Structured information extraction from complex scientific text with fine-tuned large language models

Alexander Dunn*[1,2], John Dagdelen*[1,2], Nicholas Walker[2], Sanghoon Lee[1,2], Andrew S. Rosen[2], Gerbrand Ceder[1,3], Kristin Persson[1,2,4] and Anubhav Jain[2]

[1] *Materials Science and Engineering Department, University of California, Berkeley, California, USA*
[2] *Energy Technologies Area, Lawrence Berkeley National Laboratory, California, USA*
[3] *Materials Sciences Division, Lawrence Berkeley National Laboratory, California, USA*
[4] *Molecular Foundry, Lawrence Berkeley National Laboratory, California, USA*

* Authors contributed equally

Abstract— Intelligently extracting and linking complex scientific information from unstructured text is a challenging endeavor particularly for those inexperienced with natural language processing. Here, we present a simple sequence-to-sequence approach to joint named entity recognition and relation extraction for complex hierarchical information in scientific text. The approach leverages a pre-trained large language model (LLM), GPT-3, that is fine-tuned on approximately 500 pairs of prompts (inputs) and completions (outputs). Information is extracted either from single sentences or across sentences in abstracts/passages, and the output can be returned as simple English sentences or a more structured format, such as a list of JSON objects. We demonstrate that LLMs trained in this way are capable of accurately extracting useful records of complex scientific knowledge for three representative tasks in materials chemistry: linking dopants with their host materials, cataloging metal-organic frameworks, and general chemistry/phase/morphology/application information extraction. This approach represents a simple, accessible, and highly-flexible route to obtaining large databases of structured knowledge extracted from unstructured text. An online demo is available at `http://www.matscholar.com/info-extraction`.

I. Introduction

The majority of scientific knowledge about solid-state materials is scattered across the text, tables, and figures of millions of academic research papers. Thus, it is difficult for researchers to fully leverage existing knowledge when designing experiments or to even properly understand the full body of past work. While machine learning models for direct property prediction have been increasingly employed as screening steps for materials discovery and design workflows [1, 2], this approach is limited by the amount of training data available in tabulated databases. Such limitations are particularly apparent for experimental properties and parameters (in contrast to databases of materials property data derived from *ab initio* simulations). Natural language processing (NLP) algorithms for materials, and named entity recognition (NER) models in particular, have made significant advances over the past half-decade towards structuring the existing body of textual materials science knowledge.[3, 4, 5, 6] In materials NER models, entity labels such as "material" or "property" are applied to words in text and can be used, sometimes with additional post-processing, to construct auto-generated tabular databases of materials property data aggregated from text entries.[7, 8, 9, 10, 11]

Yet, a key outstanding challenge in materials NLP is the development of relation extraction (RE) techniques to extract structured information that accurately describes the links *between* these entities. In this paper, we describe a sequence-to-sequence approach to document-level joint named entity recognition and relation extraction (NERRE) for the extraction of complex information from scientific text (Figure 1). The approach leverages a pre-trained large language model, GPT-3 [12], that is fine-tuned on approximately 500 document-completion examples to return structured records (*e.g.,* lists of JSON documents) containing desired information. Besides its high level of accuracy, the advantages of this approach are its flexibility and accessibility. Nearly any information extraction task is accommodated by specifying

Contact data: Anubhav Jain, ajain@lbl.gov

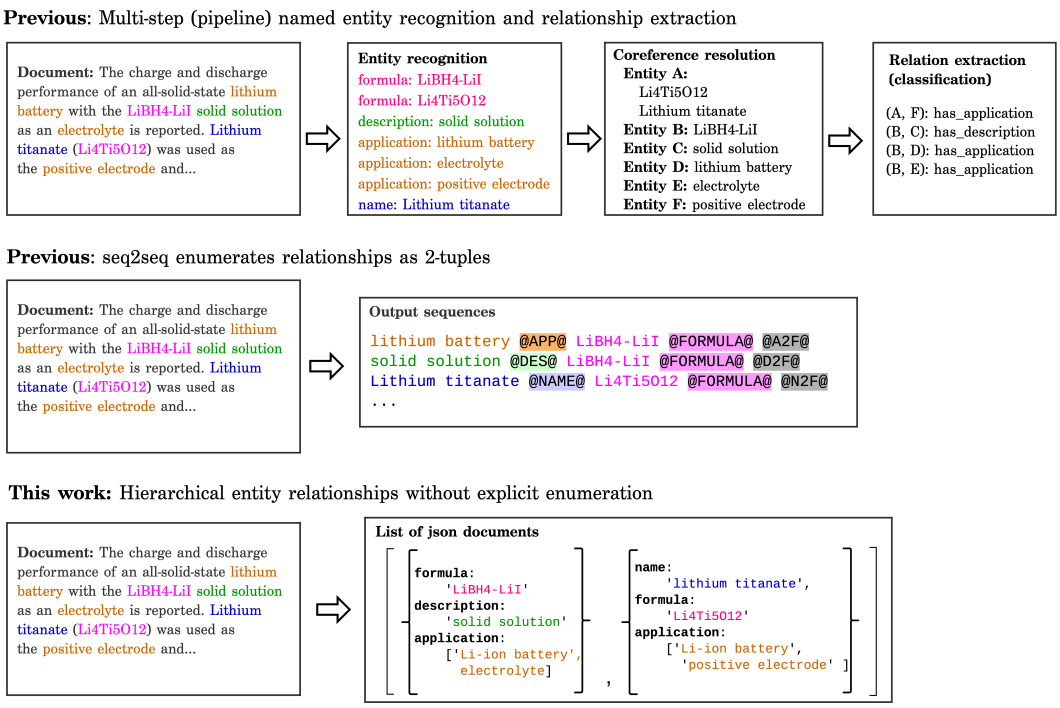

Fig. 1: Simplified comparison of previous relation extraction (RE) models and the *seq2seq*-LLM approach proposed in this work.

a new output schema, and scientific domain experts can easily construct their own models by simply reading passages and transcribing the type of output they want the model to produce. Furthermore, publicly available language model APIs can be used in place of custom, local language models, and limited machine learning expertise is required to achieve state of the art results on a variety of complex information extraction tasks.

Prior work

Prior information extraction studies in the domain of solid-state materials include NER of chemical synthesis parameters in methods section texts [13, 14, 15, 16], quantitative results of battery cycling experiments [17], or peak absorption wavelengths for UV-Vis experiments [18], among others [3, 4, 8, 9, 10, 11, 19]. Regular expressions, BiLSTM recurrent neural networks, and smaller transformer-based language models such as BERT are sufficient for such tasks. In these studies, entities (e.g., $LiCoO_2$, `"350K"`) rather than relations (e.g., `"350K"` is an experimental synthesis parameter for $LiCoO_2$) are the primary target of extraction.

Downstream tasks such as supervised machine learning or the construction of knowledge graphs require knowledge of the relationships between entities in the text. Relation extraction (RE) models are used to determine which entities are linked by a predefined set of relations. For example, in the sentence "LiCoO2 is studied as a Li-ion battery material", the material entity "LiCoO2" is linked to the application entity "Li-ion battery". There has been relatively little work on relation extraction in materials science text, but there has been much research interest in RE on general-purpose text, especially related to linking people, organizations, locations, and dates.[20, 21] These methods have traditionally relied on pipeline-based approaches where entities are first identified by an NER model, which is followed by one or more additional steps and a final relation classification model (see Figure 1, top row). State-of-the-art transformer-based implementations of pipeline implementations have been shown to perform document level relation extraction well on a variety of general-knowledge corpora[22] and more specialized domains such as chemical-disease relations[23] and gene-disease relations.[24]

However, scientific information often cannot be modeled as simple pairwise relations between entities. This is particularly apparent in inorganic materials science, where a compound's properties are determined by a complex combination of its elemental composition, atomic geometry, microstructure, morphology (*e.g.,* nanoparticles, heterostructures, and interfaces), processing history, and environmental factors such as temperature and pressure. Furthermore, inorganic materials knowledge is often inherently hierarchical such that the relations may only be valid between one entity type and a compound entity (itself comprised of several entities and relationships). For example, we may consider a zinc oxide-based material ("ZnO" linked to a morphology "nanoparticles") to be a catalyst, but "ZnO" and "nanoparticles" alone are not necessarily catalysts in themselves. In

theory, these types of relations can be modeled as n-tuples where n is the number of entities, but comprehensively enumerating all of the possible variations is both impractical and not amenable to conventional relation extraction methods because a sufficient number of training examples is required for each relation type. Moreover, a piece of materials science knowledge may be implied for multiple simultaneous values for the same entity type. For example, a sample of an "epitaxial" "La-doped" "thin film" of $HfZrO_4$ will have different physical properties than a "La-doped" "thin film" of $HfZrO_4$ and a "La-doped" sample of $HfZrO_4$. Current RE models are not designed to practically extract or preserve such kinds of highly complex, intricately related, and hierarchical relationships between arbitrary numbers of named entities; a more flexible strategy is required.

Sequence-to-sequence (seq2seq) relation extraction

Large language models (LLMs) such as GPT-3 [12], PaLM [25], Megatron [26], OPT [27], Gopher [28], and FLAN [29] have been shown to have remarkable ability to leverage semantic information between tokens in natural language sequences of varying length. They are particularly adept at sequence-to-sequence (*seq2seq*) tasks, where a text input is used to seed a text response from the model. In this paper we will refer to these inputs as "prompts" and the outputs as "completions." Use cases for seq2seq are broad[30] and include machine translation [31], answering general factual knowledge questions[32, 28], performing simple arithmetic [28], translating between languages [31, 33], summarizing text [34, 25], and chatbot applications [12, 35]. It stands to reason that these models may also be also adept at complex scientific information extraction.

Recently, end-to-end methods that use a single machine learning model have been investigated for joint named entity recognition and relation extraction (NERRE) for simple named entity recognition and pairwise relation extraction.[36, 37, 38] These methods take a sequence-to-sequence approach where a model is trained to output tuples of two or more named entities and the relation label belonging to the predefined set of possible relations between them (Figure 1, middle row). These methods have, so far, underperformed compared to the pipeline-based approaches such as Eider[39]. Fundamentally, these methods remain n-ary relation extraction systems and are subject to the same limitations.

In the domain of materials science, Huang & Cole recently fine-tuned a BERT model on battery publications and trained a model to enhance a database of NLP-extracted battery data [10]. Their approach employed a "question and answer" (Q/A) approach that extracted limited device-level information (*e.g.*, "What is the cathode?", "What is the anode?", "What is the electrolyte?") in tandem with conventional information extraction methods.[10] We note that this approach cannot be used on passages that contain information about more than one device, and it required the BERT language model to be trained on hundreds of thousands of battery research papers before being fine-tuned on the Q/A task.

In this work, we investigate a simple and facile approach to complex information extraction where a large language model is fine-tuned for simultaneous document-level named entity recognition and relation extraction. The method is simple yet able to flexibly handle complex inter-relations (including cases where information exists in a hierarchy or as lists of multiple items) without requiring enumeration of all of possible n-tuple relations or preliminary NER. We fine-tune a large language model, GPT-3, to accept a text passage (for example, a research paper abstract) and write a precisely formatted "summary" of knowledge contained in the prompt. This completion can be formatted as either English sentences or a more structured schema such as a list of JSON documents. To use this method, one only has to define the desired output structure—for example, a list of JSON objects with a predefined set of keys—and annotate $\sim 100-500$ text passages using this format. GPT-3 is then fine-tuned on these examples, and the resulting model is able to accurately extract desired information from text and output information in the same structured representation as shown in Figure 1.

This method shows strong performance on both sentence-level and document-level materials information extraction. Moreover, the method requires little knowledge of how LLMs work internally; the LLM may be simply treated by the user as a black-box that creates precisely-formatted summaries of scientific text. Therefore, researchers may use this method with little NLP experience. We also discuss how intermediate models can be used to pre-suggest entities for annotation, vastly increasing the speed and ease of annotating documents so that large training sets can be constructed relatively quickly. Although the example tasks shown are from materials science, the generality and accessibility of the method implies it may be readily applied to other domains such as physics or biology. In particular, this approach does not appear to require fine-tuning on a large corpus of domain-specific data (e.g., millions of article abstracts or paragraphs) as in previous methods; rather, the comprehensive pretraining of the LLMs along with the user-provided annotations are sufficient to accomplish a broad array of complex tasks.

II. METHODS

General seq2seq NERRE

We fine-tune GPT-3 to perform NERRE tasks using $100-1{,}000$ manually annotated text-extraction (prompt-completion) pairs. Extractions contain the desired information formatted with a predefined, consistent schema across all training examples. These schemas can range in complexity

from English sentences with predefined sentence structures to lists of JSON objects or nested JSON objects. In principle, many other potential schemas (*e.g.,* YAML, psuedocode) may also be valid, though we do not explore those here. Once fine tuned on sufficient data adhering to the schema, a model will be capable of performing the same information extraction task on new text data with high accuracy. The model outputs completions in the same schema as the training examples. We refer to this approach generally as "LLM-NERRE".

Our general workflow for training GPT-3 to perform NERRE tasks is outlined in Fig. 2. Annotations are initially performed by human domain experts to create an initial training set, and then a partially trained model is used to accelerate the collection of additional training examples. Fine-tuning is then performed on these examples to produce a "partially trained" model, which is used to pre-fill annotations that are subsequently corrected by the human annotator before being added to the training set. As discussed later in Section IV, we have found that this in-the-loop annotation procedure greatly accelerates annotation, reducing the average time per annotation of materials science abstracts from 100 seconds per abstract to around 40 seconds per abstract. Once a sufficient number of annotations have been completed, the final fine-tuned model is capable of extracting information in the desired format without human correction. Optionally, as illustrated in Figs. 3-5, the structured outputs may be further decoded and post-processed into hierarchical knowledge graphs.

Benchmark tasks and output schema design

We use the described approach on three materials information extraction tasks: solid-state impurity doping, metal–organic frameworks, and general materials information extraction. Details for each are summarized in Table 1.

Solid-state impurity doping schema

The two core entities of interest for the solid state impurity doping task are host (`host`) and dopant (`dopant`). Hosts are defined as the host crystal, sample, or material class along with crucial descriptors in its immediate context (*e.g.,* "ZnO2 nanoparticles", "LiNbO3", "half-Heuslers"). Dopants are taken to be any elements or ions that are minority species, intentionally added impurities, or specific point defects or charge carriers ("hole-doped", "S vacancies"). One host may be doped with more than one dopant (*e.g.,* separate single-doping or codoping), or the same dopant may be linked to more than one host material. There may also be many independent pairs of dopant-host relations, often within a single sentence, or many unrelated dopants and hosts (no relations). We impose no restriction on the number or structure of the dopant-host relations beyond that each relation connects a host to a dopant. We also seek to extract two additional en-

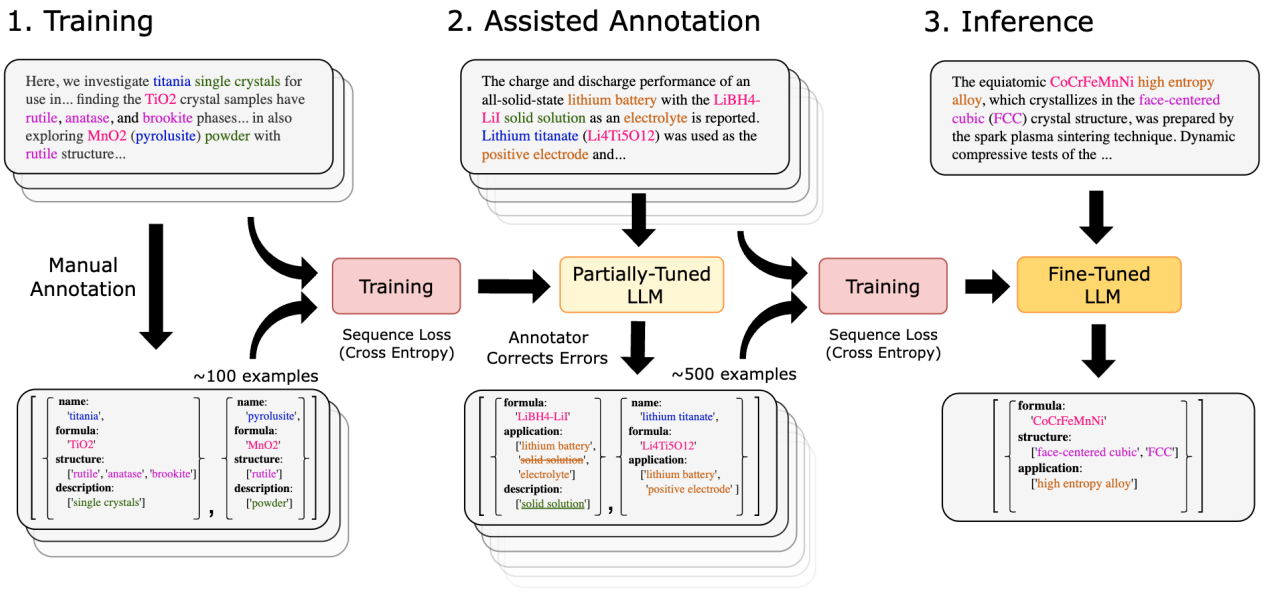

Fig. 2: Overview of our sequence-to-sequence approach to document-level joint named entity recognition and relationship extraction task. In the first step, lists of JSON documents are prepared from abstracts according to a predefined schema, and a GPT-3 model is trained. In the second step, this preliminary (intermediate) model is used to accelerate the preparation of additional training data by pre-annotation with the partially trained model and manual correction. This step may be repeated multiple times with each subsequent partial fine-tuning improving in performance. In the final step, GPT-3 is fine-tuned on the complete dataset and used for inference to extract desired information from new text.

Table 1: Parameters for the models trained on the three materials information extraction tasks.

Task	Model Name	Training Samples	Task level	Completion Format
Doping	`Doping-JSON`	413 sentences	Sentence	JSON
	`Doping-ENG`			English sentences
	`DopingExtra-ENG`			
MOFs	`MOF-JSON`	507 abstracts	Abstract	JSON
General Materials	`General-JSON`	634 abstracts		

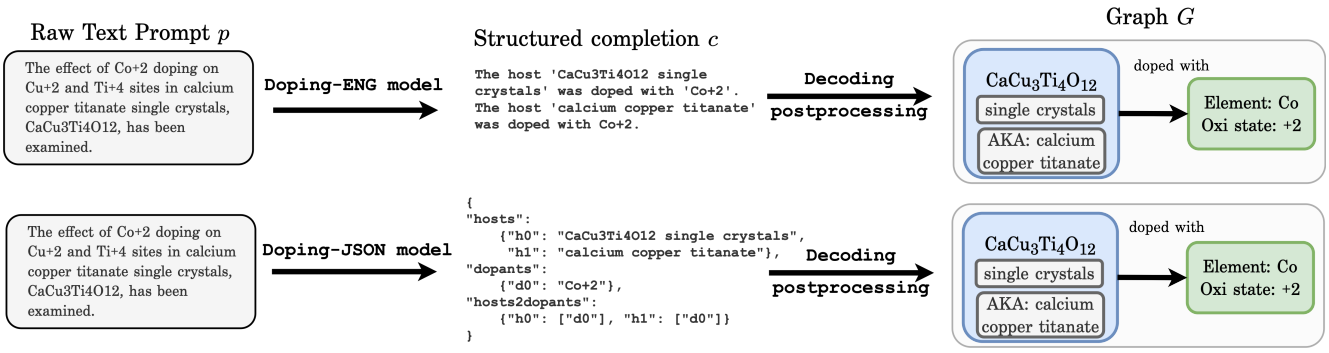

Fig. 3: Annotation schema example for the doping extraction task. A raw sentence text sequence prompt p is passed into an LLM-NERRE doping model which produces a structured sequence completion c. The structured completion format depends on the model; here, we train two separate models `Doping-ENG` and `Doping-JSON` with structured completion formats of English sentences and JSON, respectively. These models are completely independent for training and evaluation purposes. As an optional final step, the completions may be decoded and post-processed from string literals into hierarchical structured graph objects (G) for further analysis. Note this final step is separate from the NERRE models themselves, as the graph objects are decoded programmatically to a variety of formats (*e.g.,* JSON, NetworkX objects [40]); a more complex hierarchical graph example for the doping task is shown in Supplementary Figure S1.

tities: "modifiers" and "result", without explicit linking (*i.e.,* NER only). The `results` entity represents formulae with algebra in the stoichiometric coefficients (*e.g.,* $Al_xGa_{1-x}As$). These are formulae typically associated with crystalline solid solutions (*e.g.,* $CaCu_{3-x}Co_xTi_4O_{12}$) and their stoichiometric ranges. We also include stoichiometries where the algebra is substituted (*i.e., x* value specified) and the doped result is a specific composition (*e.g.,* $CaCu_{2.99}Co_{0.1}Ti_4O_{12}$). The `modifiers` is a loosely-bounded entity encapsulating other descriptors of the dopant-host relationship not captured by `dopant`, `host`, or `result`. We extract polarities (*e.g.,* "n-type", "n-SnSe"), dopant quantities (*e.g.,* "5 at.%", "$x < 0.3$"), defect types (*e.g.,* "substitutional", "antisite", "vacancy") and other modifiers of the `host` to `dopant` relationship (*e.g.,* "high-doping", "degenerately doped"). These entities (`host`, `dopant`, `result`, and `modifiers`) were chosen to define a minimal effective schema for extracting basic doping information.

All doping-related models are trained on sentence-level prompt/completion data and produce sentence-level completions. The main motivation for this design choice is that the vast majority of dopant-related data can be found within single sentences, and the remaining relational data is often difficult to resolve consistently for both annotators and models. We expand on problems with annotations and ambiguity in the Supplementary Information.

We train and evaluate three separate models for the doping task. The first two, `Doping-JSON` and `Doping-ENG`, both aim to extract only `dopant` and `host` entities and the relations between `dopant` and `host`. Modifier and result entities are ignored. These models differ only in the formats of their structured completions (schemas) as shown in Fig. 3. The `Doping-JSON` model uses is a JSON schema representing the graph of relationships between hosts and dopants within a single sentence, where unique keys identify dopant and host strings. The model aims to learn this relatively loose schema during fine-tuning. A separate key, "hosts2dopants", describes the pairwise relations according to those unique keys. The `Doping-ENG` model encodes the entity relationships as quasi-natural language summaries. These summaries represent the same information as the JSON schema, but they more closely mimic the natural language pre-training distribution of GPT-3. This schema is filled programmatically according to the schema shown in the Supplementary section on the doping schema, where each doping entry present is newline-separated. For example, the first line details a one-host-to-many dopants relationship, the second line expresses a dopant entity with no host, and the third line expresses a host entity with no dopant.

Finally, we independently train and evaluate a model to retrieve `dopant` and `host` entities and relationships as well as `modifiers` and `results` entities. We label this model

DopingExtra-ENG, as it uses the same natural-language-like schema of Doping-ENG but additionally extracts doping modifier and doping result data. Training data for DopingExtra-ENG is similar to that of the other doping models, but instead of ignoring results and modifier data, these additional entities are incorporated as additional newline-separated quasi-natural language statements. The schema for DopingExtra-ENG is shown in the Supplementary section on the doping schema.

General materials information schema

In our previous work [3, 4], we focused on NER for a specific set of entity types that are particularly relevant in materials science: materials, applications, structure/phase labels, synthesis methods, *etc.* However, we were not able to link these labeled entities together to record their relations beyond a simple "bag-of-entities" approach. In this work, we train an LLM to perform a "general materials information extraction" task that captures both entities and the complex network of interactions between them.

The schema we have designed for this task encapsulates an important subset of information about solid compounds and their applications. Each entry in the list, a self-contained JSON document, corresponds one-to-one with a material mentioned in the text. Materials entries are ordered by appearance in the text. The root of each entry starts with a compound's name and/or its chemical formula. If a name or formula is not mentioned for a material, no information about that material is extracted from the text. We also extract acronyms mentioned for a material's name/formula, although in cases where only an acronym is mentioned we do not create a material entry for the compound. Compounds that are not solids (ions, liquids, solvents, solutions, etc) are generally not extracted. The `name`, `formula`, and `acronym` fields are exclusively given string value in the JSON document for each material whereas the `description`, `structure_or_phase`, and `applications` fields are lists of an arbitrary number of strings. We label this model `General-JSON`, and an example is shown in Fig. 4.

Descriptions are defined as details about a compound's processing history, defects, modifications, or the sample's morphology. For example, consider the hypothetical text "Pt supported on CeO2 nanoparticles infused with Nb...". In this case, the "descriptions" entry for the material document referring to "Pt" might be annotated as `"['supported on CeO2']"`, and the entry in the document referring to "CeO2" would be `"['nanoparticles', 'Nb-doped']"`.

Structures/phases are defined as entities that directly imply the crystalline structure or symmetry of the compound. Crystal systems such as "cubic" or "tetragonal", structure names such as "rutile" or "NASICON", and space groups such as "Fd3m" or "space group No. 93" are all extracted in this field. We also include any information about crystal unit cells, such as lattice parameters and the angles between lattice vectors. "Amorphous" is also a valid structure/phase label.

Applications are defined as high-level use cases or major property classes for the material. Applications may represent different levels of device-level implementation. For example, a battery cathode material may have `"['Li-ion battery', 'cathode']"` as its applications entry. Generally, applications are mentioned in the order they are presented in the text, except for certain cases such as battery materials, in which case the type of device is generally mentioned before the electrode type, and catalysts, where the reaction catalyzed is generally listed following the "catalyst" entity in the list (*e.g.,* `"['catalyst', 'hydrogenation of citral']"`). More details and caveats of this schema are given in the Supplementary Information.

Metal–organic framework (MOF) schema

The schema used for the MOF cataloging task is based on the general materials information schema described in the previous section, which was modified to better suit the needs of MOF researchers. We developed this schema to extract MOF names (`name`), an entity for which there is no widely accepted standard[41], and chemical formulae (`mof_formula`), which form the root of the document. If no name or formula is present, no information is extracted for that instance. In addition, because there is a great deal of interest in using MOFs for ion and gas separation[42, 43], we extract "guest species", which are chemical species that have been incorporated, stored, or absorbed in the MOF. We extract applications the MOF is being studied for as a list of strings (e.g. `"['gas-separation']"` or `"[heterogeneous catalyst', 'Diels-Alder reactions']"`) as well as relevant descriptors for the MOF/sample, such as its morphology or processing history, similar to the general information extraction schema. Entries in the list are generally added in the order the material names/formulas appear in the text (described in more detail in Supplementary Information). The MOF extraction model is labeled `MOF-JSON`, and an example is shown in Fig. 5.

Comparison benchmarks and evaluation

To compare our model with other sequence-to-sequence approaches to information extraction, we perform a benchmark of two methods on the doping task to compare to the LLM-NERRE models. The first employs the seq2rel method of Giorgi *et al.*[36] for the host-dopant task. We formatted host-dopant relationships under tags labeled `@DOPANT@` and `@BASEMAT@` (base/host material), with their relationship signified by `@DBR@` ("dopant-base material relationship"); these sequences were constructed from the same training data as the `Doping-JSON` and `Doping-ENG` models. We trained

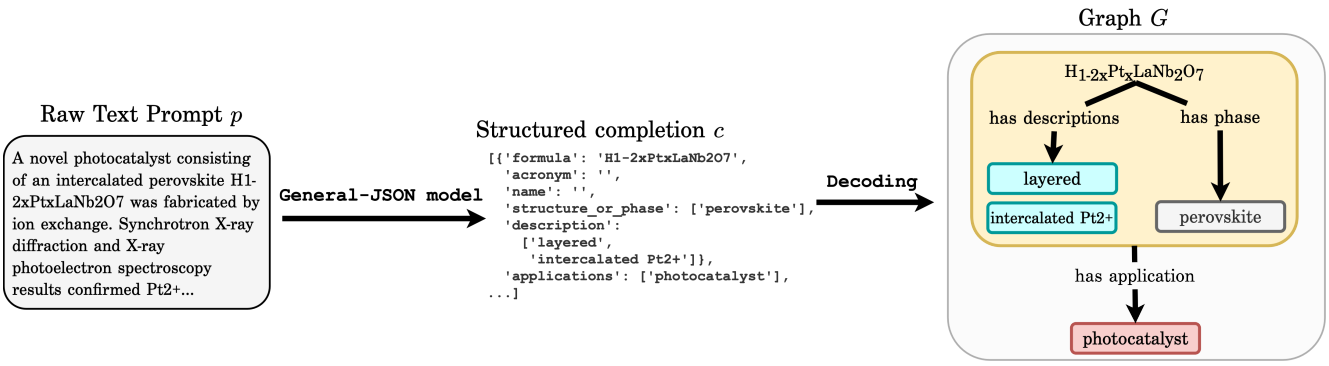

Fig. 4: Annotation schema example for the general materials-chemistry extraction task. A raw full-abstract text prompt p is passed to the `General-JSON` model which produces a structured completion c in JSON schema. The JSON schema is a list of individual material entries ordered by appearance in the text, each of which may have a name, formula, acronym, descriptors, applications, and/or phase label. The structured completion may then be programmatically decoded to a hierarchical materials graph G without a ML model.

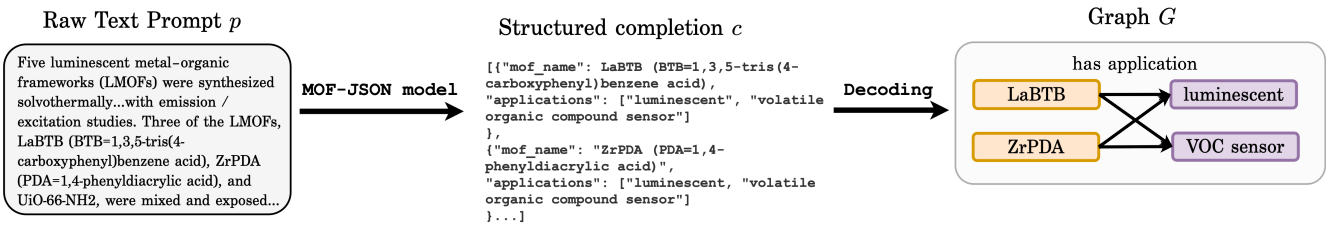

Fig. 5: Annotation schema example for the metal–organic frameworks extraction task. A raw full-abstract text prompt p is fed into the `MOF-JSON` model which produces a structured output sequence c similar to that of the `General-JSON` model shown in Fig. 4. The output sequence is a formatted string literal which can be directly loaded as JSON. The string may then be optionally decoded to a graph G for further analysis. In this example, only the MOF name and application were extracted from the passage, and both MOFs (LaBTB and ZrPDA) are linked to both applications (luminescent and VOC sensor).

seq2rel to perform sentence-level extraction with 30 epochs, batch size of 4, encoder learning rate $2 \cdot 10^{-5}$, decoder learning rate $5 \cdot 10^{-4}$, and pretrained BiomedNLP BERT tokenizer[44] (further training details are in Supporting Information). Additionally, we compare against the previously published MatBERT doping-NER model [4] combined with proximity-based heuristics for linking. With this method, a MatBERT NER model pretrained on $\sim$ 50 million materials science paragraphs and fine-tuned on 455 separate manually annotated abstracts first extracts hosts and dopants and then links them if they co-occur in the same sentence.

Datasets

Datasets were prepared from our database of more than 5 million research paper abstracts.[45] Annotations were performed by human annotators using a graphical user interface built using Jupyter[46], although in principle annotations could be conducted via a simple text editor. To accelerate the collection of training data, new annotations are collected via a "human in the loop" approach where models are trained on small datasets and their outputs are used as starting points and corrected by human annotators (see Figure 2.) This process of training and annotation is completed multiple times until a sufficiently large set of training data was achieved. Single domain experts separately annotated each of the training and test sets.

Doping dataset

Training and evaluation data was gathered from our database of research paper abstracts using the keywords "n-type", "p-type", "-dop", "-codop", "doped", "doping", and "dopant" (with exclusions for common irrelevant keywords such as "-dopamine"), resulting in $\sim$ 375k total abstracts. All doping tasks were trained on text from 162 randomly selected abstracts, comprising 1,215 total sentences and filtered with regular expressions to only include 413 relevant (potentially including doping information) sentences. Doping tasks were tested on an additional 232 sentences (77 relevant by regex) from a separate holdout test set of 31 abstracts.

General materials dataset

Training and evaluation data was gathered from our abstract database using keywords for a variety of materials properties and applications (*e.g.,* "magnetic", "laser", "space group", "ceramic", "fuel cell", "electrolytic"). A domain specialist used *ad hoc* class balancing to select keywords resulting in

$\sim 10-50$ abstracts per-keyword; all abstracts were manually screened for relevance. More details on this data collection are provided in Supplementary Information. The resulting ≈650 entries were manually annotated with the general materials information schema. Results were evaluated using a 10% random sample for validation, and this procedure was averaged over five trials with no hyperparameter tuning.

Metal–organic framework dataset

Training and evaluation data was selected from our database using the keywords "MOF", "MOFs", "metal-organic framework", "metal organic framework", "ZIF", "ZIFs", "porous coordination polymer", and "framework material", which produced approximately 6,000 results with a very high likelihood of containing MOF-related information. From these, 507 abstracts were randomly selected and annotated by a human MOF expert. Results were evaluated using the same procedure as the general materials dataset in the previous section.

GPT-3 fine tuning details

For all tasks, we fine-tune GPT-3 ('davinci', 175B parameters)[12] using the OpenAI API, which optimizes the cross-entropy loss on predicted tokens. Doping models were trained for 7 epochs at a batch size of 1, with inference temperature of 0 and output limited to a maximum length of 256 tokens (all doping models) or 512 tokens (`General-JSON`, `MOF-JSON`). The intermediate models shown in Figure 7 were trained with a number of epochs depending on the number of training samples t: 2 epochs for $2^0 \leq t < 2^6$, 4 epochs for $2^6 \leq t \leq 2^7$, and 7 epochs for $t = 2^8$. Models for the MOF and general materials extraction tasks were trained for 4 epochs with a batch size of 1. We use a learning rate multiplier of 0.1 and a prompt loss weight of 0.01 but have not performed hyperparameter tuning for these hyperparameters. For all tasks, the start and end tokens used were `"\n\n\n###\n\n\n"` and `"\n\n\nEND\n\n\n`.

Evaluation criteria

The fuzzy and complex nature of the entities and relationships detailed in the previous section necessitates the use of several metrics for scoring. We evaluate the performance of all models on two levels: sequence reconstruction and information extraction performance.

Sequence reconstruction

We measure sequence reconstruction as the ability of the model to output a sequence which is parsable (*i.e.,* output adheres to the schema format) and is similar to the true output sequence. In this evaluation, we do not decode entries or inspect individual entities or relationships; we merely measure the ability of the model to output a string similar to the true string.

We report three metrics: exact sequence match accuracy, Jaro-Winkler similarity [47], and parsability. Each of these metrics, averaged over all samples to evaluate for each task, is defined on $[0,1]$ where 0 indicates no sequence reconstruction and 1 indicates perfect reconstruction. Details and formalism on each metric are given in the Supplementary Info.

Information extraction performance

We measure information extraction performance as the ability of the model to jointly recognize entities and the relationships between them. We report two methods of scoring entity-relation data:

1. A triplet F_1 (defined later) computed on a stringent exact word-match basis (*i.e.,* how many words are correctly linked together exactly as they appear in the source text prompt).

2. A less-stringent triplet F_1 based on manual inspection from a domain expert (*i.e.,* how many sets of words are approximately correct in the context).

Exact word-match basis scoring. We score named entity relationships on a word-basis by first converting an entity E into a set of constituent k whitespace-separated words $E = \{w_1, w_2, w_3, ..., w_k\}$. Comparing two entities E^{true} and E^{test} for NER only, we count the number of exactly matching words in both sets as true positives ($E^{\text{true}} \cap E^{\text{test}}$) and the mathematical differences between the sets as false positives ($E^{\text{test}} - E^{\text{true}}$) or false negatives ($E^{\text{true}} - E^{\text{test}}$). For example, if the true entity is "Bi2Te3 thin film" and the predicted entity is "Bi2Te3 film sample", we record two true positive word exact matches ("Bi2Te3", "film"), one false negative ("thin"), and one false positive ("sample"). The lone exception to this method is for formula-type entities crucial for identifying materials, in which case E^{test} must contain *all* w_i parsable as stoichiometries for *any* of $w_i \in E^{\text{test}}$ to be considered correct. For example, if the true entity is "Bi2Te3 thin film", and the predicted entity is "thin film", we record three false negatives. Thus, any formula-type entity (Doping `host`, Doping `dopant`, General `formula`, and MOF `mof_formula`) containing a chemical composition is entirely incorrect if the composition is not an exact match. This choice of evaluation was made to avoid metrics artificially inflating the performance of the model. For an example, if we evaluate a true "Bi2Te3 nanoparticles" to a predicted "Bi2Se3 nanoparticles", we obtain very high similarities via Jaro-Winkler (0.977) and character-level BLEU-4 (0.858). In reality, this example prediction should be recorded as entirely incorrect – the material's chemistry is wrong.

We now score relationships between entities on a word basis using on the number of correct triplets. Triplets are

3-tuples relating word w_j^n of an entity E_n to word w_k^m of an entity E_m by relationship r, represented as (w_j^n, w_k^m, r). The total set of correct relationships T^{true} for a text contains many of these triplets. A test set of relationships T^{test} is evaluated by computing the number of triplets found in both sets ($T^{\text{true}} \cap T^{\text{test}}$) as true positives and the differences between these sets as false positives ($T^{\text{test}} - T^{\text{true}}$) or false negatives ($T^{\text{true}} - T^{\text{test}}$). Entity triplets are also bound to the same requirement for composition correctness if either of the words in the triplet belong to an formula-type entity (`host`, `dopant`, `formula`, `mof_formula`), *i.e.,* we count all triplets for two entities as incorrect if the formula is not an exact string match. With correct and incorrect triplets identified, F_1 scores for each relation are calculated as:

$$\text{precision} = \frac{\text{No. of correct relations retrieved}}{\text{No. of relations retrieved}} \tag{1}$$

$$\text{recall} = \frac{\text{No. of correct relations retrieved}}{\text{No. of relations in test set}} \tag{2}$$

$$F_1 = \frac{2(\text{precision} \cdot \text{recall})}{\text{precision} + \text{recall}} \tag{3}$$

To compute triplet scores across entire test sets in practice, we first select a subset of relations to evaluate. We note that this is not a full evaluation of the task we are training the model to perform, which involves linking many interrelated entities *simultaneously*, but is rather provided to help give a general sense of its performance compared to previous NERRE methods. For the doping task, we evaluate `host-dopant` relationships. For the general materials and MOF tasks, we evaluate relationships between the formula field (`formula` for general materials, `mof_formula` for MOFs) and all other remaining fields. For `description`, `structure_or_phase`, and `applications` fields, all of which may contain multiple values, all of the possible formula-value pairs are evaluated.

Manual evaluation. The metrics provided in prior sections demonstrate automatic and relatively strict methods for scoring NERRE tasks, but the underlying capabilities of the LLM models are best shown with manual evaluation. This is most apparent in the case of the `General-JSON` model, where exact boundaries on entities are fuzzier, precise definitions are difficult to universally define, and annotations include some implicit entity normalization. For example, the text "Pd ions were intercalated into mesoporous silica" may have equivalently have a correct `description` field for the material "silica" including "Pd-intercalated", "Pd ion-intercalated", "intercalated with Pd ions", *etc.*; the exact choice of which particular string is used as the "correct" answer is arbitrary.

To better address scoring of these fuzzy tasks, we introduce an adjusted score based on a domain expert's manual evaluation of whether the information extracted is a valid representation of the information actually contained in the passage. We term this adjusted score "manual information extraction score"; it constitutes a basis for precision, recall, and F_1 that quantifies the quality of overall information capture for cases where there may be equivalent or multiple ways of representing the same concept. This score was constructed to better estimate the performance of our model for practical materials information extraction tasks.

We score entities extracted by annotators but not present in the model's output as false negatives, except when reasonable variations are present. The criteria for a true positive are as follows:

1. The entity comes from the original passage or is a reasonable variation of the entity in the passage (*e.g.,* "silicon" $\longrightarrow$ "Si"). It is not invented by the model.

2. The entity is a root entity or is grouped with a valid root entity. For the `General-JSON` model, a root entity is either a material's formula or name. If both are present, the formula is used at the root.

3. The entity is in the correct field in the correct root entity's group (JSON object).

Manual information extraction scores are reported per-entity as if they were NER scores. However, the requirements for a true positive implicitly include relational information, since an entity is only correct if is *grouped with* the correct root entity.

III. RESULTS

We first report results of the sequence-level evaluation metrics as described in Section II in order to give the reader an idea of how likely an output sequence from an LLM-NERRE model is to be exactly correct, somewhat correct (via Jaro-Winkler), or parsable (*i.e.,* well-formatted and compliant with the schema the model was trained on). Sequence-level metrics represent at a high-level how likely the model is to "follow the rules". Next, we evaluate models on the materials NERRE tasks with more granularity for individual relations and entities, and we compare with a BERT-based proximity approach and seq2rel; these results will allow the reader to obtain a more in-depth view of model-performance. Finally, we use the `General-JSON` model to compare against BatteryBERT, and examine the effect of training set size on performance for the doping task.

Sequence-level results

Table 2 shows results for sequence-level matching. We find a wide variability between models for reconstructing test set output sequences exactly, with the simpler doping models containing exact matches on $58.4 - 64.9\%$ of test sequences.

Table 2: Sequence-level error metrics for completions for all tasks, evaluated and averaged over the test set for each task. Completions are only considered correct in the exact match if the full output sequence c is recovered exactly. Average Jaro-Winkler similarities are also shown to measure string similarity continuously on $[0, 1]$, where 1 indicates a perfect match and 0 indicates no match. Parsability, the ability to format c in the same manner as the training schema, is recorded for each sample as 0 (not parsable) or 1 (parsable). The average exact correctness, Jaro-Winkler similarity, and parsability are normalized to the range $0 - 100\%$.

Task	Model Name	Exact Match Sequence Accuracy (%)	Avg. Jaro-Winkler Similarity (%)	Parsable and Decodable (%)
Doping	`Doping-ENG`	59.7	94.3	98.7
	`Doping-JSON`	64.9	96.7	100
	`DopingExtra-ENG`	58.4	93.8	98.7
MOFs	`MOF-JSON`	12.5	92.1	99.7
General materials	`General-JSON`	25.6	91.7	99.1

The more complex tasks, `MOF-JSON` and `General-JSON`, have exact match accuracies ranging from 12.5 to 25.6% of test sequences. Exact matches are inherently less probable with longer output sequences, as even a single error in a very long output sequence results in an exact match failure. This trend is easier to see with the test samples' sequence reconstruction scores binned into groups based on the number of entities each test set true sample contains. As shown in Fig. 6 for `MOF-JSON` and `General-JSON`, exact match sequence reconstruction is the highest for the simplest entries (*i.e.,* those with the lowest number of entities).

All models have average Jaro-Winkler similarities $\geq$ 91.7% and parsabilities $\geq$ 98.7%. In contrast to the exact match score, Jaro-Winkler similarity and parsability do not degrade as rapidly with increasing complexity (for which we use true number of entities as a proxy) of the prompt. This is encouraging, as the model is able to retain consistent, parsable formatting even in very long output sequences with many (20+) interrelated and hierarchical entities.

Relation extraction performance

In Tables 3 to 5 we report the NERRE scores computed using the exact word-match basis detailed in Section II. We note that the pure NER and RE scores are somewhat intertwined because a model must correctly recognize entities in order to correctly extract a relationship between them. The scores in Tables 3-5 thus reflect the entire NERRE (NER + RE) task, not NER or RE separately. The raw NER-only scores broken down by entity for each of the tasks are shown in the Supplement in Table S1.

Doping task

We find the LLM-NERRE models have significantly higher F_1 than either the MatBERT-Doping + Proximity or seq2rel models (Table 3), though the LLM-NERRE models are not superior across all metrics (*e.g.,* recall). As an example, consider the MatBERT-Doping + Proximity model, a model which represents the performance of a state-of-the-

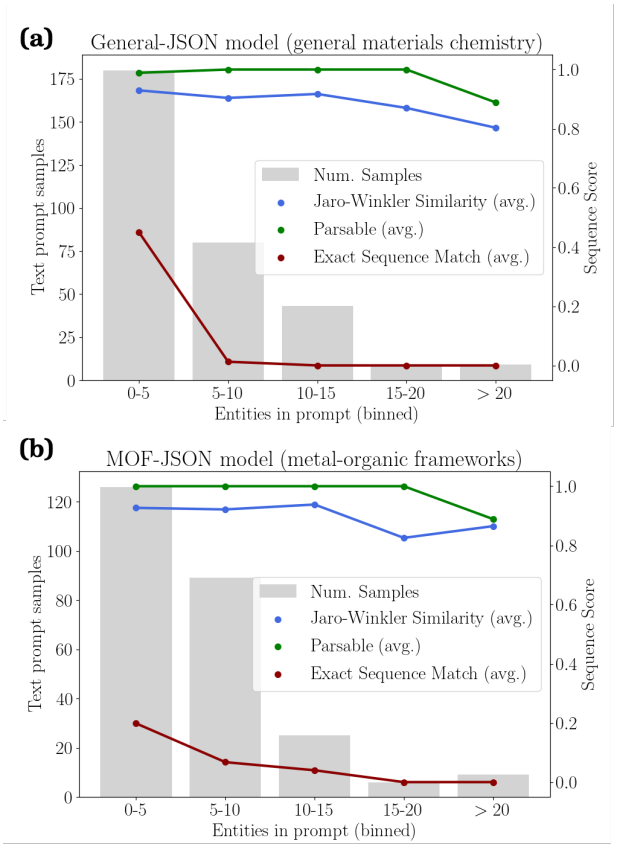

Fig. 6: Sequence reconstruction metrics segmented by number of entities in the prompt for **(a)** The `General-JSON` model and **(b)** the `MOF-JSON` model. The x axis defines the bins by the number of true entities in the test-set sample; as we progress from left to right, the bins represent more complex samples. The left y axis (corresponding to the grey bars) shows the number of samples in each bin. The right y axis (corresponding to the colored lines centered on each bin) shows the average scores of each bin by the metrics of exact string matching (red), Jaro-Winkler similarity (blue), and parsability (green). While exact string matches are very unlikely responses for the models for highly complex samples (20+ entities), the models are able to retain high sequence similarity and parsability.

art fine-tuned NER model coupled with a simple proximity co-occurrence rule for linking dopants to hosts. While this model is able to extract entity relationships from text with comparable recall (0.714) to the LLM-NERRE models, its overall F_1 is reduced by a comparatively poor precision (0.441). As shown in Supplement Table S1, this is in part caused by low NER precision for `host` entities. The discrepancy between the MatBERT-Doping model's NER performance here and the NER performance in the original publication (NER dopant $F_1 = 0.82 \pm 0.02$ and basemat $F_1 = 0.72 \pm 0.02$) [4] may be explained by our test samples being more qualitatively complex. The remaining loss in MatBERT-Doping + Proximity F_1 is due to the error inherent in the simple proximity rule defined for linking; essentially, not every host material is necessarily related to every dopant specie within a single sentence. We find higher precision (0.650) and recall (0.550) from the seq2rel model resulting in $F_1 = 0.596$ slightly higher than MatBERT-Doping + Proximity but substantially lower than any of the LLM-NERRE models. We acknowledge that the initial seq2rel model was derived from the PubMedBERT [44] pretrained BERT model as per the original implementation [36], and the seq2rel method may be improved by using a BERT model pretrained exclusively on materials text rather than biomedical text.

Notably, all three LLM-NERRE models exceed the performance of the two baselines in both NERRE and pure NER performance, regardless of schema, despite being trained on a much smaller training dataset than the MatBERT-NER model (413 sentences vs. 455 abstracts). We find $F_1 > 0.7$ across all doping models, with the best performance from the `DopingExtra-ENG` model. The higher performance for this model compared to the less complex `Doping-ENG` model may be explained by the higher entity recognition score on the `dopant` entity. The high NERRE recall, precision, and F_1 for `DopingExtra-ENG` is particularly interesting because `DopingExtra-ENG` is both more accurate and more capable (*i.e.,* extracts results and modifiers entities in addition to host-dopant relationships) than `Doping-ENG` and `Doping-JSON`, despite being trained on the same number of samples. This may be due to the model learning during training that some result/modifier entities (otherwise unlabelled in the `Doping-ENG` and `Doping-JSON models`) are not hosts or dopants, and the model predicts fewer false positive hosts or dopants. The `DopingExtra-ENG` model therefore is a notable example of asking an LLM-NERRE model to "do more" while having no significant drawback in F_1.

General and MOF tasks

As shown in Tables 4 and 5, NERRE exact word-match F_1 scores for the `General-JSON` and `MOF-JSON` models are generally lower than those of the doping models. We reiterate to the reader that these two tasks are both vastly more complex than the doping task; rather than focusing on a single essential relation (host, dopant) and its entities, these models must simultaneously extract many entity types and potential relations as groups. For example, the `General-JSON` model must simultaneously extract acronyms, names, formulae, phase labels, device applications, and material descriptions and determine all relationships between all entities. For simplicity, we report only the most essential of these relationships in the exact word-match scores, namely for links between material formula (for general) or MOF name (for MOFs) and all other entities in each model's schema. For the `General-JSON` model, the highest NERRE word-match score ($F_1 = 0.613$) was found for the material `formula` $\longrightarrow$ `name` relationship while the lowest ($F_1 = 0.341$) was found for `formula` $\longrightarrow$ `description`. The relatively lower fidelity of formula links to applications, structures, and descriptions may be caused by the generally fuzzier boundaries around these entities in the annotations, particularly in the case of description entities. For MOFs, links between guest species and specifically named MOFs are recovered most reliably ($F_1 = 0.606$) while links between MOF names and descriptions are the least reliable. This is again likely due to how reliably the expert annotator is able to delimit these entities, as guest species (*e.g.,* "Hg2+") are qualitatively easier to delimit than descriptions (*e.g.,* "mesostructured MOFs formed by Cu2+ and 5-hydroxy-1,3-benzenedicarboxylic acid").

However, these word-matching scores imply LLM-NERRE models perform worse than what is actually observed. As described in Section II, we also report manually evaluated information extraction scores (Table 6) for the general task for a 10% random subset of the total test set. These manually-evaluated scores are more representative of the true (end result, human observed) performance of the LLM-NERRE models because they account for ambiguity in how the same information can be written (e.g. "Au nanoparticles" vs "gold nanoparticles"). When manually examined by a domain expert, the information extraction F_1 scores increase dramatically for formulae (0.943 vs 0.537), materials names (0.848 vs 0.613), applications (0.832 vs 0.481), structures (0.829 vs 0.388), and descriptions (0.704 vs 0.341).

We find acronyms have the lowest manual information extraction scores, which we attribute to the fact that acronyms are relatively rare in the training class compared to the others (appearing in only 52 abstracts across the entire dataset, $\sim 9\%$ of the documents) and that material acronyms often can be confused with chemical formulas (e.g. "AuNP" is the acronym for gold nanoparticle but is also a valid chemical formula). Usually context clues are the only way to disambiguate cases like this, and including more training data with acronyms may improve the acronym extraction score further.

Table 3: Exact match NERRE scores for the (host, dopant) linking task, evaluated on a per-word basis. Links are only correct if both entities and the relationship are correct. The highest score in each category are shown in bold. Note the `DopingExtra-ENG` scores here refer to the same task as the other doping models, excluding metrics for extra information (`result`, `modifier`).

Method	Recall (Exact Match)	Precision (Exact Match)	F_1 (Exact Match)
MatBERT-Doping[4] + Proximity	0.714	0.441	0.545
seq2rel	0.65	0.55	0.60
`Doping-ENG` (this work)	0.776	0.789	0.783
`Doping-JSON` (this work)	0.684	0.758	0.719
`DopingExtra-ENG` (this work)	**0.828**	**0.872**	**0.849**

Table 4: Scores for the `General-JSON` LLM-NERRE model on the general materials information extraction task, evaluated on an exact word-match basis. Links are only correct if both entities and the relationship are correct.

Relation	Recall (Exact Match)	Precision (Exact Match)	F_1 (Exact Match)
(material formula, material name)	0.539	0.716	0.613
(material formula, acronym)	0.470	0.635	0.537
(material formula, application)	0.470	0.499	0.481
(material formula, structure)	0.368	0.411	0.388
(material formula, description)	0.304	0.392	0.341

Table 5: Scores for the `MOF-JSON` model on the the metal–organic framework (MOF) information extraction task, evaluated on an exact word-match basis. Links are only correct if both entities and the relationship are correct.

Relation	Recall (Exact Match)	Precision (Exact Match)	F_1 (Exact Match)
(MOF name, formula)	0.409	0.455	0.424
(MOF name, application)	0.461	0.560	0.504
(MOF name, guest species)	0.588	0.665	0.606
(MOF name, description)	0.247	0.464	0.318

Table 6: Manual information extraction scores for the general materials task. Scores measure the model's ability to extract inter-related data together (i.e. assigning entities correct labels and grouping them appropriately, as described in Section II.)

Entity	Extraction Recall	Extraction Precision	Extraction F_1
name	0.692	1.0	0.818
formula	0.943	0.943	0.943
acronym	0.667	0.400	0.500
applications	0.797	0.870	0.832
structure or phase	0.754	0.920	0.829
description	0.576	0.905	0.704

Effect of training set size

The effect of training set size on `Doping-ENG` test set performance is plotted in Figure 7 for training set sizes ranging from 1 to 413 text-completion pairs. We observe no parsable output sequences for training set sizes below 8 samples, but there is a sharp increase at 16 samples, potentially due to the inability of the model to produce a parsable output sequence format below a threshold number of samples. Above 64 samples, we see marginal noisy improvement. Each data point represents a single independent training/prediction experiment, and the internal model variability may explain the score variance at higher levels of training samples.

IV. DISCUSSION

The NERRE scores in Tables 3-6 provide an overview of raw performance, but the advantages and disadvantages of this method are most easily explained with a discussion on factors not directly shown with F_1 scores. The primary advantage of this method is its accessibility and ease-of-use (*Section: Accessibility and Ease-of-use*), as LLM-NERRE

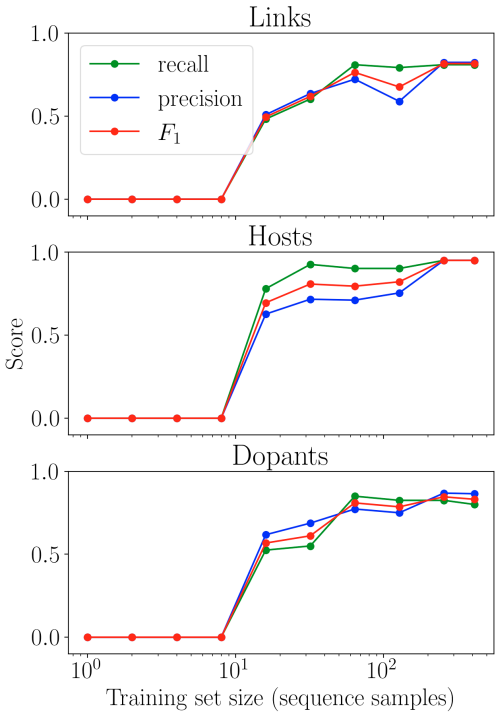

Fig. 7: Test set performance by number of training samples for the doping extraction task using the `Doping-ENG` model.

requires only specifying a basic schema, annotating a minimal number of examples, and fine-tuning a model via a publicly available API without extensive NLP knowledge or hyperparameter tuning; the final result is an accurate and extensible model with the ability to extract intricate hierarchies of semantically complex, nuanced, and specialized technical information. Additionally, error correction and normalization may be embedded directly into training examples to reduce the need for postprocessing (*Section: Implicit error correction and entity normalization*). In essence, one can show an LLM-NERRE model both what it should extract *and* how it can be condensed and presented. Finally, we discuss limitations of this approach and directions for future work. We encourage the reader to try the LLM-NERRE method themselves with an interactive demo available at `http://www.matscholar.com/info-extraction`.

Accessibility and ease-of-use

In contrast to existing pipeline-based or end-to-end NERRE methods, the LLM-NERRE method may be attractive to domain specialists as it requires little programming experience, NLP expertise, or in-depth postprocessing. Users need only to define a schema for a problem of interest, annotate examples adhering to that schema, and fine-tune a LLM model using publicly-available APIs. The flexibility of the schema also allows for highly intricate and hierarchical knowledge graphs to be extracted directly from scientific text; this is in contrast to methods requiring the enumeration of all possible entity relationships, where success depends more heavily on the tuning of output formats and requires a significant number of training examples for each entity-relation type.

We have found the in-the-loop method shown in Fig. 2 to greatly decrease annotation time as well. Typically, annotation for scientific information extraction tasks is a tedious and error-prone process. Checking intermediate model outputs for errors is qualitatively easier than creating annotations from scratch. Additionally, using GPT-3 for extraction reduces the number of total training examples needed in comparison with BERT-based models; our NERRE doping models exceed the dedicated MatBERT-NER model in entity recognition precision and F_1 despite being trained on far less text, requiring no materials-specific pretraining, and jointly linking entities (vs. MatBERT's NER-only capabilities). As shown in in Figure 7, performance of the fine-tuned models generally improves slower as the number of text-completion pairs they have been trained on increases.

As a separate experiment, we compared the usefulness of in-the-loop (error-correction) models for aiding a human domain expert in annotating novel materials-related abstracts in the `General-JSON` schema. In each trial of the experiment, the human annotator receives 10 novel abstracts and a schema pre-populated with a suggestion from an intermediate version of the `General-JSON` model trained on n samples of training data (n =1, 10, 50, 100, 300). The trained models and the human both receive the same input prompt; the task of the human annotator was to correct the intermediate model's suggestions. The time to complete each annotation was recorded. As shown in Fig. 8, the annotation time sharply decreases as the number of training samples used in the intermediate models increases; the $n = 300$ intermediate model was able to reduce the average annotation time per abstract by 57% in comparison with the $n = 1$ model. At low numbers of training samples, the models' predictions are unparsable or essentially useless, and the annotator must create annotations from scratch. At higher numbers of training samples, particularly those above 50, the intermediate model predictions require very little error correction from the annotator. As a lower bound, we also report the time needed by the annotator to simply verify whether an entry was entirely correct (verification time). Hypothetically, a perfect model creating perfect annotations would require the human annotator only to check the outputs rather than correct them. We find that by three metrics (time per abstract, time per material entry, and time per prompt token), the human annotator annotated substantially faster with a well-trained model in the loop (n samples > 50) than with a poorly trained model (n samples < 50) or no model. For example, the $n = 300$ model reduced the annotation time per token $\sim 60\%$ compared to the $n = 1$ model and is only 38% slower than the verifica-

tion time. Given additional training samples for intermediate models, we expect the annotation to asymptotically approach the verification time. Thus, this method may serve as a useful tool for building even larger benchmark datasets for information extraction tasks.

Implicit error correction and entity normalization

An advantage of the LLM-NERRE method presented here is the ability to automatically correct errors and normalize common entity patterns. While the doping models were trained to extract snippets of text exactly as they appeared in the text prompt, the `General-JSON` model's training data included simple normalizations and error corrections of entities. For example, the erroneous inclusion of whitespace in chemical formulae is common in the raw abstract text; by including a corrected formula instead of the raw string in the output training sequences, the LLM is capable of resolving the entity to a cleaner form. For example, "Li Co O2" is resolved to "LiCoO2" implicitly without additional post-processing. Similarly, given sufficient training examples, the `General-JSON` model can resolve text such as "PdO functionalized with platinum" to a normalized form such as `{formula: "PdO", description: ["Pt-functionalized"]}`. These implicit normalization and correction abilities of LLM models may prove useful for domain specialists who desire structured entity formats rather than exact string excerpts directly from the text.

Limitations

One limitation of our model is that valid output schema formatting is not rigorously enforced in the generation step. The LLM may, for any given sample, output an unparsable sequence. This is particularly apparent when the inference token limit is < 512 tokens and the schema is JSON, as JSON schema typically requires a larger number of tokens for correct formatting. For example, a nearly-correct output sequence containing 10+ correct entities may be missing a "}" character and therefore will not be parsable. Outputs are nearly always parsable ($\sim 95\%$ success rate), especially as the number of training examples increases. Failures predominantly occur when the sample exceeds the prompt-completion token limit of GPT-3, which at the time of this work is 2048. Because of this, some abstracts that are too long or too dense with information can not be processed with this method. We observe that this was the case in the vast majority of the unparseable completions where the passage and partial completion exceed the token limit and cut off early before the full completion could be output by the model.

Another limitation is the tendency of LLMs to generate or invent information that is not actually present in the input text, which is called "hallucination" in LLM literature. However, in some cases a properly conditioned language model

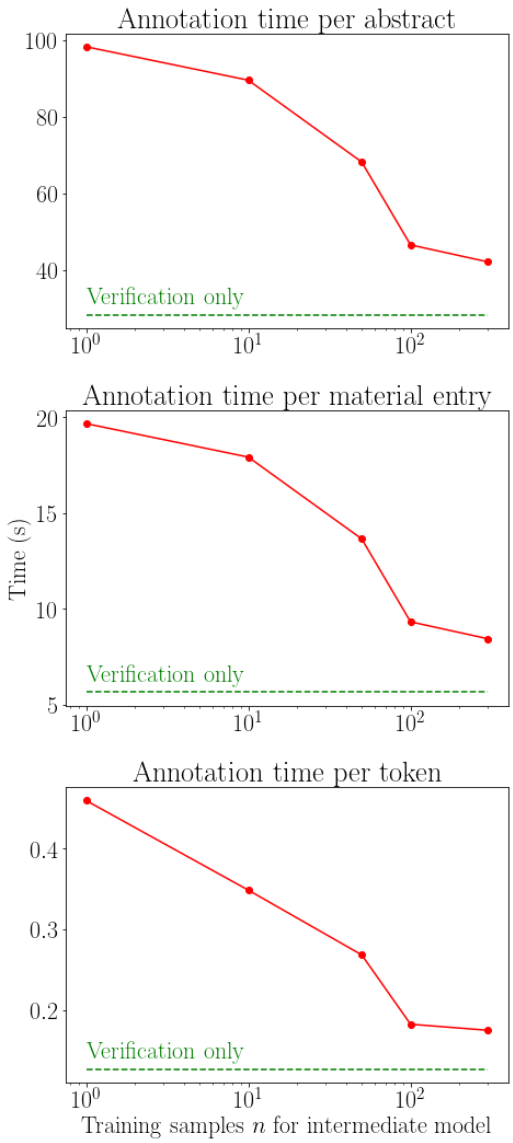

Fig. 8: Time taken for a domain expert to annotate new abstracts for the general materials chemistry task with assistance from intermediate (partially-trained) LLM-NERRE models. At each data point, an intermediate model trained on n samples sampled from the original training set (where n is shown on the x axis) infers the completion given the same text prompt presented to the annotator. The pre-populated annotation is then corrected by the annotator. Resulting times for each annotation are shown per abstract, per material mention (JSON document), and per prompt token. The green verification line represents the time taken for the annotator to simply verify whether a given annotation is entirely correct. The verification line represents a lower bound on the annotation time; even with a perfectly-performing model, the annotator must still take time to verify the annotation.

may produce useful and correct information that would reasonably be inferred by the reader. This is not possible in strict NERRE models which simply extract entities by labeling words in the original text. However, this means it is also possible for a model to extract the same information from a piece of text in multiple ways. For example, an abstract may mention "p-ZnSe doped with N" and "nitrogen-doped ZnSe" in the same passage. Is "doped with N" or "nitrogen-doped" the correct description to extract? Clearly both are correct and either one could be reasonably chosen. Moreover, "N-doped" could also be extracted and would be factually correct even though "N-doped" never occurs in the passage. For this reason, most approaches to information extraction require that the exact phrases used in the original text be extracted, but this can introduce complications in downstream post-processing. One example is the cases where "catalyst", "catalysis", "catalyzed", and "catalyzes" are all extracted as applications for Pt from different source passages, which are more difficult to disambiguate later. For this reason, we decided to prefer normalization of entities during extraction in this work.

Finally, GPT-3 is currently only accessible through OpenAI's online API. This may present difficulties from several standpoints, as the underlying model is controlled exclusively by a private entity and data must be sent to that entity for processing. Moreover, the cost for inference on large datasets using trained models may be prohibitive. However, given the "black-box" nature of this approach, this prompt/-completion engineering method is not exclusive to GPT-3 in principle. We expect open-source *seq2seq* models of sufficient complexity to be able to reproduce or exceed the results shown here.

V. Conclusion

We present a facile prompt and completion-engineering method for extracting complex, hierarchical, and highly domain-specific relation information from unstructured scientific text using large *seq2seq* language models. This method is highly flexible and we expect it can be easily adapted to many problems within scientific domains. In applying this method, we find excellent performance on three diverse tasks for materials engineering: solid-state impurity doping, metal-organic frameworks, and general materials relations. The non-technical nature of this approach implies scientists without NLP training can utilize existing *seq2seq* models such as GPT-3 to extract large structured relational datasets for highly-specific problems. As the LLM is treated essentially as a black box, we expect this approach may be used LLMs other than GPT-3, including LLMs released in the near future. We hope this approach enables domain specialists to rapidly extract relational datasets for the advancement of scientific knowledge. An online demo is available at `http://www.matscholar.com/info-extraction`.

VI. Author Contributions

J.D., A.D., and N.W. developed the information extraction method presented in this paper and J.D. and A.D. collected the abstract dataset used. J.D originated and performed supporting experiments to justify the sequence-to-sequence approach and use of GPT-3 for document-level information extraction from materials science text. N.W. and J.D. developed the sequence-to-JSON method. A.D. created the doping schemas, developed the sequence-to-sentence method, and annotated the doping dataset. J.D. created the general materials information schema, annotated the general materials dataset, trained the general materials information extraction model, and manually scored the information extraction results for the General-JSON model and the BatteryBERT validation set results. A.D. trained the Doping-JSON, Doping-ENG, models and implemented the MatBERT + Proximity doping model. S.L. trained and collected data for the seq2rel model. A.D. performed the learning curve experiments, and collected data on annotation times. A.S.R. and J.D. co-created the MOF schema and annotated the MOF dataset while J.D. trained the MOF-JSON model. A.J. contributed to task design, task scoring metrics, and overall research directions. G.C., K.P., and A.J. supervised this work. All authors contributed to writing the manuscript.

VII. Acknowledgements

This work was supported by Toyota Research Institute through the Accelerated Materials Design and Discovery program. A.S.R. acknowledges support via a Miller Research Fellowship from the Miller Institute for Basic Research in Science, University of California, Berkeley. We thank Anna Sackmann (Science Data and Engineering Librarian at UC Berkeley) for helping us to obtain Text and Data Mining agreements with the specified publishers and we also thank J. Montoya and A. Trewartha for helpful discussions.

References

[1] J. E. Saal, A. O. Oliynyk, and B. Meredig, "Machine learning in materials discovery: Confirmed predictions and their underlying approaches," *Annual Review of Materials Research*, vol. 50, no. 1, pp. 49–69, Jul. 2020. [Online]. Available: https://doi.org/10.1146/annurev-matsci-090319-010954

[2] K. Choudhary, B. DeCost, C. Chen, A. Jain, F. Tavazza, R. Cohn, C. W. Park, A. Choudhary, A. Agrawal, S. J. L. Billinge, E. Holm, S. P. Ong, and C. Wolverton, "Recent advances and applications of deep learning methods in materials science," *npj Computational Materials*, vol. 8, no. 1, Apr. 2022. [Online]. Available: https://doi.org/10.1038/s41524-022-00734-6

[3] L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K. A. Persson, G. Ceder, and A. Jain, "Named entity recognition and normalization applied to large-scale information extraction from the materials science literature," *Journal of Chemical Information and Modeling*, vol. 59, no. 9, pp. 3692–3702, Jul. 2019. [Online]. Available: https://doi.org/10.1021/acs.jcim.9b00470

[4] A. Trewartha, N. Walker, H. Huo, S. Lee, K. Cruse, J. Dagdelen, A. Dunn, K. A. Persson, G. Ceder, and A. Jain, "Quantifying the advantage of domain-specific pre-training on named entity recognition tasks in materials science," *Patterns*, vol. 3, no. 4, p. 100488, Apr. 2022. [Online]. Available: https://doi.org/10.1016/j.patter.2022.100488

[5] T. Isazawa and J. M. Cole, "Single model for organic and inorganic chemical named entity recognition in ChemDataExtractor," *Journal of Chemical Information and Modeling*, vol. 62, no. 5, pp. 1207–1213, Feb. 2022. [Online]. Available: https://doi.org/10.1021/acs.jcim.1c01199

[6] X. Zhao, J. Greenberg, Y. An, and X. T. Hu, "Fine-tuning BERT model for materials named entity recognition," in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2021. [Online]. Available: https://doi.org/10.1109/bigdata52589.2021.9671697

[7] O. Sierepeklis and J. M. Cole, "A thermoelectric materials database auto-generated from the scientific literature using chemdataextractor," *Scientific data*, vol. 9, p. 648, 10 2022.

[8] E. J. Beard and J. M. Cole, "Perovskite- and dye-sensitized solar-cell device databases auto-generated using chemdataextractor," *Scientific Data*, vol. 9, 12 2022.

[9] P. Kumar, S. Kabra, and J. M. Cole, "Auto-generating databases of yield strength and grain size using chemdataextractor," *Scientific Data*, vol. 9, 12 2022.

[10] S. Huang and J. M. Cole, "BatteryBERT: A pretrained language model for battery database enhancement," *Journal of Chemical Information and Modeling*, May 2022. [Online]. Available: https://doi.org/10.1021/acs.jcim.2c00035

[11] Q. Dong and J. M. Cole, "Auto-generated database of semiconductor band gaps using chemdataextractor," *Scientific Data*, vol. 9, 12 2022.

[12] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[13] O. Kononova, H. Huo, T. He, Z. Rong, T. Botari, W. Sun, V. Tshitoyan, and G. Ceder, "Text-mined dataset of inorganic materials synthesis recipes," *Scientific Data*, vol. 6, no. 1, Oct. 2019. [Online]. Available: https://doi.org/10.1038/s41597-019-0224-1

[14] H. Huo, C. J. Bartel, T. He, A. Trewartha, A. Dunn, B. Ouyang, A. Jain, and G. Ceder, "Machine-learning rationalization and prediction of solid-state synthesis conditions," *Chemistry of Materials*, vol. 34, no. 16, pp. 7323–7336, Aug. 2022. [Online]. Available: https://doi.org/10.1021/acs.chemmater.2c01293

[15] T. He, W. Sun, H. Huo, O. Kononova, Z. Rong, V. Tshitoyan, T. Botari, and G. Ceder, "Similarity of precursors in solid-state synthesis as text-mined from scientific literature," *Chemistry of Materials*, vol. 32, no. 18, pp. 7861–7873, Aug. 2020. [Online]. Available: https://doi.org/10.1021/acs.chemmater.0c02553

[16] Z. Wang, O. Kononova, K. Cruse, T. He, H. Huo, Y. Fei, Y. Zeng, Y. Sun, Z. Cai, W. Sun, and G. Ceder, "Dataset of solution-based inorganic materials synthesis procedures extracted from the scientific literature," *Scientific Data*, vol. 9, no. 1, May 2022. [Online]. Available: https://doi.org/10.1038/s41597-022-01317-2

[17] S. Huang and J. M. Cole, "A database of battery materials auto-generated using ChemDataExtractor," *Scientific Data*, vol. 7, no. 1, Aug. 2020. [Online]. Available: https://doi.org/10.1038/s41597-020-00602-2

[18] E. J. Beard, G. Sivaraman, Á. Vázquez-Mayagoitia, V. Vishwanath, and J. M. Cole, "Comparative dataset of experimental and computational attributes of UV/vis absorption spectra," *Scientific Data*, vol. 6, no. 1, Dec. 2019. [Online]. Available: https://doi.org/10.1038/s41597-019-0306-0

[19] J. Zhao and J. M. Cole, "A database of refractive indices and dielectric constants auto-generated using ChemDataExtractor," *Scientific Data*, vol. 9, no. 1, May 2022. [Online]. Available: https://doi.org/10.1038/s41597-022-01295-5

[20] G. Bekoulis, J. Deleu, T. Demeester, and C. Develder, "Joint entity recognition and relation extraction as a multi-head selection problem," *Expert Systems with Applications*, vol. 114, pp. 34–45, 2018.

[21] X. Han, T. Gao, Y. Lin, H. Peng, Y. Yang, C. Xiao, Z. Liu, P. Li, M. Sun, and J. Zhou, "More data, more relations, more context and more openness: A review and outlook for relation extraction," 4 2020. [Online]. Available: http://arxiv.org/abs/2004.03186

[22] Y. Yao, D. Ye, P. Li, X. Han, Y. Lin, Z. Liu, Z. Liu, L. Huang, J. Zhou, and M. Sun, "DocRED: A large-scale document-level relation extraction dataset," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 764–777. [Online]. Available: https://aclanthology.org/P19-1074

[23] J. Li, Y. Sun, R. J. Johnson, D. Sciaky, C. H. Wei, R. Leaman, A. P. Davis, C. J. Mattingly, T. C. Wiegers, and Z. Lu, "Biocreative v cdr task corpus: a resource for chemical disease relation extraction," *Database : the journal of biological databases and curation*, vol. 2016, 2016.

[24] Àlex Bravo, J. Piñero, N. Queralt-Rosinach, M. Rautschka, and L. I. Furlong, "Extraction of relations between genes and diseases from text and large-scale data analysis: Implications for translational research," *BMC Bioinformatics*, vol. 16, 2 2015.

[25] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022.

[26] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti, E. Zhang, R. Child, R. Y. Aminabadi, J. Bernauer, X. Song, M. Shoeybi, Y. He, M. Houston, S. Tiwary, and B. Catanzaro, "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model," 2022. [Online]. Available: https://arxiv.org/abs/2201.11990

[27] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer, "Opt: Open pre-trained transformer language models," 2022.

[28] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, "Training compute-optimal large language models," 3 2022. [Online]. Available: http://arxiv.org/abs/2203.15556

[29] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, "Finetuned language models are zero-shot learners," 9 2021. [Online]. Available: http://arxiv.org/abs/2109.01652

[30] BIG-bench collaboration, "Beyond the imitation game: Measuring and extrapolating the capabilities of language models," *In preparation*, 2021. [Online]. Available: https://github.com/google/BIG-bench/

[31] R. Dabre, C. Chu, and A. Kunchukuttan, "A survey of multilingual neural machine translation," *ACM Computing Surveys*, vol. 53, 9 2020.

[32] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, "Language models as knowledge bases?" 9 2019. [Online]. Available: http://arxiv.org/abs/1909.01066

[33] J. M. Han, I. Babuschkin, H. Edwards, A. Neelakantan, T. Xu, S. Polu, A. Ray, P. Shyam, A. Ramesh, A. Radford, and I. Sutskever, "Unsupervised neural machine translation with generative language models only," 2022. [Online]. Available: https://openreview.net/forum?id=SVwbKmEg7M

[34] H. Zhang, J. Xu, and J. Wang, "Pretraining-based natural language generation for text summarization," *arXiv preprint arXiv:1902.09243*, 2019.

[35] Z. Liu, M. Patwary, R. Prenger, S. Prabhumoye, W. Ping, M. Shoeybi, and B. Catanzaro, "Multi-stage prompting for knowledgeable dialogue generation," 2022. [Online]. Available: https://arxiv.org/abs/2203.08745

[36] J. Giorgi, G. Bader, and B. Wang, "A sequence-to-sequence approach for document-level relation extraction," in *Proceedings of the 21st Workshop on Biomedical Language Processing*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 10–25. [Online]. Available: https://aclanthology.org/2022.bionlp-1.2

[37] P.-L. H. Cabot and R. Navigli, "REBEL: Relation extraction by end-to-end language generation," in *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, 2021. [Online]. Available: https://doi.org/10.18653/v1/2021.findings-emnlp.204

[38] B. Townsend, E. Ito-Fisher, L. Zhang, and M. May, "Doc2dict: Information extraction as text generation," 5 2021. [Online]. Available: http://arxiv.org/abs/2105.07510

[39] Y. Xie, J. Shen, S. Li, Y. Mao, and J. Han, "Eider: Empowering document-level relation extraction with efficient evidence extraction and inference-stage fusion," 6 2021. [Online]. Available: http://arxiv.org/abs/2106.08657

[40] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference*, G. Varoquaux, T. Vaught, and J. Millman, Eds., Pasadena, CA USA, 2008, pp. 11 – 15.

[41] B. J. Bucior, A. S. Rosen, M. Haranczyk, Z. Yao, M. E. Ziebel, O. K. Farha, J. T. Hupp, J. I. Siepmann, A. Aspuru-Guzik, and R. Q. Snurr, "Identification schemes for metal–organic frameworks to enable rapid search and cheminformatics analysis," *Crystal Growth & Design*, vol. 19, no. 11, pp. 6682–6697, 2019.

[42] X. Li, M. R. Hill, H. Wang, and H. Zhang, "Metal–organic framework-based ion-selective membranes," *Advanced Materials Technologies*, vol. 6, no. 10, p. 2000790, 2021.

[43] Q. Qian, P. A. Asinger, M. J. Lee, G. Han, K. Mizrahi Rodriguez, S. Lin, F. M. Benedetti, A. X. Wu, W. S. Chi, and Z. P. Smith, "Mof-based membranes for gas separations," *Chemical reviews*, vol. 120, no. 16, pp. 8161–8266, 2020.

[44] Y. Gu, R. Tinn, H. Cheng, M. Lucas, N. Usuyama, X. Liu, T. Naumann, J. Gao, and H. Poon, "Domain-specific language model pretraining for biomedical natural language processing," 2020.

[45] V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K. A. Persson, G. Ceder, and A. Jain, "Unsupervised word embeddings capture latent knowledge from materials science literature," *Nature*, vol. 571, pp. 95–98, 2019. [Online]. Available: http://dx.doi.org/10.1038/s41586-019-1335-8

[46] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing, "Jupyter notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, Eds. IOS Press, 2016, pp. 87 – 90.

[47] W. E. Winkler, "The state of record linkage and current research problems," *Statistics of Income Division, Internal Revenue Service Publication R99/04*, 1999. [Online]. Available: http://www.census.gov/srd/www/byname.html

VIII. Supplementary Information

Ambiguity of annotations

In practice, annotation can be a complex task even for trained researchers. We have found qualitatively that the performance of the LLM-NERRE models is limited by how consistently and comprehensively the desired schema and entities can be defined. As opposed to canonical NLP examples, where distinctions between entities such as "person" and "place" are relatively clear, scientific texts often contain entities which could plausibly be considered as one or more kinds of entity depending on definition. Reducing ambiguity in the annotation schema is therefore a source of potential improvement for LLM-NERRE methods. We examine several tasks below in regards to ambiguity.

Doping task ambiguity. Definitional ambiguity is particularly apparent in cases where long-range dopant-host relationships may be referenced with no clear "correct" answer to either the annotator or model. However, most sentence-level dopant-host relationships are very clear to annotators. As mentioned in the main text, this is a primary motivator for using a sentence-level annotation scheme rather than an abstract-level annotation scheme.

General task schema ambiguity. In the `General-JSON` schema, entity class definitions are not entirely consistent throughout the training set. Class definitions drifted somewhat as annotators found more edge cases during the annotation process, and it was still not possible to completely capture all relevant information contained in a passage with this schema. Keeping these details in mind, annotation was done on a "best effort" basis, which seems to still be sufficient to train models capable of complex information extraction tasks. Additionally, this schema is not currently optimized for composite materials (*e.g.,* "F-TiO2/NO3-layered double hydroxide composite"). We annotated examples such that components of the composite were sometimes listed as separate materials and sometimes listed as a single material. While this partially contributes to lower string-match F_1 scores, we find the inconsistencies in the annotations does not significantly impair the model's ability to deal with composites.

`Doping-ENG` schema

The English sequence completion schema for the `Doping-ENG` and `DopingExtra-ENG` models are shown below. Each paradigm represents a single line in the output completion; aside from the "No information" paradigm, there may be one or more of each of the paradigms in the output sequence (*e.g.,* there may be multiple separate one-to-many host-dopant relationships, a single dopant with no host, multiple results, and multiple modifiers extracted from the same sentence). The placeholders `<HOST>`, `<DOPANT*>`, `result`, and `modifier*` are used in place of actual string literal entities. The apostrophe "`'`" character was used to more easily delimit entity captures. Output sequences not matching one of these patterns were considered not parsable.

- **One host to one-or-more dopants:** `The host '<HOST>' was doped with '<DOPANT1>', '<DOPANT2>', ... and '<DOPANTN>'.`
- **Single dopant, no host**: `'<DOPANT>' is a dopant.`
- **Single host, no dopant**: `The host '<HOST>' was doped.`
- **Single result**: `'<result>' is a possible doped result formula.`
- **One or more modifiers**: `Modifiers of the doping are '<modifier1>', '<modifier2>'... '<modifierN>'.`
- **No doping-related information:** `There is no doping information.`

Sequence reconstruction metrics

Exact Match Accuracy. The exact sequence match accuracy is defined based on the an exact match between a predicted completion sequence $\hat{c}_i$ and true completion sequence c_i, averaged over n all samples to be evaluated:

$$\text{Exact match accuracy} = \frac{\sum_i^n \delta_{\hat{c}_i,c_i}}{n} \tag{4}$$

Where δ is the Kronecker delta:

$$\delta_{\hat{c}_i,c_i} = \left\{ \begin{array}{ll} 0, & \text{if } \hat{c}_i \neq c_i \\ 1, & \text{if } \hat{c}_i = c_i \end{array} \right\} \tag{5}$$

Thus, the exact match accuracy is the most stringent metric, as any character addition (*e.g.,* addition of extra whitespace), missing character, or permutation of entries (*e.g.,* a different ordering of otherwise correct JSON documents within the output) is not an exact match. This is a lower bound on the information capture of the models, as correct information may be equivalently represented multiple ways.

Jaro-Winkler Similarity. For a more granular analysis, we use the Jaro-Winkler similarity, Φ, as a string comparator metric. As in the exact sequence match, we average over all $i = 1,2,3,...,n$ evaluation sequences where the predicted string completion is labelled $\hat{c}_i$ and the corresponding true string completion is c_i. With weights of the first string, second string, and transposition all set equal, the similarity Φ_i between a predicted completion string $\hat{c}_i$ and a true completion string c_i is defined as:

$$\Phi_i = \begin{cases} 0, & \text{if } m = 0 \\ \frac{1}{3}\left(\frac{m}{|\hat{c}_i|} + \frac{m}{|c_i|} + \frac{m-t}{m}\right), & \text{otherwise} \end{cases} \tag{6}$$

Table S1: Exact match (E.M.) named entity recognition scores for the three materials engineering tasks. Highest scores are shown in bold for entities predicted with multiple models (`host`, `dopant`).

Task	Model Name	Entity	E.M. Recall	E.M. Precision	E.M. F_1
Doping	`Doping-ENG`	`host`	**0.951**	**0.928**	**0.940**
		`dopant`	0.775	0.837	0.805
	`Doping-JSON`	`host`	0.888	0.858	0.872
		`dopant`	0.714	0.750	0.732
	`DopingExtra-ENG`	`host`	**0.951**	0.907	0.929
		`dopant`	**0.850**	**0.919**	**0.883**
		`results`	0.736	0.933	0.824
		`modifiers`	0.500	0.333	0.400
	`MatBERTDoping` + Proximity	`host`	0.885	0.547	0.67
		`dopant`	0.727	0.600	0.658
	`seq2rel`	`host`	0.805	0.458	0.584
		`dopant`	0.600	0.600	0.600
General Materials	`General-JSON`	`acronym`	0.557	0.699	0.613
		`applications`	0.700	0.732	0.715
		`name`	0.588	0.778	0.668
		`formula`	0.642	0.696	0.664
		`structure_or_phase`	0.581	0.639	0.608
		`description`	0.484	0.507	0.494
MOFs	`MOF-JSON`	`name`	0.690	0.752	0.718
		`mof_formula`	0.412	0.509	0.454
		`applications`	0.682	0.722	0.701
		`guest_species`	0.614	0.754	0.666
		`description`	0.386	0.544	0.446

Where m is the number of matching characters between $\hat{c}_i$ and c_i, t is the number of transpositions, and $|\hat{c}_i|$ and $|c_i|$ are the lengths of the predicted and true completions, respectively. The final average Jaro-Winkler similarity $\bar{\Phi}$ is the arithmetic mean averaged over n samples, $\bar{\Phi} = \sum_i^n \Phi_i / n$.

Parsability. As a final sequence reconstruction metric, we report the average percentage of samples which can be parsed from string literal into object form. For the `*-JSON` models, this indicates that the output sequence is well-formatted JSON; for `*-ENG` models, this indicates the output sequence adheres to the natural-language like schema on which the model was trained. In either case, the parsability of the output sequences simply indicates whether the sequence can be easily transformed into a relational object. If a sequence can be parsed using the same function to encode training samples, we return a parsability of one; otherwise, parsability is zero. We average the parsability over all n samples of the evaluation dataset to calculate a final average parsability percentage.

Performance on named entity recognition

We show in Table S1 the named entity recognition scores for recall, precision, and F_1 score for each of the models. For the doping task, the highest scores per common entity category (`host`, `dopant`) are shown in bold, as these tasks are evaluated with multiple models (`Doping-ENG`, `Doping-JSON`, and `DopingExtra-ENG`). Entities are evaluated on an exact per-word match basis rather than the basis of exact matches between entire multi-word entities, as it is unclear even to annotators exactly where to denote the end of some complex multi-word entities (e.g., "ZnO nanoparticle crystals" vs. "ZnO nanoparticle" vs. "ZnO"). However, since compositions are the core part of the desired data for all tasks, all words of any formula entity (and hence all its links) are marked incorrect if the entire composition is not captured exactly. Support for each class may be found in Supplementary Section VIII.

In the doping task, we generally observe the highest scores for host recognition from the `Doping-ENG` model and the highest scores for dopant recognition from the `DopingExtra-ENG` model. F_1 scores for the `Doping-JSON` model are 7-9% lower than those of the `Doping-ENG` model. Between the `DopingExtra-ENG` and `Doping-ENG` models, we find a 1% difference in `host` F_1 in favor of the simpler `Doping-ENG` model but 8% higher `dopant` F_1 in favor of the `DopingExtra-ENG` model. This is a notable result as the `DopingExtra-ENG` model must additionally extract `results` and `modifiers` entities in the output sequence. We speculate that this discrepancy is due to a combination of model variability and extra specification of non-dopant chemical species within the `DopingExtra-ENG` model's training set.

For example, when the `DopingExtra-ENG` model is trained with "high-doping" as a labelled `modifier`, it may learn "high" is not a valid dopant entry. This may be useful in the design of more complex completion schema, as `DopingExtra-ENG` is an example of a *more* complex completion schema extracting all of its entities more accurately.

GPT-3 inference parameters

All doping models were trained with 7 epochs. Intermediate models shown in Figure 7 were trained with a number of epochs depending on the number of training samples t: 2 epochs for $2^0 \leq t < 2^6$, 4 epochs for $2^6 \leq t \leq 2^7$, and 7 epochs for $t = 2^8$. The remaining models were trained with a batch size of 1 for 4 epochs with start sequence `"\n\n\n###\n\n\n"` and stop sequence `"\n\n\nEND\n\n\n"`. All doping models used GPT-3 inference parameters of 256 max tokens, 0 temperature, `"\n\n\n###\n\n\n"` start token, and `"\n\n\nEND\n\n\n` end token. All other models used 512 max tokens with remaining parameters identical to the doping models.

seq2rel parameters

Seq2rel models were trained using 267 doping sentences that had dopant-basemat links, with 4 different training:validation splits (90:10, 80:20, 70:30, 95:5) and the model with the highest validation micro-F1 score was selected. Training configurations for seq2rel used parameters for training gene-disease association (GDA) used in Giorgi et al.[36], while modifying the entity tokens to `"@DOPANT@"`,`"@BASEMAT@"` and relation token to `"@DBR@"`. To be specific:

```
model_name = microsoft/BiomedNLP-PubMedBERT-
    base-uncased-abstract-fulltext
max_length=512
max_steps=96
num_epochs=30
batch_size=1
grac_acc_steps=1
decoder_lr=5e-4
encoder_lr=2e-5
encoder_wd=0.01
reinit_layers=1
weight_dropout=0.5
beam_size=4
length_penalty=0.8
```

Doping task graph example

In Figure S1 an example end result from a LLM-NERRE model is shown in graph format. We aim here to recognize not just relationships between individual entities, but hierarchical relationships with relationship types which need not be explicitly and comprehensively enumerated beforehand. The LLM-NERRE models presented in the main text provide a step towards this kind of final product.

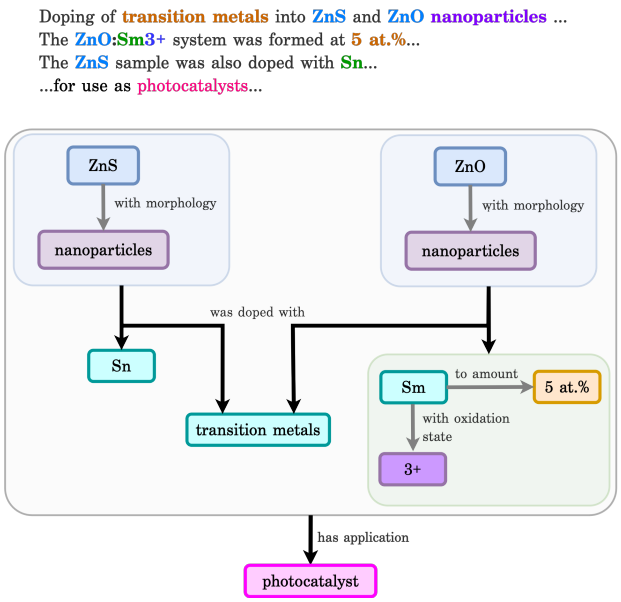

Fig. S1: An example complex graph resolved from the outputs of a LLM-NERRE model. The distinction of this graph from typical entity relationship graphs is the hierarchical format. This hierarchical graph, in contrast to a flat graph, may denote that ZnO *nanoparticles* - as opposed to bulk wurtzite ZnO - were doped by representing an entity with its own subgraph. Similarly, the samarium dopant is represented as a subgraph specifying its oxidation state and amount. Finally, all dopant relationships for ZnO and ZnS are linked to an application "photocatalyst". This hierarchical graph specifies a much more precise series of relationships extracted from text; for example, *"ZnO (as nanoparticles) was doped with Sm (having an oxidation state of +3, and to an amount of 5 atomic percent) resulting in a photocatalyst."* is more precise than a flat-graph relationship such as *"ZnO is a photocatalyst"*.

Class support for materials extraction tasks

Table S2: Class support for doping tasks among 77 test (gold) set sentences.

Class/Relation	Support
`host`	76
`dopant`	60
`host` → `dopant` (link)	72
`results`	12
`modifiers`	7

Table S3: Class support for the general materials NERRE task from 320 total test set abstracts, including the core entities and the subset of potential links shown in results.

Class/Relation	Support
`acronym`	61
`applications`	527
`name`	192
`formula`	386
`structure_or_phase`	402
`description`	288
`formula` → `acronym` (link)	14
`formula` → `application` (link)	49
`formula` → `name` (link)	49
`formula` → `structure_or_phase` (link)	368
`formula` → `description` (link)	199

Table S4: Class support for the MOF task from 255 total test set abstracts including the core entities and the subset of potential links shown in results.

Class/Relation	Support
`application`	953
`guest_species`	205
`description`	178
`name`	434
`mof_formula`	91
`mof_formula` → `application` (link)	171
`mof_formula` → `guest_species` (link)	21
`mof_formula` → `description` (link)	44
`mof_formula` → `name` (link)	29