

# Document Classification with Termolator for COVID-19 Literature

Muyan Jiang  
New York University Abu Dhabi  
Abu Dhabi, UAE  
mj2259@nyu.edu

Runyao Fan  
New York University Abu Dhabi  
Abu Dhabi, UAE  
rf1888@nyu.edu

Omar Hussein  
New York University Abu Dhabi  
Abu Dhabi, UAE  
oah242@nyu.edu

**Abstract**—The COVID-19 pandemic has led to a multiplicity of research publications related to various aspects of coronavirus. Research topics range from COVID-19 transmission mechanisms to the public health response of various countries, and the publications need to be categorized for easier and more efficient access to resources. This paper explores various machine learning-based document classification techniques to categorize COVID-19 related literature. We integrate a novel terminology dictionary with machine learning models to study the dictionary's impact on the effectiveness of various classification techniques. We report a slight boost to F1 scores as a result of our modifications.

**Index Terms**—epidemic, COVID-19, document classification, natural language processing

## I. INTRODUCTION

The COVID-19 pandemic has had wide-ranging impacts on our world. Researchers around the globe have devoted themselves to investigating different academic areas related to the pandemic, ranging from clinical practice to epidemiological analysis. Many publications have been made available for people to collaborate together and tackle the virus. LitCovid[1][2], a curated literature hub for tracking up-to-date scientific information about the coronavirus, indicates that more than 170,000 papers have been published by the time we write this paper.

Nowadays, with the abundance of available publications on COVID-19, it is particularly important for researchers to find and gauge relevant literature as easily and efficiently as possible. Therefore, we seek to develop a suitable document classification system focusing on COVID-19 literature. Specifically, we investigate whether having a terminology dictionary improves the efficacy of traditional machine learning approaches to document classification.

## II. LITERATURE REVIEW

There has been significant work done to apply different document classification techniques to biomedical data to help researchers identify relevant information to a high degree of accuracy. With the high rate at which COVID-19 related literature is published, medical scientists cannot keep up with new publications and have to use Named Entity Recognition

(NER) and Named Entity Normalization (NEN) to identify relevant publications [3].

One of the most popular paradigms currently used in the application of NLP models to classify biomedical data is the “pretrain-and-finetune” approach [4]. Indeed, researchers have shown some success in applying this model to COVID-19 data classification tasks [5]. Specifically, their research indicates that there are measurable benefits of having a dedicated biomedical vocabulary base for biomedical document classification [5].

An additional approach to document classification has been the application of aspect-based document similarity measures. This has enabled researchers to perform pairwise document classification to identify aspects in which papers are more similar in order to achieve more fine-grained classification [6].

The literature indicates that BioBERT works best as a classifier for COVID literature from the LitCovid database we utilize (achieving an F1 score of 86.1), followed by pre-trained language models and traditional machine learning algorithms in descending order of effectiveness [5].

## III. DATASET

### A. LitCovid

LitCovid is a dataset with more than 170,000 COVID-19 related papers collected and manually classified according to the following labels: “General”, “Transmission Dynamics (Transmission)”, “Treatment”, “Case Report”, “Epidemic Forecasting (Forecasting)”, “Prevention”, “Mechanism”, and “Diagnosis”. From [1], we obtain a dataset with 52,419 entries for the training file, 8,226 entries for testing, and 6,582 entries for validation. Each entry contains the information of one paper including ID, journal name, title of the paper, abstract, keywords, label, publication type, authors, date, doi, label type, etc. This information, specifically title, abstract, and keywords, is used later for feature creation in our model.

### B. NCBI Disease Corpus

We utilize the NCBI Disease Corpus when generating terminology dictionaries for each of our classification categories. From [7], we collect the NCBI Disease Corpus. It is a fully annotated biomedical research resource which contains 793 PubMed abstracts.

## IV. METHOD

### A. Data Pre-processing

The LitCovid dataset is available as CSV files with fields including pmid, journal, title, abstract, keywords, label, publication type, authors, date1, doi, and date2. The label field corresponds to each entry's category (e.g. "General", "Forecasting", etc). As we want to predict each document's category, we design our model to predict the label field for each paper. After inspection of the other fields, we identify the title, abstract, and keyword as three fields that provide useful features for the model training.

In the initial processing steps, we convert the strings in the selected three fields into tokens and remove less relevant tokens such as those containing less than three characters and words found in our stop-word dictionary. We use lemmatization and stemming to convert the tokens into their base forms.

To convert the tokens into input for various machine learning models, we use TF-IDF as the feature extraction method.

For our training set, each entry's category labels are one-hot encoded into a vector representation. In the vector representation, categories associated with a particular entry are labeled as 1, and categories that are not represented are labeled as 0. For example, if a particular entry is classified under "General" and "Transmission", both of these labels have a value of 1 and all other category labels have a value of 0.

### B. Initial Training With Traditional Machine Learning Algorithms

Our aim is to carry out a multi-label text classification task. There are two general approaches to multi-label classification tasks, namely algorithm adaptation and problem transformation. For our algorithm adaptation approach, we look at classification models that can directly return multi-class labels for inputs. The Scikit-learn (sklearn) library provides such classifiers, including k-nearest neighbors algorithm, decision trees, and random forest. We use these three classifiers with our input data and achieve varying degrees of success.

For our problem transformation approach, we look at several binary classifiers to label our dataset by combining the binary classification outputs they produce. Essentially, we look at classifiers that predict each category individually and combine the outputs of those binary algorithms to represent a multi-label classification for each entry. Using models offered by the sklearn library, we experiment with bagging, gradient boosting, naïve Bayes, label power set, linear SVC, binary relevance and One-vs-Rest classifiers.

We experiment with these various traditional machine learning models to determine which model produces the most accurate result before any modification. We adopt the best-performing model for further modification to see if we can improve its results through the addition of a terminology dictionary.

### C. Terminology Extraction

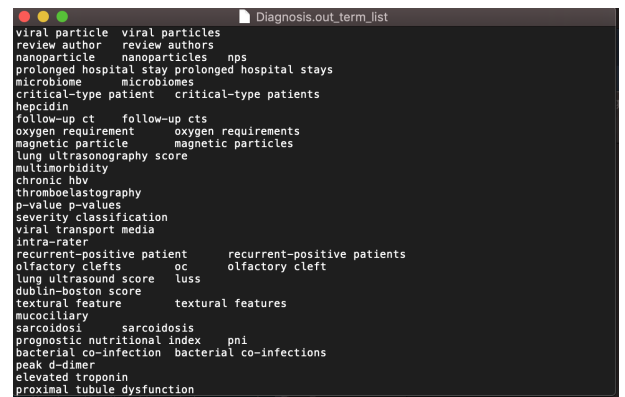
Given the academic characteristic of our dataset, we expect there to be many biomedical jargons and terminologies. As

some studies suggest that having a dedicated biomedical dictionary can assist with classification tasks for biomedical datasets [5], we build a tailored dictionary containing COVID-19 related terminologies and apply it to the model in order to modify and improve the classification algorithm.

For terminology extraction, we utilize an open-source tool for finding terminology in text called Termolator developed by Adam Meyers, Yifan He, Zachary Glass, and Shasha Liao [8]. The tool takes in two sets of documents. One set of documents acts as the foreground and the other acts as the background. Termolator extracts terminology that characterizes the foreground more than the background. As COVID-19 literature is a sub-field within the broader field of biomedical research data, Termolator, which relies on a subset-superset relationship between the foreground documents and background documents, would work well.

For our background documents, we use 500 PubMed paper abstracts from the NCBI Disease Corpus. The NCBI Disease Corpus acts as a superset of the foreground documents from the LitCovid dataset because the topic of the NCBI corpus is oriented more broadly towards medical papers while the foreground dataset focuses solely on COVID-19 data.

We use the LitCovid dataset as the source of our foreground documents and we construct a total of nine terminology dictionaries, each containing 10,000 words. Each dictionary corresponds to one of the nine categories we use to label our data. Thus, each dictionary contains terminology that characterizes certain academic features of the corresponding category. For example, for the "Diagnosis" category's dictionary, we collect words such as "follow-up CT", "prognostic nutritional index", and "thromboelastography" that are more closely related to diagnostic techniques. As another example, from "Epidemic Forecasting" report dictionary, we collect words like "fractional-order model", "disease-free equilibrium", and "controlled reproduction numbers", which are more relevant to model prediction and theoretical forecasting. We provide a sample terminology dictionary for the diagnosis category below.



```
viral particle  viral particles
review author  review authors
nanoparticle   nanoparticles  nps
prolonged hospital stay prolonged hospital stays
microbiome     microbiomes
critical-type patient  critical-type patients
hepcidin
follow-up ct    follow-up cts
oxygen requirement  oxygen requirements
magnetic particle  magnetic particles
lung ultrasonography score
multimorbidity
chronic hbv
thromboelastography
p-value p-values
severity classification
viral transport media
intra-rater
recurrent-positive patient  recurrent-positive patients
olfactory clefts            oc          olfactory cleft
lung ultrasound score      luss
dublin-boston score
textural feature            textural features
mucociliary
sarcoidosis                 sarcoidosis
prognostic nutritional index  pni
bacterial co-infection       bacterial co-infections
peak d-dimer
elevated troponin
proximal tubule dysfunction
```

Fig. 1. Diagnosis terminology dictionary.

We use the same background documents from the NCBI Disease Corpus in combination with our nine different foreground

document sets to generate the nine terminology dictionaries.

It is important to note that the number of documents in each category from the training set is not uniform, making our training set unbalanced. For this reason, instead of a random sample, we choose to use all documents from each category in the LitCovid dataset to act as the foreground for the corresponding terminology dictionary. The following table displays the distribution of the documents in the training set by category. “NaN” means there are no labels assigned to the document at all. Note that since one document can have multiple labels, the total here exceeds the total number of documents.

| Category         | Epidemic Forecasting | Transmission | General Info | Case Report | Mechanism | Diagnosis | NaN   | Treatment | Prevention |
|------------------|----------------------|--------------|--------------|-------------|-----------|-----------|-------|-----------|------------|
| No. of Documents | 687                  | 1479         | 1680         | 3484        | 6024      | 8854      | 11133 | 12096     | 18737      |

Fig. 2. Documents distribution with respect to categories.

#### D. NLP Modification/Correction with Terminology Dictionaries in SVC

After determining the best performing traditional machine learning classification model (i.e. SVC Square Hinge Loss), we use the above-mentioned nine terminology dictionaries to implement our novel modification.

Given a new input (i.e. a title and an abstract from the test set), we look at all nine dictionaries and generate a TF-IDF based similarity vector that indicates the relevance of the input to the dictionaries. We standardize the vectors and incorporate the results into the SVC classification process as follows.

We access the decision function of the SVC, which is a vector of nine entries that indicates the distance of the input to be classified to the decision boundaries (nine One-vs-Rest hyperplanes in the context of multi-class classification) and add the standardized similarity function by a factor of  $\alpha$ . Namely, the corrected decision vector becomes “original decision vector +  $\alpha$  \* standardized similarity vector”. This  $\alpha$  is a hyperparameter to be tuned and it helps us avoid influencing the initial decision function too much. We also experiment with  $\tanh(\text{similarity vector})$  to squeeze the entries of the vector between -1 and 1. The effect of this is discussed later.

As a result, when an input is very close to the decision boundary but situates incorrectly on the other side, we correct it by adding this standardized similarity vector. In theory, if one document is very “close” to, say, dictionary A, then the similarity at the corresponding bit in the standardized similarity vector is supposed to be very positive. If this document is very “far” from dictionary B, then the corresponding bit is supposed to be very negative. These numerical representations have corrective effects on the decision function and thus enhance the performance of the new model.

The overall process is illustrated by the flow chart in Figure 3 below.

## V. RESULT AND EVALUATION

### A. Model Evaluation

In a multi-label/multi-class classification setting, micro-averaging and macro-averaging are typically used as the

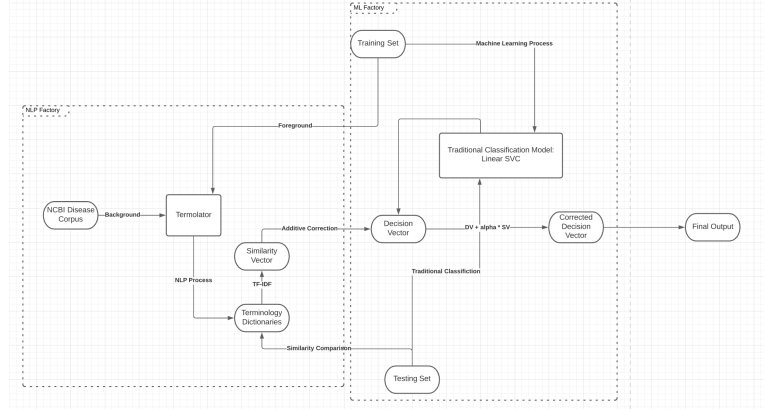


Fig. 3. Diagram of modified algorithm.

performance evaluation metrics. A macro-average will compute the metric independently for each class and then take the average. Hence, it treats all classes equally. However, a micro-average will aggregate the contributions of all classes to compute the average metric. In our scenario, we mainly focus on micro-averaging due to the unbalanced number of documents in each class (e.g. “Treatment” class has over 10,000 documents while “Epidemic Forecasting” only consists of approximately 700 documents).

Additionally, we use another indication flag – Hamming loss. In multi-class classification, Hamming loss corresponds to the Hamming distance between the ground truth and the predicted values. Namely, it is the fraction of labels that are incorrectly predicted (i.e. the fraction of wrong labels to the total number of labels).

### B. Model Comparison and Results

From our initial evaluation of unmodified traditional machine learning algorithms, we see that SVC Square Hinge Loss and Power Set SVC models give good overall results compared to the other models. Looking at the Micro F1 score, we see that all models except the k-nearest neighbors classifier give a score higher than 0.6. Nevertheless, similar to the test results we obtain through preliminary training using only the titles as features, the best-performing models when we consider the comprehensive scoring are SVC Square Hinge Loss and Power Set SVC. Models like random forest and naïve Bayes give relatively poor performance and this could be linked to the high correlation between the labels.

| Model              | Accuracy | Hamming Loss | Precision (Macro) | Precision (Micro) | Recall (Macro) | Recall (Micro) | F1 (Macro) | F1 (Micro) |
|--------------------|----------|--------------|-------------------|-------------------|----------------|----------------|------------|------------|
| KNN                | 0.2197   | 0.1871       | 0.7623            | 0.2957            | 0.1609         | 0.2012         | 0.1509     | 0.2395     |
| Decision Tree      | 0.4756   | 0.1142       | 0.5134            | 0.6124            | 0.4944         | 0.6004         | 0.5034     | 0.6063     |
| Random Forest      | 0.4515   | 0.0823       | 0.8252            | 0.8816            | 0.3003         | 0.5057         | 0.3801     | 0.6427     |
| Bagging            | 0.4961   | 0.0779       | 0.7546            | 0.7953            | 0.5271         | 0.6304         | 0.6149     | 0.7033     |
| Boosting           | 0.5192   | 0.0716       | 0.729             | 0.8439            | 0.5305         | 0.6267         | 0.627      | 0.7192     |
| Naive Bayes        | 0.4554   | 0.0858       | 0.5792            | 0.8298            | 0.2978         | 0.5208         | 0.3585     | 0.64       |
| SVC Sq. Hinge Loss | 0.6439   | 0.0578       | 0.8217            | 0.8281            | 0.6596         | 0.7639         | 0.7146     | 0.7947     |
| Power Set SVC      | 0.6939   | 0.059        | 0.7918            | 0.8158            | 0.6658         | 0.7711         | 0.712      | 0.7928     |

Fig. 4. Results of different models.

We identify SVC Square Hinge Loss as the best-performing traditional machine learning algorithm so we apply our terminology dictionary-based modification on the SVC Square Hinge

Loss model. In generating the modified vector, the parameters related to the correction process include the relative weight of the SVC model’s prediction, our dictionary similarity vector, the method we use to scale the dictionary similarity vector, as well as whether to ignore the value of the NaN entry in the dictionary similarity vector. By changing these parameters, we obtain performance scores for a variety of parameter settings.

| Modified | Alpha | Test Set Size | Scaling Method    | Omitting NaN | Accuracy | Hamming Loss | Precision (Macro) | Precision (Micro) | Recall (Macro) | Recall (Micro) | F1 (Macro) | F1 (Micro) |
|----------|-------|---------------|-------------------|--------------|----------|--------------|-------------------|-------------------|----------------|----------------|------------|------------|
| No       | N/A   | 500           | N/A               | No           | 0.648    | 0.0544       | 0.7231            | 0.8275            | 0.6556         | 0.7767         | 0.6967     | 0.8013     |
| Yes      | 0.2   | 500           | Scaling with Tanh | No           | 0.66     | 0.0518       | 0.7027            | 0.8124            | 0.6994         | 0.8239         | 0.7007     | 0.8181     |
| Yes      | 0.2   | 500           | Scaling with Tanh | Yes          | 0.658    | 0.0527       | 0.7023            | 0.8132            | 0.6918         | 0.8145         | 0.6967     | 0.8138     |
| Yes      | 0.1   | 500           | Scaling with Tanh | No           | 0.658    | 0.0518       | 0.7108            | 0.8183            | 0.6881         | 0.8146         | 0.6985     | 0.8104     |
| Yes      | 0.1   | 500           | Scaling with Tanh | Yes          | 0.654    | 0.0527       | 0.7099            | 0.8182            | 0.6818         | 0.8066         | 0.6948     | 0.8114     |
| Yes      | 0.25  | 500           | Scaling with Tanh | No           | 0.654    | 0.0527       | 0.6941            | 0.8055            | 0.7023         | 0.827          | 0.6983     | 0.8101     |
| Yes      | 0.25  | 500           | Scaling with Tanh | Yes          | 0.654    | 0.0531       | 0.6954            | 0.8057            | 0.6977         | 0.8176         | 0.6952     | 0.8131     |
| Yes      | 0.2   | 200           | Default Scaling   | No           | 0.626    | 0.0558       | 0.7135            | 0.773             | 0.8529         | 0.8509         | 0.7704     | 0.8128     |
| Yes      | 0.2   | 200           | Default Scaling   | Yes          | 0.648    | 0.0544       | 0.7227            | 0.7896            | 0.8377         | 0.8381         | 0.7586     | 0.8131     |
| No       | N/A   | Full Test Set | N/A               | No           | 0.639    | 0.0578       | 0.8277            | 0.8281            | 0.6596         | 0.7839         | 0.7146     | 0.7947     |
| Yes      | 0.2   | Full Test Set | Default Scaling   | No           | 0.6274   | 0.0587       | 0.7647            | 0.7892            | 0.7381         | 0.8177         | 0.7371     | 0.8032     |
| Yes      | 0.2   | Full Test Set | Default Scaling   | Yes          | 0.64     | 0.0581       | 0.7722            | 0.7988            | 0.7342         | 0.805          | 0.7354     | 0.8024     |
| Yes      | 0.2   | Full Test Set | Scaling with Tanh | No           | 0.6886   | 0.0579       | 0.7857            | 0.8043            | 0.7019         | 0.7993         | 0.7272     | 0.8018     |
| Yes      | 0.2   | Full Test Set | Scaling with Tanh | Yes          | 0.6431   | 0.0577       | 0.7992            | 0.8091            | 0.6983         | 0.7927         | 0.7259     | 0.8008     |

Fig. 5. Results of different parameter combinations.

To determine an appropriate value of  $\alpha$ , which determines the relative weight between the model-generated decisions and the dictionary similarity vector, we use multiple values of  $\alpha$  on a test set of size 500 and obtain varying performances. Some values of  $\alpha$ , such as 0.5 and 1, obtain significantly worse performance and are omitted in the graph. We observe that performance of models with  $\alpha = 0.2$  is comparable to that of models with other  $\alpha$  values in all individual metrics and is slightly better in Micro F1 score. As such, we set  $\alpha$  as 0.2.

We run the model on the entire test set with varying scaling methods as well as ways to handle the NaN category in the dictionary similarity vector. The default scaling provided by preprocessing.scale() function may generate values larger than 1 or smaller than -1 so we experiment using the tanh function to transform the scaled vectors such that all values are kept between -1 and 1. Moreover, we experiment with ignoring the NaN category in the dictionary similarity vector because preliminary observations show that the value of NaN is often disproportionately large in the generated dictionary similarity vectors.

We observe that the models with terminology dictionary correction obtain better Micro F1 scores on the entire test set compared to the original model. With regard to the NaN entries, models that ignore them perform better in accuracy, Hamming loss, and precision, but worse in recall and F1 score. Similarly, the models using tanh function perform better in some measurements but have slightly worse performance in terms of F1 scores.

A model may have better performance in certain aspects but worse performance in others when compared to another model because the measurements consider different things. For example, precision and recall need an output vector to be the same as the actual vector to be counted as right, while the Hamming loss looks at how similar output vectors are to actual vectors by comparing bit by bit. Micro F1 score is relatively more important in our evaluation process and by this measurement, our new model that incorporates terminology dictionary correction performs better than the original SVC Square Hinge Loss model.

To further study the performance of our current model and explore possibilities for future improvement, we generate

confusion matrices for each of the nine labels. Each confusion matrix shows the performance of our model with respect to a particular label. For example, the confusion matrix of label “Prevention” shows the number of true positive, false positive, true negative, and false negative cases generated by our model with respect to the “Prevention” category.

We notice that while most negatives can be correctly classified as negatives, the model performs less ideally when predicting positives. For example, 25 out of 36 documents with the label “General Info” are predicted as not having the label “General Info”. This could be because documents classified as “General Info” do not have many characteristic features and vocabulary, and the solutions to these classification errors can be a direction for our future improvement.

Furthermore, as noted previously, our training set is heavily unbalanced. Thus, categories with larger datasets (e.g. “Prevention”) perform much better than categories with smaller datasets (e.g. “General Info”).

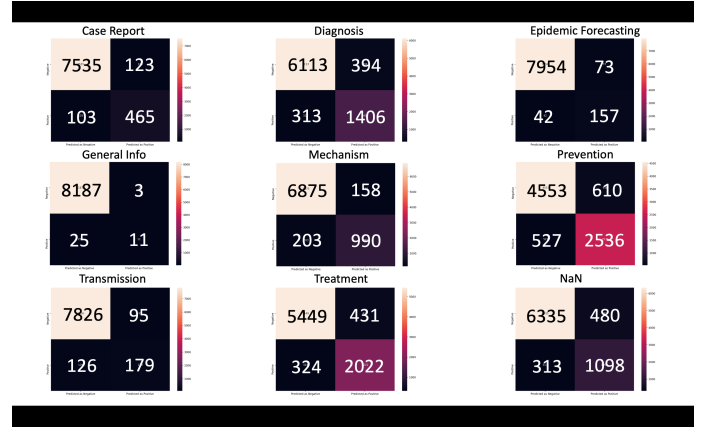


Fig. 6. Confusion matrices of nine labels (top left: true negative; top right: false positive; bottom left: false negative; bottom right: true positive).

In terms of comparing our results to the state-of-the-art, we achieve an F1 score of 80.32 compared with BioBERT’s 86.2 [5]. Considering the significant sophistication of BioBERT, especially compared to our much simpler model, our model provides reasonable results.

## VI. FUTURE WORK

There are certain limitations and potential extensions of our model. When utilizing the terminology dictionaries, we have not explored their full capacity. For example, Termolator is capable of ranking the characteristics of terminology which means the words that appear earlier in the dictionary are more representative of the corresponding category. This information can be used to assign weighted relevance to terminology so for different documents that are compared against the dictionary, we could generate more targeted and reliable similarity vectors. In our current approach, we simply use the dictionaries as a whole set and generate TF-IDF based similarity vectors.

Another potential extension with terminology dictionaries would be a different way of incorporating the similarity vector.

For now, we only experiment with two ways: weighted and scaled addition of the similarity vector. There might be a better strategy to experiment with correction methods.

## VII. CONCLUSION

From the result of different models, it is clear to see that we have improved the SVC Square Hinge Loss model by generating terminology dictionaries and combining the system predictions with the dictionary similarity vectors we generate for each data point in the test set. Considering the Micro F1 score, which reflects the performance of a multi-label classification task, we boost the F1 score to 0.8032 which is an improvement from the already sophisticated classification model.

## REFERENCES

- [1] Q. Chen, A. Allot, and Z. Lu, "Litcovid: An open database of covid-19 literature," *Nucleic Acids Research*, 2020.
- [2] Q. Chen, A. Allot, and Z. Lu, "Keep up with the latest coronavirus research," *Nature*, vol. 579, no. 7798, p. 193, 2020, ISSN: 1476-4687 (Electronic) 0028-0836 (Linking). DOI: 10.1038/d41586-020-00694-1. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pubmed/32157233>.
- [3] N. Colic, L. Furrer, and F. Rinaldi, "Annotating the pandemic: Named entity recognition and normalisation in COVID-19 literature," in *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online: Association for Computational Linguistics, Dec. 2020. DOI: 10.18653/v1/2020.nlpCOVID19-2.27. [Online]. Available: <https://www.aclweb.org/anthology/2020.nlpCOVID19-2.27>.
- [4] P. Lewis, M. Ott, J. Du, and V. Stoyanov, "Pretrained language models for biomedical and clinical tasks: Understanding and extending the state-of-the-art," in *Proceedings of the 3rd Clinical Natural Language Processing Workshop*, Online: Association for Computational Linguistics, Nov. 2020, pp. 146–157. DOI: 10.18653/v1/2020.clinicalNLP-1.17. [Online]. Available: <https://www.aclweb.org/anthology/2020.clinicalNLP-1.17>.
- [5] B. Jimenez Gutierrez, J. Zeng, D. Zhang, P. Zhang, and Y. Su, "Document classification for COVID-19 literature," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 3715–3722. DOI: 10.18653/v1/2020.findings-emnlp.332. [Online]. Available: <https://www.aclweb.org/anthology/2020.findings-emnlp.332>.
- [6] M. Ostendorff, T. Ruas, T. Blume, B. Gipp, and G. Rehm, "Aspect-based document similarity for research papers," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 6194–6206. DOI: 10.18653/v1/2020.coling-main.545. [Online]. Available: <https://www.aclweb.org/anthology/2020.coling-main.545>.
- [7] R. I. Doğan, R. Leaman, and Z. Lu, "Special report: Ncbi disease corpus: A resource for disease name recognition and concept normalization," *J. of Biomedical Informatics*, vol. 47, pp. 1–10, Feb. 2014, ISSN: 1532-0464.
- [8] P. Mayr, C. Zhang, A. Meyers, Y. He, Z. Glass, J. Ortega, S. Liao, A. Grieve-Smith, R. Grishman, and O. Babko-Malaya, "The termolator: Terminology recognition based on chunking, statistical and search-based scores," *Frontiers in Research Metrics and Analytics*, vol. 3, Jun. 2018. DOI: 10.3389/frma.2018.00019.