Supporting information for: Named Entity Recognition and Normalization Applied to Large-Scale Information Extraction from the Materials Science Literature

L. Weston,[1] V. Tshitoyan,[2] J. Dagdelen,[1], O. Kononova,[2] A. Trewartha,[2] K. A. Persson,[1], G. Ceder[2] and A. Jain[1]

[1]Lawrence Berkeley National Laboratory, Energy Technologies Area, 1 Cyclotron Road, Berkeley, CA 94720, United States

[2]Lawrence Berkeley National Laboratory, Materials Science Division, 1 Cyclotron Road, Berkeley, CA 94720, United States

S.1: Document preprocessing

Parts of abstracts (or full abstracts) that were in foreign languages were removed using text search and regular expression matching, as were articles with metadata types corresponding to “Announcement”, “BookReview”, “Erratum”, “EditorialNotes”, “News”, “Events” and “Acknowledgement”. Abstracts with titles containing keywords “Foreword”, “Prelude”, “Commentary”, “Workshop”, “Conference”, “Symposium”, “Comment”, “Retract”, “Correction”, “Erratum” and “Memorial” were also selectively removed from the corpus. Some abstracts contained leading or trailing copyright information, which were removed using regular expression matching and heuristic rules. Leading words and phrases such as “Abstract: ” were also removed using similar methods. Abstracts were tokenized using ChemDataExtractor.[1] Tokens that were identified as valid chemical formulae were normalized, such that the order of elements and common multipliers did not matter (e.g. NiFe is the same as Fe50Ni50); this was achieved using using pymatgen[2] combined with regular expression and rule based techniques. Valence states of elements were split into separate tokens (e.g. Fe(III) became two separate tokens, Fe and (III)). Additionally, if a token was not a chemical formula or an element symbol, and if only the first letter was uppercase, we lowercased the word. This way chemical formulae as well as abbreviations stayed in their common form, whereas words at the beginning of sentences as well as proper nouns were converted to lowercase. Numbers with units were often not tokenized correctly ChemDataExtractor. We addressed this in the processing step by splitting the common units from numbers and converting all numbers to a special token <nUm>. This reduced the vocabulary size by approximately 20,000 words. We found that correct preprocessing, especially the choice of phrases to include as individual tokens, significantly improved the quality of embeddings and subsequent named entities.

S.2: Word2vec training

We used the Word2vec implementation in *gensim*[3] with a few modifications. We found that the Skip-gram model with negative sampling loss (n=15) performed the best. The vocabulary was limited to words that occurred more than 5 times, however, the chemical formulas were included independent on their count. The rest of the hyperparameters were as follows: we used 200-dimensional embeddings, learning rate of 0.01 decreasing to 0.0001 in 30 epochs, context window of 8 and subsampling with 10^{-4} threshold, which subsamples approximately 400 most common words. Hyperparameters were optimized for performance on approximately 15,000 grammatical and 15,000 materials science analogies, with the score defined as the percentage of correctly "solved" analogies from the two sets. The full list of analogies used in this study is available at https://www.github.com/materialsintelligence/mat2vec.

S.3: Relevance annotation

For annotating abstracts as either "relevant" or "not relevant", the following guidelines are used.

1. The abstract must mention at least one inorganic material.
2. Studies involving organic/soft-matter materials (e.g., polymers), or biomaterials, are not considered relevant.
3. The abstract should mention either the synthesis or characterization of a specific inorganic material.
4. An exception to (3) is review papers discussing applications of a specific material.

S.4: Named entity annotation

The rules for annotating each entity type are shown below.

1. **Material (MAT):** Any inorganic solid or alloy, any non-gaseous element (at RT), e.g., “BaTiO3”, “titania”, “Fe”.

2. **Symmetry/phase label (SPL):** Names for crystal structures/phases, e.g., “tetragonal”, ‘fcc”, “rutile”, “perovskite”; or, any symmetry label such as “*Pbnm*”, or “*Pnma*”.

3. **Sample descriptor (DSC):** Special descriptions of the type/shape of the sample. Examples inlcude “single crystal”, “nanotube”, “quantum dot”.

4. **Property (PRO):** Anything measurable that can have a unit and a value, e.g., “conductivity”, “band gap”; or, any qualitative property or phenomenon exhibited by a material, e.g., “ferroelectric”, “metallic”.

5. **Application (APL):** Any high-level application such as “photovoltaics”, or any specific device such as “field-effect transistor”.

6. **Synthesis method (SMT):** Any technique for synthesising a material, e.g., “pulsed laser deposition”, “solid state reaction”, or any other step in sample production such as “annealing” or “etching”.

7. **Characterization method (CMT):** Any method used to characterize a material, experiment or theory: e.g., “photoluminescence”, “XRD”, ‘tight binding”, “DFT”. It can also be a name for an equation or model. such “Bethe-Salpeter equation”.

Overall, we annotated 800 materials science abstracts. This consisted of 22,306 annotated entities (out of 111380 tokens total).

S.5: Entity normalization feature generation

Entity normalization is achieved using supervised machine learning. Features are generated by concatenating the word embeddings for each entity. We also derive several hand-crafted features; these features describe the following properties of the entity pair:

1. edit distance (Levenshtein distance)

2. embedding cosine similarity

3. stemming/lemmatization (are the stems/lemmas equal)

4. synonyms/antonyms (are the entities known synonyms or antonyms according to WordNet[4])

S.6: Entity normalization training data

Acquiring training data for entity normalization is a significant challenge. There are of the order of roughly 10^4 unique entities for each entity type, and therefore there are of the order 10^8 possible entity pairs for each type. In most cases, each entity will have one (or no) synonyms; as a result, if entity pairs were generated randomly, around 10^4 entities would be labelled as negative (i.e., not a synonym) for every positive case that was generated. Thus, labelling of training data would be far too time consuming in this manner.

In order to overcome this problem, we generate training data in the following way: We begin with 200 hand-crafted entity pairs with approximately balanced classes. This is used to train a model that classifies synonyms; Logistic Regression was used when the dataset was small, and a Random Forest classifier was used once the dataset grew larger. Next, a large number (> 1000) of randomly generated entity pairs are created. The classifier is used to determine which of these entity pairs are synonyms; any pair that the model identifies as a synonym is sent to the annotator to label. Initially, most of the predictions are incorrect, however the initial model does correctly identify some new synonym pairs. The process is repeated iteratively, and at each step the amount of training data is increased. Over many cycles, the precision of the model improves, and the model begins to make mostly correct predictions. In the end, this process yielded 10000 annotated entity pairs, with roughly balanced classes.

S.7: Model accuracy vs. training set size

The final model is optimized with a training set of 620 abstracts. We have performed a series of experiments in which the model is assessed as a function of training set size. The results are shown in Fig. S1. The model performance quickly increases as more abstracts are added to the training set. However, the performance does not significantly improve between a training set size of 500 and 620, suggesting that additional data will not provide better accuracy with the current model architecture.

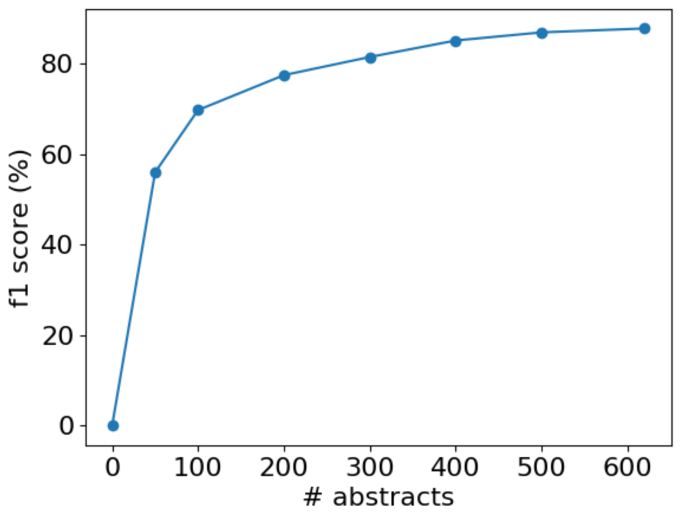

FIG. S1: Model accuracy ($f1$) as a function of the number of labelled abstracts in the training set.

S.8: Test set accuracy metrics

In this section we show a more detailed analysis (precision, recall, $f1$) of our final model on the test set.

TABLE 1: Accuracy metrics for the test set.

Label	precision (p)	recall (r)	$f1$
Total	86.35	87.74	87.04
MAT	87.33	93.47	90.30
SPL	87.65	77.14	82.05
DSC	90.31	94.03	92.13
PRO	85.46	81.03	83.19
APL	83.69	77.78	80.63
SMT	79.12	83.75	81.37
CMT	83.32	88.88	86.01

References

(1) Swain, M. C.; Cole, J. M. Chemdataextractor: a Toolkit for Automated Extraction of Chemical Information from the Scientific Literature. *J. Chem. Inf. Model.* **2016**, *56*, 1894–1904.

(2) Ong, S. P.; Richards, W. D.; Jain, A.; Hautier, G.; Kocher, M.; Cholia, S.; Gunter, D.; Chevrier, V. L.; Persson, K. A.; Ceder, G. Python Materials Genomics (pymatgen): A Robust, Open-source Python Library for Materials Analysis. *Comp. Mater. Sci.* **2013**, *68*, 314–319.

(3) `https://radimrehurek.com/gensim/`.

(4) Miller, G. A. WordNet: a Lexical Database for English. *Communications of the ACM* **1995**, *38*, 39–41.