POLLYDARTON – NLP-DRIVEN DATA CURATION FOR POLYMER NANOCOMPOSITE DATA RESOURCE

Jordan Axelrod, Defne Çirci, Logan Cooper, Thomas Lilly, Shota Miki
Duke University
{jordan.axelrod, defne.circi, logan.d.cooper, thomas.lilly, shota.miki}@duke.edu

ABSTRACT

Materials science researchers spend significant time sifting through articles to identify relevant methods and properties that could aid their own research. We can model this challenge as a multi-document search, where each publication is a document. However, such a system would perform poorly without labels for sentences indicating whether they are actions, constituents, or properties due to the way scientific articles are typically structured. We propose a wide range of models with acceptable performance that could be used depending on system constraints. We find that finetuned pretrained models exceed the performance of methods previously tested. These results highlight the importance of using transfer learning when working with limited and unbalanced datasets.

1 INTRODUCTION

Scientific publications are considered to be the most reliable source of information about the processing and construction of new materials. Because searching through these publications by hand is time-consuming, experimental data about the structure and properties of the materials has been made available in several manually curated databases. This has led to the development of techniques to automate how we learn about materials, for example, with machine learning algorithms that can predict specific properties – such as electrical, mechanical, or thermal ones – from the structure of a material.

The relationship between processing/synthesis routes and the materials produced is another important focus for materials scientists. In order to reproduce the materials described in an article, the processing steps need to be followed precisely. Even for new materials discovery, these processing steps can guide or inspire new production recipes. Once again, the manual search for these recipes is a bottleneck to data extraction. NLP offers an opportunity to automate the information extraction from these articles. This project aims to facilitate optimized synthesis procedures of polymer nanocomposites by finding the most relevant sentences in a given article. These will be classified as "experimental work" by the algorithm. The code for this project can be found here.

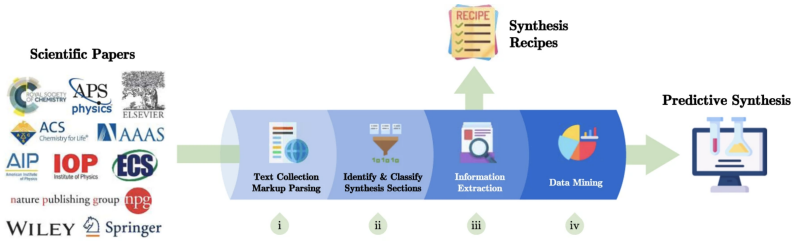

Figure 1: Representation of the standard text mining pipeline: (i) scrape papers in markup format from major article publishers; (ii) identify and classify synthesis sections; (iii) extract key information including materials, amounts, sequenced operations, and conditions; (iv) store synthesis procedures into a database for future analysis Wang et al. (2022).

Our process closely follows that of Wang (2019), who applied several standard text embedding techniques and classification algorithms to a dataset of 2,000 labeled sentences (see section 3 for more details).

2 Related Work

2.1 Applications of NLP in Materials Science

A plethora of materials science articles with valuable information is already available online. Obtaining the data from these articles has been challenging, as they are not presented in a way that a machine can understand. NLP has been used in the materials science domain for the last decade to tackle this problem. Initially, the focus was primarily on chemical texts. Examples of algorithms trained on these include ChemicalTagger (Hawizy et al., 2011),ChemDataExtractor (Swain & Cole, 2016), and OSCAR (Gallarati et al., 2022). ChemDataExtractor was developed to extract chemical information from scientific documents. It uses a corpus of $\sim$ 68,000 chemistry and physics papers to generate a database of around $\sim$ 40,000 chemical compound records and associated magnetic phase transition temperatures (Court & Cole, 2018). This work planted the first seeds of combining materials science and NLP by providing a tokenizer and chemical entity extraction algorithm that are transferable to this domain (Shetty & Ramprasad, 2021b;a). Recently, scientists have been working on utilizing existing articles to create materials synthesis and processing databases. So far, researchers have taken this approach to metal oxides, germanium-containing zeolites, perovskites, and solution-based inorganic materials. An annotation schema of synthesis procedures has also been developed (Mysore et al., 2019; Kuniyoshi et al., 2020). As a result, it has become possible to create new synthesis recipes and find sets of parameters that can lead to new materials synthesis or replicate existing ones (Kononova et al., 2019; Huo et al., 2019; Kim et al., 2017a;b; 2019; Jensen et al., 2019). Materials science corpora are also used for training word embedding models (Kim et al., 2020; 2017c). Notably, this includes a word2vec embedding model using 3.3 million scientific abstracts (Tshitoyan et al., 2019).

Moreover, fine-tuning a pre-trained language model for specific tasks has been shown to improve results (Olivetti et al., 2020). This has motivated scientists to develop two BERT models (pre)trained on materials science literature: MATBert and MatSciBert (Trewartha et al., 2022; Gupta et al., 2022). MatSciBert is trained on 150,000 full-text articles with a focus on inorganic materials, while MATBert is trained on two million articles to understand the specific language used in materials science. It can also do paragraph-level scientific reasoning. Both are trained on papers in the inorganic materials domain, but they are different in terms of the types of inorganic materials present in the training data.

2.2 Applications of NLP in Polymer Nanocomposites

As mentioned in the previous section, the focus has been on inorganic materials. Polymer nanocomposites are organic materials and have garnered the attention of many scientists due to their unique physical properties, such as melting temperature and stiffness (Zhao et al., 2016). These unique properties are due to their structure of small amounts of nanoparticles suspended inside a polymer. Scientists have created a framework to extract processing information from scientific articles (Wang, 2019; Hu, 2022), see also section 2.1. The framework that Wang (see section 1) uses consists of four steps: filtering out the irrelevant papers from the article database, identifying paragraphs to select those with a focus on materials processing, classifying sentences in the processing paragraphs depending on their meanings, and lastly, extracting exact experimental procedure and relevant conditions. We are going to focus on the third step of this process in our project.

3 Approach

3.1 Background

Our approach to this topic was heavily influenced by a PhD dissertation by Wang (2019). This paper uses a dataset of "2000 sentences [from materials science papers] that [have been] manually labeled by material experts" (ibid.), a sample of which is available in appendix A.1. The four labels are:

1. Materials Constituents (`constituents`): The materials or chemicals that form a polymer (e.g., resin, nanoparticles, metals, solvents, etc.).
2. Materials Properties (`properties`): The physical properties of a material (e.g., thickness, viscosity, tensile strength, melting point, etc.)
3. Experimental Action (`action`): What researchers did to/with a polymer in the course of their experiments (e.g., heating, mixing, casting, drying, synthesizing, etc.)
4. Not Relevant (`unrelated`): All other sentences in a paper (e.g., references to other papers or sections, filler sentences, vague descriptions, etc.)

The goal is to train an algorithm on this data to determine which category a given sentence falls into.

3.2 Methods

Wang uses a mixture of techniques to achieve this. These techniques fall into two broad categories: classification techniques and embedding techniques. For classification, the paper uses Logistic Regression (LR), Support Vector Machine (SVM), and an Attention Neural Net[1]. For embedding, it uses Bag of Words (BOW), TF-IDF, and word2vec.

Our first goal was to reproduce the results of the original paper. We did so by using the LR and SVM models as well as the BOW and TF-IDF vectorizors available in scikit-learn (Pedregosa et al., 2011), building a neural network in PyTorch (Paszke et al., 2019), and using a word2vec model based on the Gensim implementation (Rehurek & Sojka, 2011)[2], but trained further on articles in the materials science space (Kim et al., 2017c) using a transfer learning approach. Although using Kim et al.'s word embeddings – which are trained on 640,000 full-text metallic material synthesis articles – is not ideal due to the differences in these types of materials, it still holds some relevance because of similarities in experimental procedures.

3.3 Extensions

In addition, we tried several methods to improve the performance of the original models. The first was to perform more preprocessing. Specifically, we introduced several types of new special characters that group rare words with the same function[3]. We also performed large amounts of hyperparameter tuning on all of our models. Finally, we experimented with pretraining using RoBERTa (Liu et al., 2019). Given the large size of the BERT models relative to the training data, we decided to keep the post-encoder classifier simple. This approach did not utilize complex preprocessing or embeddings prior to byte pair encoding.

4 Experiments

4.1 Datasets and Evaluation

As mentioned in section 3.1, the dataset was the same that was used by Wang (2019). It consists of 2,000 hand-labeled sentences from over 100 different materials/polymer papers generated by the Brinson Group at Duke (Wang et al., 2022). The original paper uses the F1 score as its evaluation metric, so we use the same.

One challenge with the dataset was its imbalance. It had by far more action sentences than anything else. The exact counts were:

- Action: 592
- Constituent: 173
- Unrelated: 120
- Property: 65

[1] See appendix A.4 for examples of attention scores for both the neural net and word2vec embeddings.
[2] Version=3.8.1.
[3] See appendix A.2 for more details.

4.1.1 Additional Data

We were also provided some additional data by Dr. Bingyin Hu from the Brinson Group at Duke University. This dataset comprised 2,958 polymer nanocomposite articles published by the American Chemical Society. It was provided in a JSON format, so we were able to find relevant articles by searching the keys for "Experimental Section." This yielded an additional 6,882 sentences across 219 articles. We labeled these with a simple rule-based model, which looked for notable words in the sentences (e.g., specific verbs for Actions) descriptors of properties for Properties, and terms associated with manufacturing for constituents.

The label proportions were:

- Action: 1455
- Constituent: 307
- Unrelated: 4329
- Property: 791

However, testing this rule-based model against our original data performed very poorly. Table 1 and figure 2 illustrate this. However, limited human evaluation by domain-knowledgeable individuals approved of the labels this model gave the new data, noting that many of the sentences were somewhat ambiguous (i.e., talked about both constituents and actions in one sentence).

Label	F1 Score
Action	0.22
Constituent	0.13
Unrelated	0.22
Property	0.46

Table 1: Results of predicting the original data with our rule-based model. The scores are unaveraged F1 scores for each class. Results rounded to two decimal places.

Figure 2: Confusion matrix for the rule-based model on the original data. The scores are percentages of the true labels. Chart generated with seaborn (Waskom, 2021).

4.2 Model and Training Details

For the LR and SVM models, Wang (2019) did not provide any information on the hyperparameters. This includes the regularization term for the LR model and the kernel for the SVM. For these

specific parameters, we ran some experiments to find out which choices gave the best (i.e. closest to the original) results. This resulted in us using the defaults for LR, i.e. LBFGS solver and L2-regularization and a sigmoid kernel for SVM [4].

Our neural network follows the architecture described by Wang (2019) as well. The network has three segments. The first is an input segment which takes a tokenized input sentence and embeds it with pre-trained word embeddings. The second portion is an encoder, which consists of a bidirectional gated recurrent unit (GRU). Finally, we use an attention mechanism to determine which words are important for classification, and run the results through a softmax function to do prediction. For this, we chose an Adam optimizer and cross-entropy loss function.

We used several BERT models on the smaller training data we have. These include RoBERTa, scientific language-aware language model SciBERT (Beltagy et al., 2019) and material science aware language models MatSciBERT (Trewartha et al., 2022), MatBERT (Gupta et al., 2022) [5]. While previous works use the version of MatBERT hosted on Huggingface (Wolf et al., 2020), namely `allenai/scibert_scivocab_uncased`, `m3rg-iitd/matscibert` and `roberta-large`. The MatBERT model we use was downloaded from the GitHub repository for publication.

4.3 Results

4.3.1 Reproduction

In general, we are able to reproduce Wang (2019), see table 2. Although some of our results are slightly off the original ones (specifically LR + BOW and SVM + word2vec), the rest are relatively close. We chalk up most of the difference to differences in our train-test split and unreported hyperparameters in Wang (2019).

Method	F1 Score (Reproduction)	F1 Score (Wang)
LR + BOW	0.86	0.81
LR + TF-IDF	0.78	0.79
LR + word2vec	0.78	0.80
SVM + BOW	0.85	0.82
SVM + TF-IDF	0.83	0.83
SVM + word2vec	0.79	0.85
Attention NN	0.84	0.84
Attention NN + word2vec	0.85	0.88

Table 2: Results of our reproduction compared to results reported in Wang (2019). All decimals are rounded to two decimal places.

4.3.2 Preprocessing and Tuning

The first additions we made involved increasing the amount of preprocessing that we did on the text. The bulk of this meant the addition of more special characters to the text; for example: replacing integers with `<int>`, decimals with `<dec>`, and temperatures with `<temp>`. In addition to that, we also removed many line breaks which were present in the original data and unaccounted for in the original paper. A more complete breakdown is available in appendix A.2.

The hyperparameter tuning was relatively simple. We used randomized search cross-validation (with scikit-learn's built-in `RandomizedSearchCV`) to tune the hyperparameters of the different models. We focused on tuning the regularization term and the class weighting. Additionally, we scaled the different types of feature vectors using `StandardScalar`.

[4] Our presentation used an RBF kernel SVM for reproduction, which is one of the reasons why the reproduction results here differ from those reported earlier.

[5] For SciBERT, MatSciBERT and MatBERT, following hyperparameters are used: batch size = 32, learning rate = 5e-5, epochs = 10, dropout = 0.1

Method	+ Preprocessing	+ Tuning
LR + BOW	0.82 (-0.04, +0.01)	0.83 (-0.03, +0.02)
LR + TF-IDF	0.81 (+0.03, +0.02)	0.79 (+0.01, +0.00)
LR + word2vec	0.83 (-0.02, -0.04)	0.78 (+0.00, -0.02)
SVM + BOW	0.73 (-0.10, -0.10)	0.69 (-0.15, -0.13)
SVM + TF-IDF	0.75 (-0.08, -0.08)	0.72 (-0.11, - 0.11)
SVM + word2vec	0.76 (-0.03, -0.09)	0.62 (-0.17, -0.23)
Attention NN	0.84 (+0.00, +0.00)	N/A
Attention NN + word2vec	0.86 (+0.01, -0.02)	N/A

Table 3: Results of our improvements to the models from Wang. The parenthesized values are the change from our reproduced baseline and from Wang's baseline in that order. All decimals are rounded to two decimal places.

From the results in table 3, we see that neither of these approaches was especially effective in improving the performance of the model [6]. While the preprocessing tends to lead to small gains for the LR and Attention methods, it leads to large losses for SVM.

4.3.3 Additional Data

Despite the problems with the labeling of the additional data, we tried splitting it into train and test sets and appending those to our original training and test sets. The effect of the additional data was negligible for all but the neural network, which showed some modest improvement. The exact scores are reported in appendix A.3.

4.3.4 RoBERTa Model

Since the available training data was somewhat limited, we thought that a pre-trained model with finetuning would yield better results than the attention-based neural network. RoBERTa was selected because it naturally fits our classification task as an encoder model since we don't require text generation. It remains among the highest-ranked single models in many benchmark datasets, including SQuAD, MNLI-m, and SST-2 (Liu et al., 2019). The following hyperparameters were used for finetuning: max length = 128, batch size = 8, learning rate = 1e-5, epochs = 14, dropout = 0.3. Training also used a linear decay learning rate scheduler with a 10% warmup period. The hidden layer's size matched the output's size, making it 768 for `roberta-base` and 1024 for `roberta-large`.

Method	F1 Score
roberta-base	0.90
roberta-base + preprocessing	0.92
roberta-large	0.92
roberta-large + preprocessing	0.93

Table 4: F1 Score performance of fine-tuned RoBERTa model with single pre-classification hidden dimension. The highlighted differences are between model size and custom data preprocessing. The weighted average was used to calculate the multi-class F1 score.

4.3.5 Additional Models

We experimented with several additional models, namely Random Forest, XGBoost, and an alternative attention neural network. XGBoost, in particular, was employed to help with the problem of imbalanced data. In this experiment, sentences were featurized using bag of words. No pretrained models were used (i.e., word embeddings for the neural model were learned from scratch). The

[6] In the initial presentation, we were using validation results, not test results, which ended up being much lower.

Attention neural network was configured with the following: trigram embedding with dim size = 8, self-attention layer, dropout = 0.3, averaged embedding outputs, tanh activation function, direct projection to classification layer, log softmax for class probabilities.

In this experiment, even a moderate size neural model easily overfitted to the training data, hence the small size of the employed models. Also, using LSTM blocks instead of trigrams made the performance of the model unstable. As we can see in table 5, the results of these models were at or below the baseline values in table 2, and were therefore abandoned early on.

Model	w/o preprocessing	w/ preprocessing
Random forest	0.720	0.767
XGBoost	0.809	0.815
Attention on trigrams	0.813	0.831

Table 5: Evaluation of additional models. Results are accuracies averaged across three-fold cross-validation on the training set.

4.3.6 SciBERT, MatSciBERT, MatBERT

The results can be seen in Table 6.

Model	F1 Score
SciBERT	0.91
MatSciBERT	0.91
MatBERT	0.92

Table 6: Evaluation of scientific language aware models. Results are weighted-averaged F1 scores.

5 Analysis

5.1 Preprocessing and Tuning

As noted in section 4.3.2, adding preprocessing to our models improves performance while tuning degrades typically leads to overfitting. This would seem to suggest that our approach to hyperparameter tuning was acting as an avenue for overfitting rather than allowing the model to zero in on more general relationships in the training data. It might also suggest that Logistic Regression and SVM are poorly-suited to the task at hand.

We believe that preprocessing was able to improve the results by reducing the number of tokens the model had to handle and as a result, making it easier for the models to ”attach meaning” to tokens like temperatures and decimals. As an example, the model should not need to know whether a polymer was heated to 100 degrees or 1000 degrees, both should still result in an ”action” label.

5.2 Additonal Data

While the additional data had problems with it, the fact that it improved the performance of the neural network models does carry some promise. The new data were classified with an imperfect rule-based model, and yet the neural nets were able to correctly identify many points both the original and additional data. This might imply these neural models are able to learn the sorts of common-sense rules that we were able to identify. A more robust rule-based model (and perhaps even a learned one, i.e. a decision tree) might be able to make more progress here.

5.3 RoBERTa

The most limiting aspect of this model choice was the byte pair encoding used for tokenization. This made it more difficult to experiment with other types of pre-trained embeddings without signif-

icantly increasing the complexity of the model and training time. The most significant preprocessing improvements were achieved by removing line breaks and replacing temperature numbers with a generic tag. The most significant hyperparameters were the batch size and dropout rate. There didn't appear to be any systemic error patterns in mispredicted sentences. The model struggled with variants of the sentence, "The chemicals were obtained from the following sources and used without further purification." Otherwise, the errors seem indistinguishable from what a human would make. With more time and computing power, we would have liked to try an ensembled version of this model.

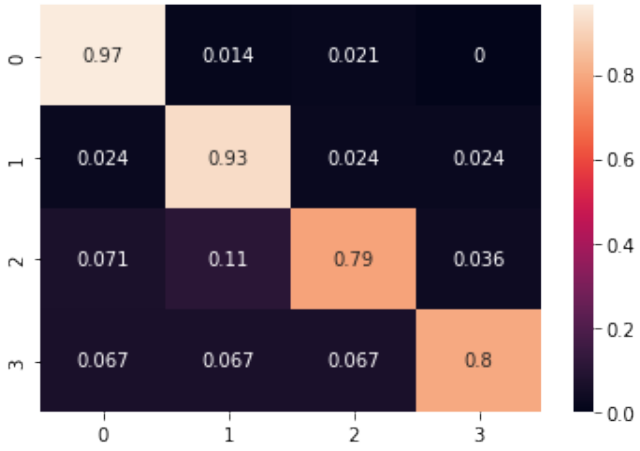

Figure 3: Confusion matrix for RoBERTa pre-trained variant. The labels are defined as follows: Action = 0, Constituent = 1, Unrelated = 2, Property = 3.

6 Conclusion

In recent years, NLP has become applicable to the field of materials science. One particular way that it can be applied is by picking specific properties and experiments out of published papers for inclusion in public databases of materials. We attempted to replicate and improve one attempt at this by Wang (2019). We were able to replicate Wang's results. We tried several methods to improve Wang's models, and found that the best ways to do so were to one: stick to attention-based neural networks; two: add preprocessing to the text; and three: use pretrained word embeddings such as RoBERTa. Through these methods, we were able to bring the top F1 scores of Wang's models from 0.88 to around 0.92. While these slight improvements are useful, they may suggest that this task is nearing its point of diminishing returns.

Author Contributions

- **Jordan Axelrod:** Paper reproduction (LR, SVM, Neural Network)
- **Defne Çirci:** Idea, Literature Review, Collecting more training data, Experimentation with SciBERT, MatSciBERT and MatBERT, Presentation, Final Report (writing)
- **Logan Cooper:** Text data processing (tokenization, etc.), Experimentation with SVM and Logistic Regression, rule-based model testing, Presentation, Final Report (writing + revising)
- **Thomas Lilly:** RoBERTa experimentation, Final Report (related sections, revising)
- **Shota Miki:** Advanced preprocessing, additional models (boosting etc.), Final Report (revising)

References

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

Callum J Court and Jacqueline M Cole. Auto-generated materials database of curie and néel temperatures via semi-supervised relationship extraction. *Scientific data*, 5(1):1–12, 2018.

Simone Gallarati, Puck van Gerwen, Ruben Laplaza, Sergi Vela, Alberto Fabrizio, and Clemence Corminboeuf. Oscar: an extensive repository of chemically and functionally diverse organocatalysts. *Chemical Science*, 13(46):13782–13794, 2022.

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

Lezan Hawizy, David M Jessop, Nico Adams, and Peter Murray-Rust. Chemicaltagger: A tool for semantic text-mining in chemistry. *Journal of cheminformatics*, 3(1):1–13, 2011.

Bingyin Hu. *Data Curation of a Findable, Accessible, Interoperable, Reusable Polymer Nanocomposites Data Resource - Materialsmine*. PhD thesis, 2022.

Haoyan Huo, Ziqin Rong, Olga Kononova, Wenhao Sun, Tiago Botari, Tanjin He, Vahe Tshitoyan, and Gerbrand Ceder. Semi-supervised machine-learning classification of materials synthesis procedures. *npj Computational Materials*, 5(1):1–7, 2019.

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

Edward Kim, Kevin Huang, Stefanie Jegelka, and Elsa Olivetti. Virtual screening of inorganic materials synthesis parameters with deep learning. *npj Computational Materials*, 3(1):1–9, 2017a.

Edward Kim, Kevin Huang, Adam Saunders, Andrew McCallum, Gerbrand Ceder, and Elsa Olivetti. Materials synthesis insights from scientific literature via text extraction and machine learning. *Chemistry of Materials*, 29(21):9436–9444, 2017b.

Edward Kim, Kevin Huang, Alex Tomala, Sara Matthews, Emma Strubell, Adam Saunders, Andrew McCallum, and Elsa Olivetti. Machine-learned and codified synthesis parameters of oxide materials. *Scientific data*, 4(1):1–9, 2017c.

Edward Kim, Kevin Huang, Olga Kononova, Gerbrand Ceder, and Elsa Olivetti. Distilling a materials synthesis ontology. *Matter*, 1(1):8–12, 2019.

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

Olga Kononova, Haoyan Huo, Tanjin He, Ziqin Rong, Tiago Botari, Wenhao Sun, Vahe Tshitoyan, and Gerbrand Ceder. Text-mined dataset of inorganic materials synthesis recipes. *Scientific data*, 6(1):1–11, 2019.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Sheshera Mysore, Zach Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. *arXiv preprint arXiv:1905.06939*, 2019.

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

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

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

Pranav Shetty and Rampi Ramprasad. Automated knowledge extraction from polymer literature using natural language processing. *Iscience*, 24(1):101922, 2021a.

Pranav Shetty and Rampi Ramprasad. Machine-guided polymer knowledge extraction using natural language processing: The example of named entity normalization. *Journal of Chemical Information and Modeling*, 61(11):5377–5385, 2021b.

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

Amalie Trewartha, Nicholas Walker, Haoyan Huo, Sanghoon Lee, Kevin Cruse, John Dagdelen, Alexander Dunn, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. Quantifying the advantage of domain-specific pre-training on named entity recognition tasks in materials science. *Patterns*, 3(4):100488, 2022.

Vahe Tshitoyan, John Dagdelen, Leigh Weston, Alexander Dunn, Ziqin Rong, Olga Kononova, Kristin A Persson, Gerbrand Ceder, and Anubhav Jain. Unsupervised word embeddings capture latent knowledge from materials science literature. *Nature*, 571(7763):95–98, 2019.

Yixing Wang. *Combining Finite Element with Data Analytical Approaches for Structure-Property Modeling in Polymer Nanocomposites*. PhD thesis, 2019.

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

Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6 (60):3021, 2021. doi: 10.21105/joss.03021. URL https://doi.org/10.21105/joss.03021.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

He Zhao, Xiaolin Li, Yichi Zhang, Linda S Schadler, Wei Chen, and L Catherine Brinson. Perspective: Nanomine: A material genome approach for polymer nanocomposites analysis and design. *APL Materials*, 4(5):053204, 2016.

A Appendices

A.1 Data Sample

Label	Text
action	The products were dried under vacuum at 60 °C for 24 h.
action	In this process, the magnetic stirring lasted 2 h at room temperature.
constituent	The resin to curing agent mass ratio is 1:1 for stoichiometric curing.
constituent	Lexan 121 (General Electric) was chosen for the polymer matrix.
property	The thickness of the films was 10-20 μm.
property	The selected flexibilizer was a low viscosity polyglycol.
unrelated	Three steps were used to prepare the nanocomposites.
unrelated	The details of sample formulations are discussed in [15].

Table 7: A small sample of the training dataset

A.2 Preprocessing Details

- Remove line breaks (e.g. "elec- tron" → "electron").
- Lower case (e.g. "Material" → "material").
- Separate following symbols from other words: ['(', ')', '[', ']', ',', ';', ':', '/', '%', '+', '-', '', '–', '','° c', '°c', '∘ c', '∘c', 'μ', 'ml'] (e.g. "(12" → "(12").
- Unify various expressions of temperature (e.g. '°c', '∘ c' → "⟨temp⟩").
- Unify various expressions of integers (e.g. '60', '2' → "⟨int⟩")
- Unify various expressions of decimals (e.g. '0.5', '1.5' → "⟨dec⟩").
- Unify various expressions of ratios (e.g. '1:1', '1:1.5' → "⟨ratio⟩").

A.3 Additional Data Improvement

Method	F1 Score
LR + BOW	0.81
LR + TF-IDF	0.79
LR + word2vec	0.77
SVM + BOW	0.74
SVM + TF-IDF	0.79
SVM + word2vec	0.63
Attention NN	0.88
Attention NN + word2vec	0.88

Table 8: Results of the baseline models with the additional data. All decimals rounded to two decimal places.

A.4 Attention Weight Examples

Note that the sentence in these examples was labelled as an action sentence.

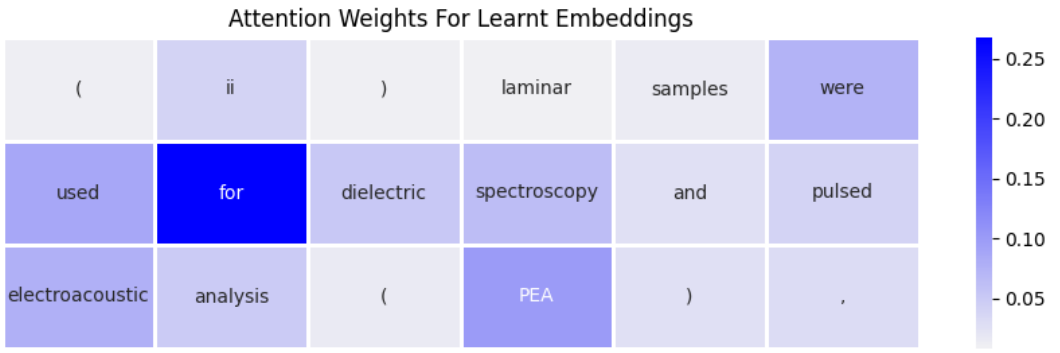

Figure 4: Heatmap of the attention for learned word embeddings.

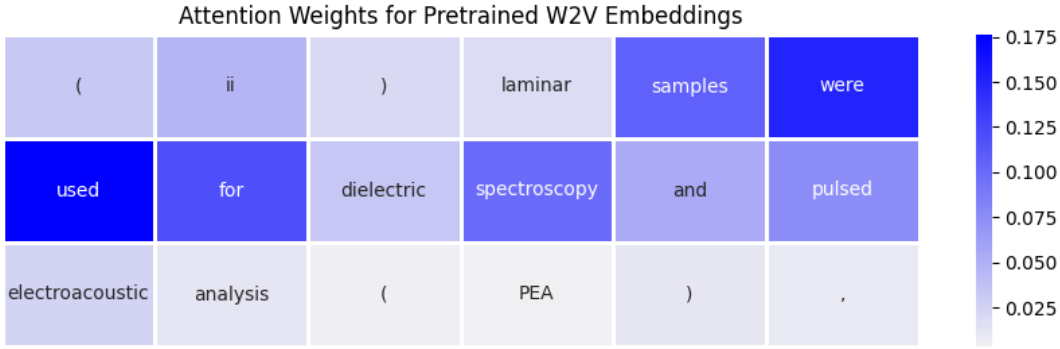

Figure 5: Heatmap for the attention for word2vec embeddings.