language models acquire relevant materials science knowledge. Yet, the underperformance of MatSciBERT compared to MatBERT and SciBERT indicates that the curation of pretraining data does significantly affect performance. ii) The importance of the pretraining corpus is further reinforced by the difference in performance between SciBERT and ScholarBERT, both of which were trained on corpora of general scientific text, but show vastly different results. In fact, ScholarBERT underperforms all other models, including the general language BERT, for all tasks except event argument extraction where ScholarBERT performs best compared to all other models. iii) The fact that most scientific BERT models outperform BERT pretrained on general language suggests that pretraining on high-quality scientific text is beneficial for resolving tasks involving materials science text and potentially scientific texts from other domains. This notion of enhanced performance on MatSci-NLP when pretraining on scientific text is further reinforced by the performance of BioBERT by Wada et al. (2020). BioBERT outperforms BERT on most tasks even though it was trained on text from the biomedical domain that has minor overlap with the materials science domain. This strongly indicates that scientific language, regardless of the domain, has a significant distribution shift from general language that is used to pretrain common language models.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Task-Schema
MatSciBERT (Gupta et al., 2022)	$0.501_{\pm0.057}$ $0.320_{\pm0.078}$	$0.485_{\pm0.043}$ $0.238_{\pm0.017}$	$0.457_{\pm0.021}$ $0.228_{\pm0.038}$	$0.651_{\pm0.045}$ $0.438_{\pm0.052}$	$0.670_{\pm0.036}$ $0.435_{\pm0.061}$	$0.688_{\pm0.045}$ $0.463_{\pm0.040}$	$0.671_{\pm0.060}$ $0.456_{\pm0.042}$
MatBERT (Walker et al., 2021)	$0.537_{\pm0.036}$ $0.330_{\pm0.063}$	$0.523_{\pm0.021}$ $0.267_{\pm0.014}$	$0.557_{\pm0.010}$ $0.301_{\pm0.006}$	$0.721_{\pm0.033}$ $0.514_{\pm0.045}$	$0.699_{\pm0.020}$ $0.478_{\pm0.032}$	$0.705_{\pm0.025}$ $0.470_{\pm0.029}$	$0.722_{\pm0.023}$ $0.517_{\pm0.041}$
BatteryBERT (Huang and Cole, 2022)	$0.469_{\pm0.050}$ $0.288_{\pm0.055}$	$0.488_{\pm0.011}$ $0.241_{\pm0.009}$	$0.431_{\pm0.044}$ $0.200_{\pm0.022}$	$0.660_{\pm0.013}$ $0.450_{\pm0.031}$	$0.622_{\pm0.069}$ $0.423_{\pm0.039}$	$0.660_{\pm0.033}$ $0.416_{\pm0.054}$	$0.663_{\pm0.038}$ $0.456_{\pm0.048}$
SciBERT (Beltagy et al., 2019)	$0.500_{\pm0.055}$ $0.300_{\pm0.080}$	$0.502_{\pm0.030}$ $0.248_{\pm0.015}$	$0.504_{\pm0.052}$ $0.275_{\pm0.031}$	$0.680_{\pm0.066}$ $0.458_{\pm0.060}$	$0.660_{\pm0.042}$ $0.435_{\pm0.061}$	$0.686_{\pm0.039}$ $0.460_{\pm0.042}$	$0.685_{\pm0.056}$ $0.460_{\pm0.044}$
ScholarBERT (Hong et al., 2022)	$0.472_{\pm0.137}$ $0.234_{\pm0.094}$	$0.429_{\pm0.258}$ $0.250_{\pm0.142}$	$0.367_{\pm0.075}$ $0.165_{\pm0.044}$	$0.461_{\pm0.016}$ $0.271_{\pm0.022}$	$0.513_{\pm0.041}$ $0.295_{\pm0.055}$	$0.467_{\pm0.019}$ $0.260_{\pm0.018}$	$0.468_{\pm0.028}$ $0.276_{\pm0.024}$
BioBERT (Wada et al., 2020)	$0.487_{\pm0.059}$ $0.281_{\pm0.026}$	$0.488_{\pm0.032}$ $0.238_{\pm0.017}$	$0.360_{\pm0.007}$ $0.151_{\pm0.002}$	$0.663_{\pm0.044}$ $0.442_{\pm0.079}$	$0.587_{\pm0.022}$ $0.365_{\pm0.018}$	$0.632_{\pm0.040}$ $0.404_{\pm0.046}$	$0.670_{\pm0.061}$ $0.442_{\pm0.057}$
BERT (Devlin et al., 2018)	$0.498_{\pm0.051}$ $0.266_{\pm0.044}$	$0.488_{\pm0.043}$ $0.239_{\pm0.011}$	$0.394_{\pm0.009}$ $0.166_{\pm0.008}$	$0.670_{\pm0.020}$ $0.440_{\pm0.052}$	$0.601_{\pm0.046}$ $0.382_{\pm0.039}$	$0.636_{\pm0.052}$ $0.394_{\pm0.051}$	$0.658_{\pm0.030}$ $0.439_{\pm0.021}$
Overall (All Models)	$0.493_{\pm0.064}$ $0.288_{\pm0.063}$	$0.486_{\pm0.062}$ $0.246_{\pm0.032}$	$0.439_{\pm0.003}$ $0.212_{\pm0.022}$	$0.644_{\pm0.034}$ $0.430_{\pm0.049}$	$0.622_{\pm0.035}$ $0.402_{\pm0.049}$	$0.639_{\pm0.044}$ $0.410_{\pm0.043}$	$0.688_{\pm0.046}$ $0.435_{\pm0.039}$

Table 3: Consolidated results among all MatSci-NLP tasks on different training settings for various BERT-based encoder models pretrained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the performance of different schema according to heatmap ranging from best and worst. The concentration of red hues on right side indicates that the question-answering inspiring schema generally outperform conventional fine-tuning method. Our proposed Task-Schema generally outperforms all other schemas across most enconder models.

Second, imbalanced datasets in MatSci-NLP skew performance metrics: We can see from Table 2 that the micro-F1 scores are significantly higher than the macro-f1 across all tasks. This indicates that the datasets used in the MatSci-NLP are consistently imbalanced, including in the binary classification tasks, and thereby push the micro-F1 higher compared to the macro-F1 score. In the case of paragraph classification, for example, the number of positive examples is 492 compared with the total number of 1500 samples. As such, only models with a micro-F1 score above 0.66 and macro-F1 above 0.5 can be considered to have semantically meaningful understanding of the task. This is even more pronounced for sentence classification where only $876/9466 \approx 10\%$ corresponds to one label. All models except ScholarBERT outperform a default guess of the dominant class for cases. While imbalanced datasets may approximate some real-world use cases of materials science text analysis, such as extracting specialized materials information, a highly imbalanced can be misguiding in evaluating model performance.

To alleviate the potentially negative effects of imbalanced data, we suggest three simple yet effective methods: 1) Weighted loss functions: This involves weighting the loss function to give higher weights to minority classes. Focal loss (Lin et al., 2017), for example, is a loss function that dynamically modulates the loss based on the prediction confidence, with greater emphasis on more difficult examples. As such, Focal loss handles class imbalance well due to the additional attention given to hard examples of the minority classes. 2) Class-balanced samplers: Deep learning frameworks, such as Pytorch, have class-balanced batch samplers that can be used to oversample minority classes within each batch during training, which can help indirectly address class imbalance. 3) Model architecture tweaks: The model architecture and its hyper-parameters can be adjusted to place greater emphasis on minority classes. For example, one can apply separate prediction heads for minority classes or tweak L2 regularization and dropout to behave differently for minority and majority classes.

5.2 How do in-context data schema and multitasking affect the learning efficiency in low-resource training settings? (Q2)

To assess the efficacy of the proposed textual schemas shown in Figure 1, we evaluate four different QA-inspired schemas: 1) *No Explanations* - here the model receives only the task description; 2) *Potential Choices* - here the model receives the class labels given by the task; 3) *Examples* - here the model receives an example of a correct answer, 4) *Task-Schema* - here the model receives our proposed textual schema. We compare the schemas to three conventional fine-tuning methods: 1) *Single Task* - the traditional method to solve each task separately using the language model and a classification head; 2) *Single Task Prompt* - here we change the format of the task to the same QA-format as "No Explanations", but train each task separately; 3) *MMOE* by Ma et al. (2018) uses multiple encoders to learn multiple hidden embeddings, which are then weighed by a task-specific gate unit and aggregated to the final hidden embedding using a weighted sum for each task. Next, a task-specific classification head outputs the label probability distribution for each task.

Based on the results shown in Table 3, we gather the following insights:

First, Text-to-Schema methods perform better for all language models. Overall, the Task-Schema method we proposed performs best across all tasks in the MatSci-NLP benchmark. The question-answering inspired schema ("No Explanations", "Potential Choices", "Examples", "Task-Schema ") perform better than fine-tuning in a traditional single task setting, single task prompting, as well as fine-tuning using the MMOE multitask method. This holds across all models for all the tasks in MatSci-NLP showing the efficacy of structured language modeling inspired by question-answering.

Second, schema design affects model performance. The results show that both the pretrained model and the input format affect performance. This can be seen by the fact that while all scientific models outperform general language BERT using the Task-Schema method, BERT outperforms some models, mainly ScholarBERT and BioBERT, in the other text-to-schema settings and the conventional training settings. Nevertheless, BERT underperforms the stronger models (MatBERT, SciBERT, MatSciBERT) across all schema settings for all tasks in MatSci-NLP, further emphasizing the importance of domain-specific model pretraining for materials science language understanding.

6 Conclusion and Future Works

We proposed MatSci-NLP, the first broad benchmark on materials science language understanding tasks constructed from publicly available data. We further proposed text-to-schema multitask modeling to improve the model performance in low-resource settings. Leveraging MatSci-NLP and text-to-schema modeling, we performed an in-depth analysis of the performance of various scientific language models and compare text-to-schema language modeling methods with other input schemas, guided by (Q1) addressing the pretrained models and (Q2) addressing the textual schema. Overall, we found that the choice of pretrained models matters significantly for downstream performance on MatSci-NLP tasks and that pretrained language models on scientific text of any kind often perform better than pretrained language models on general text. MatBERT generally performed best, highlighting the benefits of pretraining with high-quality domain-specific language data. With regards to the textual schema outlined in (Q2), we found that significant improvements can be made by improving textual schema showcasing the potential of fine-tuning using structured language modeling.

The proposed encoder-decoder architecture, as well as the proposed multitask schema, could also be useful for additional domains in NLP, including both scientific and non-scientific domains. The potential for open-domain transferability of our method is due to: 1) Our multitask training method and associated schemas do not depend on any domain-specific knowledge, allowing them to be easily transferred to other domains. 2) The encoder of our proposed model architecture can be exchanged in a modular manner, which enables our model structure to be applied across multiple domains. 3) If the fine-tuning data is diverse across a wide range of domains, our method is likely to learn general language representations for open-domain multitask problems. Future work could build upon this paper by applying the model and proposed schema to different scientific domains where fine-tuning data might be sparse, such as biology, physics and chemistry. Moreover, future work can build upon the proposed schema by suggesting novel ways of modeling domain-specific or general language that lead to improvements in unified multi-task learning.

Limitations

One of the primary limitations of NLP modeling in materials science, including this work, is the low quantity of available data as discussed in Section 2. This analysis is affected by this limitation as well given that our evaluations were performed in a low-data setting within a dataset that was already limited in size. We believe that future work can improve upon this study by applying larger datasets, both in the number of samples and in the scope of tasks, to similar problem settings. The small nature of the datasets applied in this study also presents the danger that some of the models may have memorized certain answers instead of achieving a broader understanding, which could be mitigated by enlarging the datasets and making the tasks more complex.

Moreover, we did not study the generalization of NLP models beyond the materials science domain, including adjacent domains such as chemistry and physics. This targeted focus was intentional but imposes limitations on whether the proposed techniques and insights we gained from our analysis are transferable to other domains, including applying NLP models for scientific tasks outside of materials science.

Another limitation of our study is the fact that we focused on BERT-based models exclusively and did not study autoregressive models, including large language models with billions of parameters highlighted in the introduction. The primary reason for focusing on BERT-based models was the diversity of available models trained on different scientific text corpora. Large autoregressive models, on the other hand, are mostly trained on general text corpora with some notable exceptions, such as Galactica (Taylor et al., 2022). We believe that future work analyzing a greater diversity of language models, including large autoregressive models pretrained on different kinds of text, would significantly strengthen the understanding surrounding the ability of NLP models to perform text-based tasks in materials science.

While the results presented in this study indicate that domain-specific pretraining can lead to noticeable advantages in downstream performance on text-based materials science tasks, we would like to highlight the associated risks and costs of pretraining a larger set of customized language models for different domains. The heavy financial and environmental costs associated with these pretraining procedures merit careful consideration of what conditions may warrant expensive pretraining and which ones may not. When possible, we encourage future researchers to build upon existing large models to mitigate the pretraining costs.

Broader Impacts and Ethics Statement

Our MatSci-NLP benchmark can help promote the research on NLP for material science, an important and growing research field. We expect that the experience we gained from the material science domain can be transferred to other domains, such as biology, health, and chemistry. Our Text-to-Schema also helps with improving NLP tasks' performance in low-resource situations, which is a common challenge in many fields.

Our research does not raise major ethical concerns.

Acknowlegments

This work is supported by the Mila internal funding - Program P2-V1: Industry Sponsored Academic Labs (project number: 10379), the Canada CIFAR AI Chair Program, and the Canada NSERC Discovery Grant (RGPIN-2021-03115).

References

Georgios Balikas, Anastasia Krithara, Ioannis Partalas, and George Paliouras. 2015. Bioasq: A challenge on large-scale biomedical semantic indexing and question answering. In *International Workshop on Multimodal Retrieval in the Medical Domain*, pages 26–39. Springer.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Kamal Choudhary, Brian DeCost, Chi Chen, Anubhav Jain, Francesca Tavazza, Ryan Cohn, Cheol Woo Park, Alok Choudhary, Ankit Agrawal, Simon JL Billinge, et al. 2022. Recent advances and applications of deep learning methods in materials science. *npj Computational Materials*, 8(1):1–26.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. *arXiv preprint arXiv:2105.03011*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Annemarie Friedrich, Heike Adel, Federico Tomazic, Johannes Hingerl, Renou Benteau, Anika Maruscyk, and Lukas Lange. 2020. The sofc-exp corpus and neural approaches to information extraction in the materials science domain. *arXiv preprint arXiv:2006.03039*.

Alexandru B Georgescu, Peiwen Ren, Aubrey R Toland, Shengtong Zhang, Kyle D Miller, Daniel W Apley, Elsa A Olivetti, Nicholas Wagner, and James M Rondinelli. 2021. Database, features, and machine learning model to identify thermally driven metal–insulator transition compounds. *Chemistry of Materials*, 33(14):5591–5605.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

Tanishq Gupta, Mohd Zaki, NM Krishnan, et al. 2022. Matscibert: A materials domain language model for text mining and information extraction. *npj Computational Materials*, 8(1):1–11.

Kai Hakala and Sampo Pyysalo. 2019. Biomedical named entity recognition with multilingual bert. In *Proceedings of the 5th workshop on BioNLP open shared tasks*, pages 56–61.

Zhi Hong, Aswathy Ajith, Gregory Pauloski, Eamon Duede, Carl Malamud, Roger Magoulas, Kyle Chard, and Ian Foster. 2022. Scholarbert: Bigger is not always better. *arXiv preprint arXiv:2205.11342*.

Shu Huang and Jacqueline M Cole. 2022. Batterybert: A pretrained language model for battery database enhancement. *Journal of Chemical Information and Modeling*.

Zach Jensen, Soonhyoung Kwon, Daniel Schwalbe-Koda, Cecilia Paris, Rafael Gómez-Bombarelli, Yuriy Román-Leshkov, Avelino Corma, Manuel Moliner, and Elsa A Olivetti. 2021. Discovering relationships between osdas and zeolites through data mining and generative neural networks. *ACS central science*, 7(5):858–867.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*.

Christopher Karpovich, Zach Jensen, Vineeth Venugopal, and Elsa Olivetti. 2021. Inorganic synthesis reaction condition prediction with generative machine learning. *arXiv preprint arXiv:2112.09612*.

Edward Kim, Zach Jensen, Alexander van Grootel, Kevin Huang, Matthew Staib, Sheshera Mysore, Haw-Shiuan Chang, Emma Strubell, Andrew McCallum, Stefanie Jegelka, et al. 2020. Inorganic materials synthesis planning with literature-trained neural networks. *Journal of chemical information and modeling*, 60(3):1194–1201.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Olga Kononova, Tanjin He, Haoyan Huo, Amalie Trewartha, Elsa A Olivetti, and Gerbrand Ceder. 2021. Opportunities and challenges of text mining in materials research. *Iscience*, 24(3):102155.

Fusataka Kuniyoshi, Kohei Makino, Jun Ozawa, and Makoto Miwa. 2020. Annotating and extracting synthesis process of all-solid-state batteries from scientific literature. *arXiv preprint arXiv:2002.07339*.

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

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2event: Controllable sequence-to-structure generation for end-to-end event extraction. *arXiv preprint arXiv:2106.09232*.

Minh-Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2015. Multi-task sequence to sequence learning. *arXiv preprint arXiv:1511.06114*.

Jiaqi Ma, Zhe Zhao, Xinyang Yi, Jilin Chen, Lichan Hong, and Ed H Chi. 2018. Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1930–1939.

Rubayyat Mahbub, Kevin Huang, Zach Jensen, Zachary D Hood, Jennifer LM Rupp, and Elsa A Olivetti. 2020. Text mining for processing conditions of solid-state battery electrolytes. *Electrochemistry Communications*, 121:106860.

MatSciRE. 2022. Material science relation extraction (matscire).

Santiago Miret, Marta Skreta, Benjamin Sanchez-Lengelin, Shyue Ping Ong, Zamyla Morgan-Chan, and Alan Aspuru-Guzik. Ai4mat - neurips 2022.

Sheshera Mysore, Zach 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. *arXiv preprint arXiv:1905.06939*.

Elsa A Olivetti, Jacqueline M Cole, Edward Kim, Olga Kononova, Gerbrand Ceder, Thomas Yong-Jin Han, and Anna M Hiszpanski. 2020. Data-driven materials research enabled by natural language processing and information extraction. *Applied Physics Reviews*, 7(4):041317.

Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 2019 Workshop on Biomedical Natural Language Processing (BioNLP 2019)*.

Long N Phan, James T Anibal, Hieu Tran, Shaurya Chanana, Erol Bahadroglu, Alec Peltekian, and Grégoire Altan-Bonnet. 2021. Scifive: a text-to-text transformer model for biomedical literature. *arXiv preprint arXiv:2106.03598*.

Ghanshyam Pilania. 2021. Machine learning in materials science: From explainable predictions to autonomous design. *Computational Materials Science*, 193:110360.

Chen Qu, Liu Yang, Minghui Qiu, W Bruce Croft, Yongfeng Zhang, and Mohit Iyyer. 2019. Bert with history answer embedding for conversational question answering. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 1133–1136.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Hoo-Chang Shin, Yang Zhang, Evelina Bakhturina, Raul Puri, Mostofa Patwary, Mohammad Shoeybi, and Raghav Mani. 2020. Biomegatron: Larger biomedical domain language model. *arXiv preprint arXiv:2010.06060*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374.

Vineeth Venugopal, Sourav Sahoo, Mohd Zaki, Manish Agarwal, Nitya Nand Gosvami, and NM Anoop Krishnan. 2021. Looking through glass: Knowledge discovery from materials science literature using natural language processing. *Patterns*, 2(7):100290.

Shoya Wada, Toshihiro Takeda, Shiro Manabe, Shozo Konishi, Jun Kamohara, and Yasushi Matsumura. 2020. A pre-training technique to localize medical bert and enhance biobert.

Nicholas Walker, Amalie Trewartha, Haoyan Huo, Sanghoon Lee, Kevin Cruse, John Dagdelen, Alexander Dunn, Kristin Persson, Gerbrand Ceder, and Anubhav Jain. 2021. The impact of domain-specific pre-training on named entity recognition tasks in materials science. *Available at SSRN 3950755*.

Zheren Wang, Kevin Cruse, Yuxing Fei, Ann Chia, Yan Zeng, Haoyan Huo, Tanjin He, Bowen Deng, Olga Kononova, and Gerbrand Ceder. 2022a. Ulsa: Unified language of synthesis actions for the representation of inorganic synthesis protocols. *Digital Discovery*.

Zheren Wang, Olga Kononova, Kevin Cruse, Tanjin He, Haoyan Huo, Yuxing Fei, Yan Zeng, Yingzhi Sun, Zijian Cai, Wenhao Sun, et al. 2022b. Dataset of solution-based inorganic materials synthesis procedures extracted from the scientific literature. *Scientific Data*, 9(1):1–11.

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.

Shanchan Wu and Yifan He. 2019. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2361–2364.

Kyosuke Yamaguchi, Ryoji Asahi, and Yutaka Sasaki. 2020. Sc-comics: a superconductivity corpus for materials informatics. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 6753–6760.

Appendix

A Experimental Details

We performed fine-tuning experiments using a single GPU with a learning rate was 2e-5, the hidden size of the encoders being 768, except ScholarBERT which is 1024, using the Adam (Kingma and Ba, 2014) optimizer for a max number of 20 training epochs with early stopping. All models are implemented with Python and PyTorch, and repeated five times to report the average performance. The full set of hyperparameters is available in our publicaly released code at https://github.com/BangLab-UdeM-Mila/NLP4MatSci-ACL23.

B Additional Text-to-Schema Experiments

To arrive at our data presented in Table 3, we conducted experiments for all the language models across all tasks in MatSci-NLP. The results for seven tasks in MatSci-NLP are shown in subsequent tables:

- Named Entity Recognition in Table 4.
- Relation Classification in Table 5.
- Event Argument Extraction in Table 6.
- Paragraph Classification in Table 7.
- Synthesis Action Retrieval in Table 8.
- Sentence Classification in Table 9.
- Slot Filling in Table 10.

The experimental results summarized in the aforementioned tables reinforce the conclusions in our analysis of (Q2) in Section 5.2 with the text-to-schema based fine-tuning method generally outperforming the conventional single and multitask methods across all tasks and all language models.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	$0.690_{\pm 0.018}$	$0.707_{\pm 0.089}$	$0.451_{\pm 0.114}$	$0.655_{\pm 0.066}$	$0.732_{\pm 0.048}$	$0.753_{\pm 0.060}$	$0.707_{\pm 0.076}$
(Gupta et al., 2022)	$0.403_{\pm 0.029}$	$0.445_{\pm 0.071}$	$0.188_{\pm 0.065}$	$0.410_{\pm 0.051}$	$0.480_{\pm 0.087}$	$0.505_{\pm 0.066}$	$0.470_{\pm 0.092}$
MatBERT	$0.705_{\pm 0.011}$	$0.796_{\pm 0.029}$	$0.691_{\pm 0.060}$	$0.805_{\pm 0.018}$	$0.756_{\pm 0.071}$	$0.778_{\pm 0.015}$	$0.798_{\pm 0.031}$
(Walker et al., 2021)	$0.469_{\pm 0.037}$	$0.558_{\pm 0.044}$	$0.400_{\pm 0.070}$	$0.574_{\pm 0.061}$	$0.524_{\pm 0.088}$	$0.547_{\pm 0.039}$	$0.569_{\pm 0.055}$
BatteryBERT	$0.690_{\pm 0.014}$	$0.673_{\pm 0.029}$	$0.439_{\pm 0.185}$	$0.733_{\pm 0.026}$	$0.607_{\pm 0.169}$	$0.743_{\pm 0.015}$	$0.722_{\pm 0.045}$
(Huang and Cole, 2022)	$0.464_{\pm 0.018}$	$0.407_{\pm 0.045}$	$0.168_{\pm 0.110}$	$0.483_{\pm 0.049}$	$0.369_{\pm 0.140}$	$0.497_{\pm 0.015}$	$0.470_{\pm 0.043}$
SciBERT	$0.686_{\pm 0.015}$	$0.754_{\pm 0.029}$	$0.598_{\pm 0.027}$	$0.708_{\pm 0.115}$	$0.724_{\pm 0.045}$	$0.754_{\pm 0.054}$	$0.734_{\pm 0.079}$
(Beltagy et al., 2019)	$0.464_{\pm 0.035}$	$0.493_{\pm 0.063}$	$0.298_{\pm 0.048}$	$0.465_{\pm 0.115}$	$0.471_{\pm 0.069}$	$0.509_{\pm 0.064}$	$0.497_{\pm 0.091}$
ScholarBERT	$0.206_{\pm 0.350}$	$0.179_{\pm 0.088}$	$0.109_{\pm 0.142}$	$0.134_{\pm 0.036}$	$0.263_{\pm 0.109}$	$0.168_{\pm 0.044}$	$0.168_{\pm 0.067}$
(Hong et al., 2022)	$0.069_{\pm 0.131}$	$0.108_{\pm 0.057}$	$0.018_{\pm 0.033}$	$0.071_{\pm 0.023}$	$0.122_{\pm 0.073}$	$0.098_{\pm 0.045}$	$0.101_{\pm 0.034}$
BioBERT	$0.665_{\pm 0.018}$	$0.708_{\pm 0.119}$	$0.204_{\pm 0.114}$	$0.723_{\pm 0.075}$	$0.455_{\pm 0.114}$	$0.725_{\pm 0.024}$	$0.715_{\pm 0.031}$
(Wada et al., 2020)	$0.403_{\pm 0.030}$	$0.431_{\pm 0.115}$	$0.019_{\pm 0.000}$	$0.474_{\pm 0.071}$	$0.188_{\pm 0.065}$	$0.452_{\pm 0.044}$	$0.459_{\pm 0.055}$
BERT	$0.606_{\pm 0.009}$	$0.636_{\pm 0.034}$	$0.235_{\pm 0.069}$	$0.670_{\pm 0.056}$	$0.455_{\pm 0.138}$	$0.664_{\pm 0.047}$	$0.657_{\pm 0.079}$
(Devlin et al., 2018)	$0.304_{\pm 0.024}$	$0.382_{\pm 0.041}$	$0.055_{\pm 0.040}$	$0.441_{\pm 0.060}$	$0.267_{\pm 0.089}$	$0.418_{\pm 0.046}$	$0.416_{\pm 0.058}$

Table 4: Results of **named entity recognition** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	$0.671_{\pm 0.083}$	$0.545_{\pm 0.102}$	$0.490_{\pm 0.139}$	$0.747_{\pm 0.128}$	$0.800_{\pm 0.058}$	$0.818_{\pm 0.137}$	$0.791_{\pm 0.046}$
(Gupta et al., 2022)	$0.439_{\pm 0.137}$	$0.219_{\pm 0.035}$	$0.218_{\pm 0.073}$	$0.461_{\pm 0.190}$	$0.482_{\pm 0.064}$	$0.530_{\pm 0.203}$	$0.507_{\pm 0.073}$
MatBERT	$0.714_{\pm 0.023}$	$0.644_{\pm 0.050}$	$0.591_{\pm 0.267}$	$0.871_{\pm 0.020}$	$0.804_{\pm 0.071}$	$0.848_{\pm 0.045}$	$0.875_{\pm 0.015}$
(Walker et al., 2021)	$0.487_{\pm 0.075}$	$0.310_{\pm 0.078}$	$0.297_{\pm 0.143}$	$0.623_{\pm 0.035}$	$0.513_{\pm 0.138}$	$0.569_{\pm 0.019}$	$0.630_{\pm 0.047}$
BatteryBERT	$0.594_{\pm 0.085}$	$0.592_{\pm 0.084}$	$0.423_{\pm 0.097}$	$0.823_{\pm 0.073}$	$0.801_{\pm 0.081}$	$0.854_{\pm 0.029}$	$0.786_{\pm 0.113}$
(Huang and Cole, 2022)	$0.359_{\pm 0.075}$	$0.297_{\pm 0.025}$	$0.167_{\pm 0.074}$	$0.553_{\pm 0.074}$	$0.466_{\pm 0.111}$	$0.592_{\pm 0.066}$	$0.472_{\pm 0.150}$
SciBERT	$0.699_{\pm 0.105}$	$0.585_{\pm 0.125}$	$0.643_{\pm 0.088}$	$0.799_{\pm 0.139}$	$0.783_{\pm 0.085}$	$0.814_{\pm 0.125}$	$0.819_{\pm 0.067}$
(Beltagy et al., 2019)	$0.495_{\pm 0.099}$	$0.267_{\pm 0.042}$	$0.311_{\pm 0.098}$	$0.527_{\pm 0.204}$	$0.474_{\pm 0.099}$	$0.528_{\pm 0.180}$	$0.545_{\pm 0.119}$
ScholarBERT	$0.603_{\pm 0.179}$	$0.619_{\pm 0.248}$	$0.243_{\pm 0.351}$	$0.416_{\pm 0.013}$	$0.543_{\pm 0.060}$	$0.367_{\pm 0.080}$	$0.428_{\pm 0.148}$
(Hong et al., 2022)	$0.178_{\pm 0.186}$	$0.384_{\pm 0.154}$	$0.078_{\pm 0.139}$	$0.334_{\pm 0.006}$	$0.252_{\pm 0.062}$	$0.236_{\pm 0.119}$	$0.274_{\pm 0.110}$
BioBERT	$0.692_{\pm 0.105}$	$0.538_{\pm 0.108}$	$0.306_{\pm 0.032}$	$0.743_{\pm 0.199}$	$0.674_{\pm 0.093}$	$0.666_{\pm 0.220}$	$0.797_{\pm 0.092}$
(Wada et al., 2020)	$0.458_{\pm 0.087}$	$0.243_{\pm 0.029}$	$0.079_{\pm 0.017}$	$0.442_{\pm 0.215}$	$0.323_{\pm 0.092}$	$0.324_{\pm 0.118}$	$0.465_{\pm 0.134}$
BERT	$0.564_{\pm 0.130}$	$0.626_{\pm 0.103}$	$0.368_{\pm 0.112}$	$0.792_{\pm 0.056}$	$0.696_{\pm 0.046}$	$0.636_{\pm 0.094}$	$0.782_{\pm 0.056}$
(Devlin et al., 2018)	$0.357_{\pm 0.076}$	$0.306_{\pm 0.075}$	$0.100_{\pm 0.018}$	$0.533_{\pm 0.041}$	$0.382_{\pm 0.039}$	$0.382_{\pm 0.043}$	$0.494_{\pm 0.061}$

Table 5: Results of **relation classification** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	$0.108_{\pm 0.062}$	$0.148_{\pm 0.182}$	$0.280_{\pm 0.127}$	$0.448_{\pm 0.091}$	$0.498_{\pm 0.045}$	$0.484_{\pm 0.015}$	$0.436_{\pm 0.066}$
(Gupta et al., 2022)	$0.041_{\pm 0.020}$	$0.050_{\pm 0.071}$	$0.122_{\pm 0.063}$	$0.251_{\pm 0.075}$	$0.310_{\pm 0.036}$	$0.292_{\pm 0.052}$	$0.251_{\pm 0.075}$
MatBERT	$0.152_{\pm 0.093}$	$0.160_{\pm 0.169}$	$0.341_{\pm 0.006}$	$0.453_{\pm 0.108}$	$0.483_{\pm 0.063}$	$0.515_{\pm 0.040}$	$0.451_{\pm 0.091}$
(Walker et al., 2021)	$0.029_{\pm 0.021}$	$0.033_{\pm 0.033}$	$0.174_{\pm 0.027}$	$0.274_{\pm 0.087}$	$0.298_{\pm 0.037}$	$0.288_{\pm 0.064}$	$0.288_{\pm 0.066}$
BatteryBERT	$0.149_{\pm 0.072}$	$0.162_{\pm 0.166}$	$0.232_{\pm 0.196}$	$0.397_{\pm 0.105}$	$0.438_{\pm 0.063}$	$0.443_{\pm 0.023}$	$0.457_{\pm 0.024}$
(Huang and Cole, 2022)	$0.030_{\pm 0.039}$	$0.036_{\pm 0.029}$	$0.104_{\pm 0.088}$	$0.233_{\pm 0.086}$	$0.298_{\pm 0.037}$	$0.250_{\pm 0.068}$	$0.277_{\pm 0.034}$
SciBERT	$0.152_{\pm 0.123}$	$0.160_{\pm 0.189}$	$0.312_{\pm 0.015}$	$0.449_{\pm 0.079}$	$0.442_{\pm 0.135}$	$0.484_{\pm 0.042}$	$0.451_{\pm 0.077}$
(Beltagy et al., 2019)	$0.041_{\pm 0.068}$	$0.033_{\pm 0.032}$	$0.159_{\pm 0.024}$	$0.259_{\pm 0.072}$	$0.264_{\pm 0.103}$	$0.287_{\pm 0.075}$	$0.276_{\pm 0.080}$
ScholarBERT	$0.349_{\pm 0.102}$	$0.444_{\pm 0.091}$	$0.262_{\pm 0.062}$	$0.454_{\pm 0.094}$	$0.454_{\pm 0.095}$	$0.431_{\pm 0.081}$	$0.489_{\pm 0.083}$
(Hong et al., 2022)	$0.250_{\pm 0.101}$	$0.253_{\pm 0.103}$	$0.102_{\pm 0.108}$	$0.312_{\pm 0.131}$	$0.264_{\pm 0.102}$	$0.296_{\pm 0.144}$	$0.356_{\pm 0.109}$
BioBERT	$0.119_{\pm 0.080}$	$0.160_{\pm 0.170}$	$0.054_{\pm 0.000}$	$0.489_{\pm 0.058}$	$0.491_{\pm 0.027}$	$0.473_{\pm 0.034}$	$0.488_{\pm 0.036}$
(Wada et al., 2020)	$0.030_{\pm 0.011}$	$0.034_{\pm 0.032}$	$0.013_{\pm 0.000}$	$0.305_{\pm 0.090}$	$0.295_{\pm 0.059}$	$0.268_{\pm 0.061}$	$0.274_{\pm 0.049}$
BERT	$0.198_{\pm 0.041}$	$0.160_{\pm 0.170}$	$0.232_{\pm 0.002}$	$0.400_{\pm 0.017}$	$0.414_{\pm 0.064}$	$0.451_{\pm 0.074}$	$0.418_{\pm 0.053}$
(Devlin et al., 2018)	$0.042_{\pm 0.055}$	$0.033_{\pm 0.033}$	$0.049_{\pm 0.008}$	$0.194_{\pm 0.025}$	$0.214_{\pm 0.092}$	$0.265_{\pm 0.104}$	$0.225_{\pm 0.091}$

Table 6: Results of **event argument extraction** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	$0.685_{\pm 0.074}$	$0.673_{\pm 0.003}$	$0.607_{\pm 0.277}$	$0.706_{\pm 0.013}$	$0.694_{\pm 0.041}$	$0.686_{\pm 0.158}$	$0.719_{\pm 0.116}$
(Gupta et al., 2022)	$0.588_{\pm 0.152}$	$0.402_{\pm 0.001}$	$0.386_{\pm 0.150}$	$0.633_{\pm 0.115}$	$0.524_{\pm 0.175}$	$0.583_{\pm 0.226}$	$0.623_{\pm 0.183}$
MatBERT	$0.753_{\pm 0.031}$	$0.671_{\pm 0.002}$	$0.673_{\pm 0.001}$	$0.727_{\pm 0.089}$	$0.776_{\pm 0.059}$	$0.649_{\pm 0.039}$	$0.756_{\pm 0.073}$
(Walker et al., 2021)	$0.730_{\pm 0.016}$	$0.402_{\pm 0.001}$	$0.404_{\pm 0.004}$	$0.601_{\pm 0.212}$	$0.722_{\pm 0.076}$	$0.509_{\pm 0.155}$	$0.691_{\pm 0.188}$
BatteryBERT	$0.663_{\pm 0.088}$	$0.672_{\pm 0.001}$	$0.672_{\pm 0.002}$	$0.621_{\pm 0.160}$	$0.626_{\pm 0.113}$	$0.672_{\pm 0.031}$	$0.633_{\pm 0.075}$
(Huang and Cole, 2022)	$0.585_{\pm 0.156}$	$0.402_{\pm 0.000}$	$0.402_{\pm 0.001}$	$0.564_{\pm 0.180}$	$0.574_{\pm 0.092}$	$0.540_{\pm 0.129}$	$0.610_{\pm 0.046}$
SciBERT	$0.690_{\pm 0.074}$	$0.673_{\pm 0.002}$	$0.568_{\pm 0.289}$	$0.703_{\pm 0.041}$	$0.711_{\pm 0.076}$	$0.662_{\pm 0.169}$	$0.696_{\pm 0.094}$
(Beltagy et al., 2019)	$0.605_{\pm 0.150}$	$0.402_{\pm 0.001}$	$0.370_{\pm 0.089}$	$0.598_{\pm 0.204}$	$0.598_{\pm 0.203}$	$0.562_{\pm 0.202}$	$0.546_{\pm 0.243}$
ScholarBERT	$0.620_{\pm 0.161}$	$0.603_{\pm 0.271}$	$0.658_{\pm 0.029}$	$0.672_{\pm 0.003}$	$0.662_{\pm 0.144}$	$0.668_{\pm 0.016}$	$0.663_{\pm 0.032}$
(Hong et al., 2022)	$0.386_{\pm 0.150}$	$0.371_{\pm 0.122}$	$0.407_{\pm 0.010}$	$0.482_{\pm 0.001}$	$0.534_{\pm 0.260}$	$0.405_{\pm 0.007}$	$0.433_{\pm 0.122}$
BioBERT	$0.629_{\pm 0.041}$	$0.672_{\pm 0.002}$	$0.671_{\pm 0.001}$	$0.658_{\pm 0.211}$	$0.709_{\pm 0.033}$	$0.680_{\pm 0.193}$	$0.675_{\pm 0.144}$
(Wada et al., 2020)	$0.507_{\pm 0.033}$	$0.402_{\pm 0.001}$	$0.401_{\pm 0.001}$	$0.588_{\pm 0.258}$	$0.651_{\pm 0.081}$	$0.622_{\pm 0.226}$	$0.578_{\pm 0.102}$
BERT	$0.709_{\pm 0.090}$	$0.672_{\pm 0.001}$	$0.672_{\pm 0.003}$	$0.685_{\pm 0.050}$	$0.727_{\pm 0.102}$	$0.629_{\pm 0.291}$	$0.665_{\pm 0.057}$
(Devlin et al., 2018)	$0.585_{\pm 0.093}$	$0.468_{\pm 0.283}$	$0.402_{\pm 0.001}$	$0.562_{\pm 0.221}$	$0.602_{\pm 0.238}$	$0.468_{\pm 0.283}$	$0.532_{\pm 0.194}$

Table 7: Results of **paragraph classification** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	0.383$_{\pm 0.024}$	0.334$_{\pm 0.004}$	0.424$_{\pm 0.249}$	0.676$_{\pm 0.071}$	0.631$_{\pm 0.081}$	0.741$_{\pm 0.157}$	0.692$_{\pm 0.179}$
(Gupta et al., 2022)	0.082$_{\pm 0.009}$	0.063$_{\pm 0.001}$	0.169$_{\pm 0.096}$	0.505$_{\pm 0.094}$	0.445$_{\pm 0.153}$	0.549$_{\pm 0.179}$	0.484$_{\pm 0.254}$
MatBERT	0.346$_{\pm 0.006}$	0.334$_{\pm 0.001}$	0.549$_{\pm 0.087}$	0.792$_{\pm 0.073}$	0.669$_{\pm 0.061}$	0.744$_{\pm 0.010}$	0.717$_{\pm 0.040}$
(Walker et al., 2021)	0.067$_{\pm 0.004}$	0.063$_{\pm 0.000}$	0.300$_{\pm 0.045}$	0.653$_{\pm 0.184}$	0.497$_{\pm 0.086}$	0.557$_{\pm 0.082}$	0.549$_{\pm 0.091}$
BatteryBERT	0.280$_{\pm 0.004}$	0.334$_{\pm 0.001}$	0.311$_{\pm 0.062}$	0.670$_{\pm 0.046}$	0.558$_{\pm 0.179}$	0.492$_{\pm 0.181}$	0.614$_{\pm 0.128}$
(Huang and Cole, 2022)	0.118$_{\pm 0.041}$	0.063$_{\pm 0.000}$	0.073$_{\pm 0.028}$	0.496$_{\pm 0.117}$	0.358$_{\pm 0.149}$	0.282$_{\pm 0.184}$	0.419$_{\pm 0.149}$
SciBERT	0.281$_{\pm 0.009}$	0.334$_{\pm 0.001}$	0.455$_{\pm 0.081}$	0.727$_{\pm 0.114}$	0.623$_{\pm 0.069}$	0.740$_{\pm 0.133}$	0.701$_{\pm 0.138}$
(Beltagy et al., 2019)	0.052$_{\pm 0.027}$	0.063$_{\pm 0.001}$	0.207$_{\pm 0.095}$	0.564$_{\pm 0.137}$	0.456$_{\pm 0.135}$	0.533$_{\pm 0.160}$	0.516$_{\pm 0.217}$
ScholarBERT	0.437$_{\pm 0.104}$	0.489$_{\pm 0.105}$	0.330$_{\pm 0.007}$	0.389$_{\pm 0.001}$	0.492$_{\pm 0.165}$	0.389$_{\pm 0.001}$	0.322$_{\pm 0.260}$
(Hong et al., 2022)	0.193$_{\pm 0.076}$	0.266$_{\pm 0.105}$	0.070$_{\pm 0.015}$	0.190$_{\pm 0.000}$	0.308$_{\pm 0.156}$	0.191$_{\pm 0.001}$	0.178$_{\pm 0.051}$
BioBERT	0.300$_{\pm 0.015}$	0.324$_{\pm 0.001}$	0.334$_{\pm 0.062}$	0.662$_{\pm 0.060}$	0.561$_{\pm 0.128}$	0.545$_{\pm 0.157}$	0.647$_{\pm 0.140}$
(Wada et al., 2020)	0.073$_{\pm 0.002}$	0.062$_{\pm 0.000}$	0.073$_{\pm 0.027}$	0.426$_{\pm 0.078}$	0.346$_{\pm 0.133}$	0.347$_{\pm 0.128}$	0.446$_{\pm 0.231}$
BERT	0.348$_{\pm 0.047}$	0.334$_{\pm 0.001}$	0.313$_{\pm 0.083}$	0.668$_{\pm 0.061}$	0.593$_{\pm 0.059}$	0.594$_{\pm 0.081}$	0.656$_{\pm 0.099}$
(Devlin et al., 2018)	0.091$_{\pm 0.020}$	0.063$_{\pm 0.000}$	0.073$_{\pm 0.037}$	0.495$_{\pm 0.058}$	0.424$_{\pm 0.086}$	0.371$_{\pm 0.103}$	0.515$_{\pm 0.067}$

Table 8: Results of **synthesis action retrieval** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	0.888$_{\pm 0.093}$	0.908$_{\pm 0.001}$	0.907$_{\pm 0.001}$	0.908$_{\pm 0.010}$	0.903$_{\pm 0.019}$	0.905$_{\pm 0.020}$	0.914$_{\pm 0.008}$
(Gupta et al., 2022)	0.602$_{\pm 0.151}$	0.476$_{\pm 0.001}$	0.493$_{\pm 0.069}$	0.601$_{\pm 0.159}$	0.573$_{\pm 0.135}$	0.616$_{\pm 0.150}$	0.660$_{\pm 0.079}$
MatBERT	0.908$_{\pm 0.011}$	0.908$_{\pm 0.001}$	0.907$_{\pm 0.000}$	0.906$_{\pm 0.016}$	0.910$_{\pm 0.012}$	0.903$_{\pm 0.018}$	0.909$_{\pm 0.009}$
(Walker et al., 2021)	0.441$_{\pm 0.038}$	0.476$_{\pm 0.001}$	0.476$_{\pm 0.000}$	0.645$_{\pm 0.025}$	0.561$_{\pm 0.135}$	0.600$_{\pm 0.089}$	0.614$_{\pm 0.134}$
BatteryBERT	0.908$_{\pm 0.012}$	0.907$_{\pm 0.000}$	0.908$_{\pm 0.000}$	0.895$_{\pm 0.050}$	0.890$_{\pm 0.036}$	0.907$_{\pm 0.002}$	0.912$_{\pm 0.015}$
(Huang and Cole, 2022)	0.452$_{\pm 0.045}$	0.475$_{\pm 0.001}$	0.476$_{\pm 0.000}$	0.679$_{\pm 0.080}$	0.685$_{\pm 0.074}$	0.519$_{\pm 0.144}$	0.684$_{\pm 0.095}$
SciBERT	0.896$_{\pm 0.080}$	0.907$_{\pm 0.000}$	0.825$_{\pm 0.218}$	0.908$_{\pm 0.009}$	0.902$_{\pm 0.017}$	0.902$_{\pm 0.020}$	0.911$_{\pm 0.017}$
(Beltagy et al., 2019)	0.421$_{\pm 0.159}$	0.469$_{\pm 0.004}$	0.535$_{\pm 0.079}$	0.586$_{\pm 0.166}$	0.596$_{\pm 0.161}$	0.623$_{\pm 0.130}$	0.617$_{\pm 0.143}$
ScholarBERT	0.805$_{\pm 0.020}$	0.839$_{\pm 0.268}$	0.908$_{\pm 0.001}$	0.908$_{\pm 0.000}$	0.900$_{\pm 0.019}$	0.907$_{\pm 0.001}$	0.906$_{\pm 0.007}$
(Hong et al., 2022)	0.458$_{\pm 0.099}$	0.477$_{\pm 0.004}$	0.485$_{\pm 0.000}$	0.476$_{\pm 0.000}$	0.509$_{\pm 0.093}$	0.476$_{\pm 0.001}$	0.478$_{\pm 0.008}$
BioBERT	0.908$_{\pm 0.001}$	0.907$_{\pm 0.001}$	0.907$_{\pm 0.001}$	0.910$_{\pm 0.012}$	0.899$_{\pm 0.047}$	0.908$_{\pm 0.015}$	0.915$_{\pm 0.021}$
(Wada et al., 2020)	0.476$_{\pm 0.001}$	0.478$_{\pm 0.001}$	0.503$_{\pm 0.005}$	0.614$_{\pm 0.175}$	0.610$_{\pm 0.078}$	0.638$_{\pm 0.089}$	0.686$_{\pm 0.098}$
BERT	0.911$_{\pm 0.010}$	0.907$_{\pm 0.000}$	0.907$_{\pm 0.001}$	0.906$_{\pm 0.007}$	0.905$_{\pm 0.010}$	0.892$_{\pm 0.035}$	0.910$_{\pm 0.016}$
(Devlin et al., 2018)	0.475$_{\pm 0.036}$	0.476$_{\pm 0.000}$	0.476$_{\pm 0.000}$	0.549$_{\pm 0.086}$	0.581$_{\pm 0.153}$	0.563$_{\pm 0.136}$	0.633$_{\pm 0.133}$

Table 9: Results of **sentence classification** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

NLP Model	Single Task	Single Task Prompt	MMOE	No Explanations	Potential Choices	Examples	Text2Schema
MatSciBERT	0.083$_{\pm 0.047}$	0.086$_{\pm 0.072}$	0.043$_{\pm 0.023}$	0.419$_{\pm 0.074}$	0.433$_{\pm 0.121}$	0.428$_{\pm 0.187}$	0.436$_{\pm 0.142}$
(Gupta et al., 2022)	0.087$_{\pm 0.045}$	0.010$_{\pm 0.011}$	0.016$_{\pm 0.005}$	0.182$_{\pm 0.043}$	0.169$_{\pm 0.069}$	0.169$_{\pm 0.075}$	0.194$_{\pm 0.062}$
MatBERT	0.179$_{\pm 0.074}$	0.151$_{\pm 0.121}$	0.148$_{\pm 0.148}$	0.547$_{\pm 0.050}$	0.493$_{\pm 0.078}$	0.502$_{\pm 0.034}$	0.548$_{\pm 0.058}$
(Walker et al., 2021)	0.087$_{\pm 0.030}$	0.024$_{\pm 0.022}$	0.057$_{\pm 0.067}$	0.276$_{\pm 0.047}$	0.230$_{\pm 0.067}$	0.221$_{\pm 0.011}$	0.273$_{\pm 0.051}$
BatteryBERT	0.093$_{\pm 0.074}$	0.073$_{\pm 0.033}$	0.032$_{\pm 0.031}$	0.540$_{\pm 0.092}$	0.433$_{\pm 0.155}$	0.506$_{\pm 0.065}$	0.520$_{\pm 0.057}$
(Huang and Cole, 2022)	0.009$_{\pm 0.012}$	0.008$_{\pm 0.011}$	0.008$_{\pm 0.009}$	0.270$_{\pm 0.108}$	0.211$_{\pm 0.056}$	0.236$_{\pm 0.072}$	0.262$_{\pm 0.073}$
SciBERT	0.098$_{\pm 0.054}$	0.099$_{\pm 0.075}$	0.125$_{\pm 0.073}$	0.469$_{\pm 0.112}$	0.432$_{\pm 0.106}$	0.446$_{\pm 0.167}$	0.481$_{\pm 0.144}$
(Beltagy et al., 2019)	0.020$_{\pm 0.021}$	0.013$_{\pm 0.018}$	0.047$_{\pm 0.016}$	0.207$_{\pm 0.066}$	0.183$_{\pm 0.061}$	0.179$_{\pm 0.071}$	0.224$_{\pm 0.010}$
ScholarBERT	0.286$_{\pm 0.042}$	0.289$_{\pm 0.044}$	0.063$_{\pm 0.007}$	0.323$_{\pm 0.058}$	0.276$_{\pm 0.080}$	0.338$_{\pm 0.053}$	0.296$_{\pm 0.085}$
(Hong et al., 2022)	0.110$_{\pm 0.009}$	0.111$_{\pm 0.019}$	0.005$_{\pm 0.004}$	0.111$_{\pm 0.027}$	0.076$_{\pm 0.024}$	0.117$_{\pm 0.015}$	0.109$_{\pm 0.044}$
BioBERT	0.096$_{\pm 0.171}$	0.094$_{\pm 0.118}$	0.042$_{\pm 0.024}$	0.517$_{\pm 0.031}$	0.319$_{\pm 0.059}$	0.424$_{\pm 0.145}$	0.452$_{\pm 0.114}$
(Wada et al., 2020)	0.023$_{\pm 0.020}$	0.015$_{\pm 0.024}$	0.004$_{\pm 0.001}$	0.241$_{\pm 0.082}$	0.110$_{\pm 0.048}$	0.177$_{\pm 0.119}$	0.191$_{\pm 0.045}$
BERT	0.086$_{\pm 0.032}$	0.082$_{\pm 0.065}$	0.034$_{\pm 0.026}$	0.566$_{\pm 0.042}$	0.421$_{\pm 0.137}$	0.476$_{\pm 0.079}$	0.520$_{\pm 0.019}$
(Devlin et al., 2018)	0.011$_{\pm 0.005}$	0.012$_{\pm 0.018}$	0.005$_{\pm 0.006}$	0.306$_{\pm 0.073}$	0.204$_{\pm 0.078}$	0.225$_{\pm 0.066}$	0.257$_{\pm 0.022}$

Table 10: Results of **slot filling** task among seven tasks on different schema settings for various BERT models pre-trained on different domain specific text data. For each model, the top line represents the micro-F1 score and the bottom line represents the macro-F1 score. We report the mean across 5 experiments with a confidence interval of two standard deviations. We highlight the best performing method.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Yes - in Section 7.

☑ A2. Did you discuss any potential risks of your work?
Yes - in Section 7.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Yes - in Section 1.

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

B ☑ Did you use or create scientific artifacts?

Yes - in Section 3.

☑ B1. Did you cite the creators of artifacts you used?
Yes - in Section 3.

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Yes - in Section 3.

☑ 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)?
Yes - in Section 3 and Section 7.

□ 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?
Not applicable. Left blank.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Yes - in Section 3.

☑ 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.
Yes - in Section 3.

C ☑ Did you run computational experiments?

Yes - in Section 5.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Yes - in Appendix Section A.

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?
Yes - in Appendix Section A.

☑ 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?
Yes - in Section 5.

□ 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.)?
Not applicable. Left blank.

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

Left blank.

□ 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.?
Not applicable. Left blank.

□ 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)?
Not applicable. Left blank.

□ 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?
Not applicable. Left blank.

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

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