Vocabulary Matters: An Annotation Pipeline and Two Deep Learning Algorithms for Enzyme Named Entity Recognition

Meiqi Wang¹, Avish Vijayaraghavan¹-², Tim Beck³,⁴, and Joram M. Posma¹,⁴, a

¹Section of Bioinformatics, Division of Systems Medicine, Department of Metabolism, Digestion and Reproduction, Imperial College London, W12 0NN, United Kingdom
²UKRI Centre for Doctoral Training in AI for Healthcare, Imperial College London, United Kingdom
³Department of Genetics and Genome Biology, University of Leicester, LE1 7RH, United Kingdom
⁴Health Data Research (HDR) UK, United Kingdom
⁵These authors contributed equally.

Abstract
Enzymes are indispensable substances in many biological processes. With biomedical literature growing exponentially, it becomes more difficult to review the literature effectively. Hence, text-mining techniques are needed to facilitate and speed up literature review. The aims of this study are to create a corpus with annotated enzymes to train and evaluate enzyme named-entity recognition (NER) models. A novel pipeline was built using a combination of dictionary matching and rule-based keyword searching to automatically annotate enzyme entities in over 4,800 biomedical full texts. Two Bidirectional Long Short-Term Memory (BiLSTM) networks using BioBERT and SciBERT as tokeniser and word embedding layers were trained on this corpus and evaluated on a manually annotated test set of 526 full-text publications. The dictionary- and rule-based annotation pipeline achieved an F1-score of 0.863 (precision 0.996, recall 0.762). The SciBERT-BiLSTM model (F1-score 0.965, precision 0.981, recall 0.954) largely outperformed the BioBERT-BiLSTM model (F1-score 0.955, precision 0.981, recall 0.937). This study contributed a novel dictionary- and rule-based automatic pipeline with almost perfect precision which runs in a matter of seconds on a standard laptop. Both deep learning (DL) models achieved state-of-the-art performance (F1 > 0.95) for enzyme NER, with the SciBERT-based model outperforming the BioBERT-based model in terms of recall, demonstrating the vocabulary used by models matters. The proposed pipeline with the DL models can facilitate more effective enzyme text-mining and information extraction research for literature review and are the first algorithms specifically for enzyme NER.

Availability: All codes are available for automatic annotation and model training (including data), with instructions on how to deploy the model on new text, from https://github.com/omicsNLP/enzymeNER.

Introduction
Enzymes are proteins that regulate biochemical reactions by decreasing the activation energy required for the chemical conversion from a substrate into a product to occur. The largest online enzyme database, BRENDA (1), contains over 100K different enzyme names that are obtained by manual extraction from literature, semi-automatic text mining or by integration from other sources and is growing exponentially (2). Manual extraction of information from large numbers of publications is time-consuming, labour-intensive and may lead to human errors. With the biomedical literature expanding at an increased rate (3), there is a clear need to develop new methods to automatically identify and extract enzyme names from the biomedical scientific literature.

Natural language processing (NLP) is a sub-field of artificial intelligence that uses statistics-based techniques to understand human language, and has multiple different applications such as text prediction, classification, summarisation and translation, as well as speech recognition (4). Recent advances in the field have resulted in obtaining high performance of various NLP tasks with deep learning (DL) models and domain-specific word embeddings. Named-entity recognition (NER) is an NLP task that locates and identifies specific entities in texts and classifies them into predefined categories. In recent years, NER models using DL models have flourished given their high performance on many tasks (5), using DL architectures such as bidirectional long-short term memory (BiLSTM) based recurrent neural networks (6, 7) and transformers (8–11). There has been much recent success of biomedical natural language processing (bioNLP) such as for gene/protein, disease and chemical entity recognition with fine-tuned pre-trained language models (12–15), as well as for reading clinical notes in electronic health records (16). The concept of pre-training refers to initialising network weights based on a large-scale corpus (a collection of texts/sentences). This is done to generate distributed representations of words, or N-dimensional vectors, as inputs for subsequent NER models. In NLP, this processing is known as word embedding which is a type of pre-trained language model. The word embedding process can not only machine-understandable expressions of real words, but also
capture semantic and syntactic properties of a word which might not be explicitly present without word embedding (5), and this is context- and domain-dependent. Until a few years ago, static word embedding such as Word2Vec (17) and GloVe (18) were predominantly used. These have now been replaced by contextualised word embeddings that enhance the DL model performance (19) by focusing on neighbouring words. These pre-trained models can then be fine-tuned (changing network weights) for different downstream NLP tasks using transfer learning (5). The choice of pre-trained model has a big influence on the final performance because general domain datasets, such as those to train Bidirectional Encoder Representations from Transformers (BERT) (9) (trained on Wikipedia and Google Books corpus), may not contain the vocabulary and word embeddings that are necessary to process text from biomedical literature (19).

BioBERT is a biomedical language representation model that was pre-trained on biomedical domain corpora (PubMed abstracts with 4.5B words and PubMed Central (PMC) full-text publications with 13.5B words) after initialisation with BERT-weights and can be fine-tuned for NER, relation extraction and question answering text-mining problems and has shown state-of-the-art performance for several NER tasks (12). SciBERT (11) is a related model which was initialised from scratch and trained on full texts publications from the Semantic Scholar platform (20), and thus does not inherit any vocabulary from BERT making it potentially more useful for bioNLP tasks as it was trained on scientific publications from the biomedical domain (82%) and computer science (18%). Training of models for biomedical NER relies on the availability of biomedical text corpora in machine-readable format (21). Most corpora used for training (and evaluating) bioNLP algorithms are those either curated by the BioCreative committee or NCBI team and contain annotations for different entities including for diseases (22, 23), genes/proteins (24–26), mutations (27) and chemical compounds (23, 28, 29). The availability of corpora dictates the developments made in the field of bioNLP, with mostly incremental gains over prior methods. For the task of enzyme NER no corpus exists that is specific for this task, thus only corpora with annotated protein entities can be used. However the specificity is a problem since not all proteins are enzymes, thereby training on protein corpora for enzyme recognition will inflate false positives. Moreover, the existing corpora with protein (and other) annotations come mostly from abstracts, as can be seen from data used by CollaboNet (30). In this article we aim to fill this gap by describing our dictionary-based annotation workflow to create an annotated enzyme corpus in machine-readable format, curated from full-text publications. We then describe two DL-based NER models trained using different word embeddings and evaluate these on a manually annotated test set.

Materials and Methods

Data - Publications. The corpus was created from PMC Open Access publications from four different omics domains: genome-wide association studies (GWAS), proteomics, metabolomics and microbiome. The GWAS and metabolomics publications that were used we described previously (31, 32), the proteomics and microbiome publications were acquired in a similar manner as the metabolomics publication (32). For proteomics the field keyword search terms were ‘proteomics’, ‘proteome’, ‘proteomic’ and ‘proteome-wide’, the sample keyword search terms were ‘urine’, ‘urinary’, ‘blood’, ‘serum’, ‘plasma’, ‘faecal’, ‘faeces’, ‘fecal’, ‘feces’, ‘stool’, ‘cerebrospinal fluid’, ‘CSF’ and ‘biofluid’, with similar disease search terms. For the microbiome corpus OA publications relating to airway, faecal, skin, urinary and vaginal microbiomes were searched for with additional keywords such as ‘16S’, ‘rRNA’, ‘sequencing’, ‘shotgun’, ‘Illumina’, ‘MiSeq’ and ‘PCR’.

The publications were converted from HTML, standardised and converted to the machine-readable BioC-JSON format using Auto-CORPus (31). First, Auto-CORPus converts the main text of each publication from HTML to BioC JSON format. In this file, the pipeline splits the publications into different sections using the Information Artifact Ontology (IAO) (33) and in each section, each paragraph forms a single ‘text’-feature. Second, Auto-CORPus transforms tables inside publications to a table-JSON format, and it extracts abbreviations from both the main text and from separate abbreviations sections (if available) within the main text and generates another JSON file for abbreviations with linked full definitions. Here, we use the main text (BioC-JSON) and abbreviation JSON files for pre-processing and to build the datasets for DL models.

Data - Enzyme Nomenclature. A list of enzymes codes with names and synonyms was downloaded on 28/10/2021 using the KEGG REST API (https://www.kegg.jp/kegg/rest/keggapi.html). This list contains over 28K enzyme names/synonyms that were used for the dictionary matching here. Several changes were made to this dictionary to improve the matching step later as the matching list in this study. Missing entities were added that have the same meaning but different spelling styles compared with other items within the list. For example, some enzymes connect different words using hyphens while others use spaces instead, whereas others use Greek letters opposed to spelling these out using the Latin alphabet. Therefore, additional entries were created in the dictionary to account for these differences and capture more entities when matching the texts.

Annotation Pipeline. The second improvement made relates to speeding up the searching process. We reconstruct the dictionary list in a novel hierarchical format by clustering entities with the same features into smaller streams based on enzyme classes. We performed this based on analysis of the enzyme nomenclature, where most (but not all) enzyme names end in ‘ase’. We use the seven categories of enzymes based on the different chemical reactions they are involved in. For example, enzymes involved in oxidation reactions are likely ending in ‘oxidase’ whereas ‘reductase’ enzymes perform reduction reactions. Using this structure, we classify the items within the list in a tree-structured (hierarchical) dictionary, in that way, when searching through the
text, the pipeline can use regular expression (RegEx) rules to search the node-words from the root of the word to determine which branch-list to use to speed up the searching and auto-annotation process. Supervised NER algorithms rely on annotated corpora, however none exist for enzymes. We created an enzyme-specific corpus using a fast, automated pipeline for annotation using the hierarchical dictionary of enzyme terms. The pipeline uses rule-based RegEx to assist with the search process of potential entities in full-text articles. The BioC JSON files from Auto-CORPus were used as input to the annotation pipeline, with the enzyme entities in each paragraph added to the ‘annotations’ section of the JSON document which includes a unique id, an external identifier (EC number) if a direct match is found (if no direct match is found this is replaced by the root word) and the location in the text (based on the offset of the paragraph in the entire article). The pipeline consists of two search steps to find entities. First, direct matches of dictionary terms in the text (exact match to a known entity). Second, using keyword searching using root enzyme terms (e.g. ‘transferase’) to perform rule-based fuzzy matching to identify entities that do not directly match to elements from the dictionary. Specifically, the pipeline utilizes the spaCy package (v3.2) to split each paragraph into individual sentences, in which each word is contrasted with patterns created by the searching rules based on the structure of enzyme dictionary. If a (root) word is matched, the pipeline will then first search throughout the given node in the list for a content-matched item (first step). In this step, the greedy algorithm was used to ensure the matched entity is the one with the maximum length in this sentence. For example, the term ‘aliphatic alcohol dehydrogenase’ also contains other matches to terms ‘alcohol dehydrogenase’ and ‘dehydrogenase’, and this approach ensures any terms contained within another are not selected. If no direct match is found, the pipeline continues with step 2. The principle of partial searching is that if a word is regarded as a part of an enzyme name, such as a word that is matched with a node-word in the dictionary and words ending with ‘ase’ are considered as a part of a potential enzyme. The next steps considers the words preceding and following the identified entity and evaluates if these are part of the enzyme entity. In this step common words ending in ‘-ase’, e.g., ‘phase’ and ‘database’, are ignored. To establish the number of words before and after an entity to consider, we analysed the known enzyme names in the dictionary. Over 92% enzyme entities in the list end in ‘-ase’, and from these over 99% consist of five words or less, indicating that we can restrict our search to four words before the identified ‘ase’-word. For words after the entity we use additional RegEx rules as most of the contents after the ‘ase’-word are enclosed in brackets, but cases exist where there is a single number or Greek letter. Therefore, we created rules based on these features to support backward searching and adding these to the identified entity. To increase the number of training cases, we also include sentences in which abbreviations of enzymes appear. We applied the pipeline to the list of definitions (abbreviations output JSON file from Auto-CORPus) in the article, and the abbreviations of any enzymes identified are then searched in the main text. These abbreviations are annotated in the same way as full terms, however for training these are replaced by their full definitions. Abbreviations are not used by themselves as this can increase the number of false positives as some abbreviations can have different definitions in different articles.

**Training, Validation, Test Splits.** The corpus of 3,525 full-text publications that contained any enzyme mention was split 75:10:15 into training/validation/test sets with each portion containing an equal proportion of GWAS, proteomics, metabolomics and microbiome articles (Table 1). The training and validation sets were used to construct the deep learning models, with the test set separately undergoing manual annotation and used to evaluate the performance of the deep learning algorithms. The training and validation sets were annotated using the annotation pipeline. Only the sentences containing enzyme entities constitute the dataset for training models. Therefore, we extract those annotated sentences to build up the final corpus for DL models. Each sentence gets a unique ID which consists of the PMCID (i.e., paper ID with PMC identifier), a sentenceID that determines which section (e.g., Results) and paragraph the sentence belongs to. Unlike our metabolite NER model (32) where abbreviations of metabolites were kept in, here we replaced all abbreviations in the full text with the definitions from the abbreviation output files. To increase the number of annotations for training, we consider not only the textual abstract, methods, results, and discussion sections (as done for metabolite NER), but we also include the introduction and references sections as supplement. The test set consists of 526 full-text publications. These were annotated in a three-stage process. First, the articles were annotated using the automatic annotation pipeline (see below). Second, two annotators independently annotated the 526 articles using TeamTat (34), resolved incorrect annotations from the pipeline, added missing concept identifiers (EC numbers) and annotated missing entities. The final step was done in collaborative mode in TeamTat where all annotators can see the annotations made by both annotators in the independent mode. TeamTat indicates any conflicts between annotators that need to be resolved by the annotators (and mediated by a third annotator in case of disagreements). Conflicts include cases where only one annotator has annotated an entity, where annotated entities are overlapping and where the

<table>
<thead>
<tr>
<th>Domain</th>
<th>Pubs (n)</th>
<th>Train (n)</th>
<th>Val (n)</th>
<th>Test (n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>GWAS</td>
<td>884</td>
<td>664</td>
<td>88</td>
<td>132</td>
</tr>
<tr>
<td>proteomics</td>
<td>446</td>
<td>336</td>
<td>44</td>
<td>66</td>
</tr>
<tr>
<td>metabolomics</td>
<td>1,006</td>
<td>756</td>
<td>100</td>
<td>150</td>
</tr>
<tr>
<td>microbiome</td>
<td>1,189</td>
<td>893</td>
<td>118</td>
<td>178</td>
</tr>
</tbody>
</table>
concept ID is different between annotators.

**Deep Learning Models.** In this study, the BiLSTM network structure is used because of its state-of-the-art performance achieved in NER tasks in the chemical domain. ChemListem (35), a type of BiLSTM model, outperformed other methods for chemical entities recognition and was validated by the CEMP BioCreative V.5 challenge (36), and MetaboListem and TAbOliSTM (32) achieved similar performance for metabolite named entity recognition. To further optimise ChemListem, the ability to accurately capture the features of enzyme entities can be enhanced by replacing GloVe embedding method with other word embedding models which have displayed accurate results in biomedical text mining. Here, we use two different word embeddings which are BioBERT (12) and SciBERT (11), and then separately combine the BiLSTM architecture from ChemListem to improve the performance of the enzyme NER task. As our previous study (32) demonstrated, BioBERT embedding outperforms BERT, which is reasonable considering that BioBERT was pre-trained on a large sum of publications from biomedical domain. Therefore, we use BioBERT instead of BERT as word embedding to compare the performance with SciBERT. All these models are trained on the dataset we created with the annotation pipeline and are evaluated in the gold-standard test set (i.e., manually annotated dataset).

**Pre-processing, Tokenisation and Word Embedding.** Prior work used a Python translation of the Open-Source Chemistry Analysis Routines (OSCAR) software (37) as tokeniser (32, 35). In this study, BioBERT (12) (biobert-base-cased-v1.2) and SciBERT (11) (scibert_scivocab_cased) are employed as transformer-based word tokenisers to generate the vocabulary and corresponding word embedding architectures used in the different NER models.

The SOBIE system is applied to mark each token as a specific tag, in which ‘O’ (for outside) labels a token that is not part of an entity, ‘S’ refers to singleton that marks a one-word token (i.e., the whole of an entity), ‘B’ marks the beginning word of a multi-word token, ‘I’ marks one inside an entity, and ‘E’ marks one at the end. A pre-classifier subsystem was implemented as in prior work (32, 35) using a random forest. The pre-classifier works by generating name-internal binary features that are either ‘O’ or ‘SBE’ labels and assign a probability to each token for it to be part of an enzyme name (S, B, I or E). These tokens are segmented by BioBERT or SciBERT, respectively, to match the corresponding models. Afterwards, these pre-trained features are passed to the BiLSTM network.

**BiLSTM Network.** The BiLSTM model contains two inputs that are concatenated: the first input layer is a Conv1D with a dropout of 0.5 to process the name-internal features from the pre-classifier, this is a convolutional layer with 256 neural dimensions. The second input are the features from the word embedding layer (here BioBERT and SciBERT are separately applied to this layer) where text tokens are converted to a sequence of integers. These contextual embeddings were initialised by pre-trained weights of the transformer models with a 768 hidden dimensional size, feeding the output from the embedding model through a SpatialDropout1D layer with a dropout rate of 0.1 to prevent overfitting. Both input layers are concatenated to result in a 1,024 dimensional vector (256+768) as final input into the BiLSTM network that contains a 64-dimensional output space. The activation function selection is with a tanh (hyperbolic tangent), with a sigmoid activation function applied to the recurrent step of the network. The regularisation penalty used for the kernel weights matrix was 11_12 in which both the 11 and 12 loss are limited up to 1 × 10^-6. The output layer is a time-distributed dense layer with a softmax activation function to ensure the sum of the 5 output probabilities (‘O’, ‘S’, ‘B’, ‘I’, and ‘E’) for each token totals to 1.

Models were trained for 10 epochs with each epoch trained in mini batches on a workstation with an Intel Xeon W-2125 CPU with 128GB RAM (4x32GB) and an NVIDIA RTX5000 GPU with 16GB memory and 3,072 CUDA parallel-processing cores. The model with the highest F1 score (see below) for the validation dataset is stored and used to evaluate on the manually annotated test set.

**Comparison of our model on protein NER using BERN2.** We compare our enzyme NER results with the BERN2 multi-task model (15), which is based on BioBERT, for the gene/protein entity tag as no enzyme-specific NER model currently exists. The BERN2 model can be accessed remotely through a web API (http://bern2.korea.ac.kr/) but this is limited to 300 requests per 100 seconds per user. For a quicker turnaround time on our large corpus, we followed the authors’ guidelines in the BERN2 repository (https://github.com/dmis-lab/BERN2) to install and run a local version of the model on a Linux workstation with an Intel Core i7 CPU (10700K, 3.8GHz, 8 Core) with 32GB RAM and NVIDIA GeForce RTX 3080 GPU with 10GB memory. Some slight modifications were made to set up the BERN2 model on a Linux workstation with a GPU. In the “run_bern2.sh” script, we changed “python” to “python3”.

Every time it runs, BERN2 creates a “log/” directory to record run details which can help identify the location of a problem. From this, we found an issue with the installation of GNormPlusJava (the gene/protein normalisation tool) which required us to re-download an alternate version of the CRF++ tool. BERN2 may try to access a port that is already being used - if the process using the port is not important, it can be killed using the following command fuser -k [port number]/tcp. If BERN2 is not running properly, it can be restarted cleanly by: deleting the “log/” folder, stopping BERN2 using the appropriate script, and deleting any CUDA artefacts.

Once the BERN2 model is running on a local port within its conda environment, we make requests to it using the URL associated with the port. Running our corpus through the model returns a set of found entities per paper (and optionally per paragraph) which fall into one of nine biomedical classes: gene/protein, disease, drug/chemical, species, mutation, cell line, cell type, DNA, and RNA. We filter entities for...
the gene/protein class to enable protein NER comparison to our model.

Applicability of our model to improve metabolite NER. The metabolite NER (32) and enzyme NER tasks were carried out independently, and hence use different training, validation and test sets. To compare the output of both algorithms we evaluated the overlap of the two test sets, yielding a total of 18 metabolomics publications that have not been used as part of the training or validation stages in either algorithm. We extracted all sentences with manually annotated metabolites and/or enzymes and compare the outputs of both NER algorithms to evaluate whether our model can help to improve the accuracy of the metabolite NER output.

Evaluation Metrics. To evaluate the performance of the annotation pipeline and DL models, we calculate 3 properties of a confusion matrix: true positive (TP) entities, false positives (FPs) and false negatives (TNs). FPs are entities predicted to be an enzyme by a model which does not match an entity in the test set. FNs are entities that were not annotated by the model but are contained in the test set annotations. Using these measures, we calculate the precision (probability of correctly predicted positive values in the total predicted positive outcomes, Equation 1), recall (also known as sensitivity, probability of correctly predicted positives relative to all true positive cases, Equation 2) and F1-score (weighted average of precision and recall, Equation 3).

\[
\text{precision} = \frac{TP}{TP + FP} \tag{1}
\]

\[
\text{recall} = \frac{TP}{TP + FN} \tag{2}
\]

\[
F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{3}
\]

Partial positive (PP) matches, i.e. where the only root term is correctly identified or where the indication of an isoform is missing but the remainder of an entity is correctly annotated, were separately tracked and the final count of PPs was split equally between TP and FN.

Results and Discussion

Corpus creation and hierarchical dictionary. The reconstructed dictionary list, using the hierarchical format (Figure 1), speeds up the searching process compared to a standard dictionary search. An entire full-text article can be annotated in 0.5s using this approach, compared to 1s per file using the standard approach, due to the reduction in size of the branch list resulting in less evaluations.

Our annotation pipeline was used to create an annotated corpus for supervised learning. The enzyme entities in the texts are extracted and coded into the “annotations” section of the BioC JSON file (output of Auto-CORPUs) that follows the text content of each paragraph, see example in Figure 2. Each annotation has a number of key-value pairs: ‘text’ is the name of the identified entity, ‘infons’ contains the information on the identifier (E.C. number, or keyword), type (enzyme), annotator, annotation time, ‘id’ is a unique ID for each annotation in the text and indicates the order in which the entity is annotated, and ‘locations’ indicates the position of the annotation within the full-text.

The annotated corpus is easy to use, and can be reused in future to convert the text and annotations to different formats to be used for training different algorithms. Moreover, the hierarchical annotation approach may also be applicable for speeding up the process of annotating other biomedical entities such as pathways, gene names or metabolites. For example, pathways also have a similar nomenclature as enzymes.

![Fig. 1. Example of the hierarchical structure of the dictionary showing a part of the ‘-ase’ branch. Enzymes not ending in ‘-ase’ are contained in the dictionary in other branches.](image1)

```
"annotations": [
    {
        "text": "isocitrate lyase",
        "infons": {
            "identifier": "ec:4.1.3.1",
            "type": "enzyme",
            "annotator": "m.wang21@imperial.ac.uk",
            "updated_at": "2022-01-30T00:04:38Z"
        },
        "id": 19,
        "locations": [
            {
                "length": 16,
                "offset": 65685
            }
        ]
    }
]
```

![Fig. 2. An example of annotations. That is an individual dictionary that displays the detail of an entity, which has four elements (i.e., ‘text’, ‘infons’, ‘id’, ‘locations’) to represent a unique entity in the text.](image2)
in which entities end in similar ways such as the root-words ‘synthesis’, ‘biosynthesis’, ‘degradation’ and ‘metabolism’. The JSON structure also lends itself to contain multiple entity types which means multi-task learning corpora can be prepared for biomedical NER tasks. However, our pipeline includes some modifications to improve the accuracy and effectiveness of annotation. The weakness of the annotation pipeline mainly lies in the second searching step (i.e., keyword searching). Once a root word has been identified, the preceding and following words are evaluated whether these are part of the enzyme. The dictionary contains several spellings of an enzyme, however different forms may still exist that may include (or exclude) the use of hyphens which will prevent terms to be matched directly. Likewise, different isofoms may be indicated by added numbers or letters at the end. Our annotation pipeline uses a rule-based regular expression search to evaluate if the next word may indicate an isoform, however this approach is not foolproof. Finally, another difficulty is that ambiguous semantics and multiple expressions of enzymes are unfavourable for annotators to precisely distinguish an enzyme name within the sentence. For example, some descriptive modifiers such as ‘human’, ‘serum’, or ‘bacterial’ are truly parts of an enzyme name, like ‘human carboxylesterase 2’ (EC:3.3.1.84), but more generally, these words should be simply regarded as the adjectives of enzymes, such as ‘human proteases’ and are therefore not included in our annotations unless for some cases where there are exact matches. The difficulty of manual annotation highlights how this process may be replaced by an algorithm before being subjected to human verification.

Our training data set created from the annotated JSON files additionally replaces all abbreviations of enzymes by their definitions, hence the evaluation of abbreviated entities relies on correctly identifying abbreviation-definition pairs. Our annotation pipeline annotated over 30,000 entities in over 3,000 articles (Table 2). As expected, the proteomics articles have the largest average number of entities (23) followed by metabolomics (9) and GWAS (8). Across all annotated entities, the most common single word annotation is polymerase, whereas when including multi-word entities the most common root terms are kinase and protease (Table 3). No enzyme-specific corpus exists, and our corpus is also larger than most bioNLP corpora (38).

**Model Evaluation.** As described in the results, we tracked all partial matches and split these between TPs and FNs. We also analysed the list of PP items. Some of these PPs are similar (but not identical) between the manually annotated test set and the annotation (or DL) pipeline. The main differences related to a capital letter or a number not captured by the annotation pipeline (i.e. isofoms). Most other differences relate to the method only identifying the root term (‘kinase’ instead of ‘Ser/Thr kinase’). The former being a minor mistake (could be considered a TP), whereas the latter clearly misses an important part of the enzyme (and perhaps should be a FN). Resolving PPs into TPs/FNs would involve a fourth independent person (from the annotators and arbiter) to go over each of these entries which was not feasible given the time/effort required. Therefore here we regarded half of the PPs as TP and the other as FN.

The precision of the pipeline is effectively 1.00 (see Table 4), and there is only one false positive (‘protein A’). However, the lower recall of 0.69 demonstrates that there are certain enzyme entities that were not found by the annotation pipeline (higher degree of FNs). Most of these missed entities are abbreviations (with added characters or incomplete matches) which is hard to identify without the understanding of the context in which entities such as ‘CN-II’ or ‘dN-1’ first appear, for this reason we separately evaluated the results in Table 4 while omitting abbreviations from the calculations, which resulted in a recall of 0.76. A logical next step is to further improve abbreviation detection algorithms (currently using rule-based methods) by taking a DL approach to focus on relation extraction between abbreviations and definitions.

Another aspect we identified is that punctuation and hyphens on relation extraction between abbreviations and definitions. The precision of the pipeline is effectively 1.00 (see Table 4), and there is only one false positive (‘protein A’). However, the lower recall of 0.69 demonstrates that there are certain enzyme entities that were not found by the annotation pipeline (higher degree of FNs). Most of these missed entities are abbreviations (with added characters or incomplete matches) which is hard to identify without the understanding of the context in which entities such as ‘CN-II’ or ‘dN-1’ first appear, for this reason we separately evaluated the results in Table 4 while omitting abbreviations from the calculations, which resulted in a recall of 0.76. A logical next step is to further improve abbreviation detection algorithms (currently using rule-based methods) by taking a DL approach to focus on relation extraction between abbreviations and definitions.

Another aspect we identified is that punctuation and hyphens impact the performance of the annotation pipeline (e.g. ‘dipeptidyl peptidase-4’ and ‘coenzyme A-linked enzyme’ were not identified by the pipeline). Further improving the pipeline requires both programming new rule-based algorithms and a human-in-the-loop system to more effectively devise rules to improve the accuracy of the automatic annotation pipeline.

As an alternative to dictionary searching combined with rule-based annotation we created two DL models based on the BiLSTM architecture that achieved state-of-the-art results in metabolite named-entity recognition (32). Here we extended that work, in the context of enzyme NER, by comparing the best embedding found previously (BioBERT (12)) with the newer SciBERT (11) model which includes its own vocabulary. Both methods achieve very similar precision (0.97) on the full test data, which increases to 0.98 when excluding abbreviations, however their recall far exceeds the annotation

<table>
<thead>
<tr>
<th>Domain</th>
<th>Annotated enzymes</th>
<th>Mean</th>
<th>St. dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GWAS</td>
<td>6,736</td>
<td>8</td>
<td>17.77</td>
</tr>
<tr>
<td>proteomics</td>
<td>10,037</td>
<td>23</td>
<td>29.30</td>
</tr>
<tr>
<td>metabolomics</td>
<td>9,447</td>
<td>9</td>
<td>13.76</td>
</tr>
<tr>
<td>microbiome</td>
<td>5,502</td>
<td>5</td>
<td>7.88</td>
</tr>
<tr>
<td>total</td>
<td>31,722</td>
<td>9</td>
<td>17.11</td>
</tr>
</tbody>
</table>

**Table 3.** The number of components of entities identified by keyword searching for the 10 most found root terms separated out by the number of words.

<table>
<thead>
<tr>
<th>Root term</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>≥4</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>kinase</td>
<td>631</td>
<td>446</td>
<td>201</td>
<td>74</td>
<td>1,352</td>
</tr>
<tr>
<td>protease</td>
<td>661</td>
<td>452</td>
<td>108</td>
<td>27</td>
<td>1,248</td>
</tr>
<tr>
<td>polymerase</td>
<td>1,017</td>
<td>140</td>
<td>27</td>
<td>13</td>
<td>1,197</td>
</tr>
<tr>
<td>synthase</td>
<td>53</td>
<td>107</td>
<td>257</td>
<td>105</td>
<td>522</td>
</tr>
<tr>
<td>dehydrogenase</td>
<td>137</td>
<td>232</td>
<td>36</td>
<td>17</td>
<td>422</td>
</tr>
<tr>
<td>metalloproteinase</td>
<td>117</td>
<td>178</td>
<td>24</td>
<td>2</td>
<td>321</td>
</tr>
<tr>
<td>phosphatase</td>
<td>68</td>
<td>65</td>
<td>83</td>
<td>25</td>
<td>241</td>
</tr>
<tr>
<td>reductase</td>
<td>98</td>
<td>97</td>
<td>37</td>
<td>8</td>
<td>240</td>
</tr>
<tr>
<td>transferase</td>
<td>71</td>
<td>83</td>
<td>57</td>
<td>24</td>
<td>235</td>
</tr>
<tr>
<td>hydrolase</td>
<td>64</td>
<td>134</td>
<td>25</td>
<td>9</td>
<td>232</td>
</tr>
</tbody>
</table>
pipeline with 0.94 for the BioBERT-BiLSTM and 0.95 for the SciBERT model. This is also reflected in the F1-scores of 0.96 for SciBERT-BiLSTM, 0.95 for BioBERT-BiLSTM and 0.81 for the annotation pipeline. When excluding abbreviations these increase to 0.97 (SciBERT-BiLSTM), 0.96 (BioBERT-BiLSTM) and 0.86 (annotation pipelines). This highlights the benefit of using DL algorithms for NER over (traditional) dictionary-based approaches where a slightly lower precision for DL methods is accompanied with a substantially higher recall.

Intuitively, we would expect that the BioBERT-based model would yield better performance than SciBERT, since it was specifically pretrained on a biomedical corpus. However, we find that the SciBERT-based model (for tokenisation and work embedding) has the best performance overall. SciBERT (11) was trained on a corpus that contains over 80% of papers from the broad biomedical domain which results in the transformer learning features of the words frequently used in biomedical literature. In addition, SciBERT constructed a new vocabulary, SCIVOCAB, for tokenisation and word embedding which has a substantial difference from the WordPiece vocabulary used by BERT and inherited by BioBERT (which was built on top of BERT). As Beltagy et al. indicate, the overlap between the two vocabularies of tokens is a mere 42%, and SCIVOCAB provides more detailed tokens with biomedical features (11).

With the development of biomedical NLP, more and more DL architectures are built or pretrained as transformer models, with BioBERT (12) being the most popular. BioBERT has been reused by the same authors to create a model (BERN2) for joint NER and named entity normalisation for 9 different entities including proteins (achieving an F1-score of 0.84 on the BC2GM corpus (24)). To the best of our knowledge, our work is the first time in which algorithms are developed for an annotation pipeline and NER models specifically for enzymes (opposed to all proteins). We therefore evaluate the performance of the BERN2 model on our manually annotated enzyme corpus and compare it with our enzyme-specific models. Since not all proteins are enzymes, and hence there can be many false positives, we evaluate BERN2 using the recall only. BERN2 achieves a recall of 0.86, clearly outperforming our dictionary-based annotation pipeline but not reaching the same level as our dedicated enzyme NER models (recall 0.94-0.95).

It remains to be seen whether a pure transformer model is able to reach similar results and this would form a logical next step of the work that can build on our creation of a corpus for enzyme-specific NER. Moreover, our annotated data contains over 30,000 entities (Table 2) compared with approximately 20,000 for the BC2GM corpus (24). While the protein NER corpora will include enzymes in the training data, our training set is much larger which means that the models can be trained and evaluated with more data. Moreover, the methods used here for enzymes are amenable to any type of protein to facilitate the creation of a multi-task protein NER model and other biomedical text-mining tools.

The evaluation of the NER tasks is a significant step of improving the training model performance. In this study, our metrics for the DL models were generated based on tokens, i.e., the model classifies every token to its unique label from the (‘S’, ‘O’, ‘B’, ‘I’, ‘E’) scheme, and we have a classification evaluation that calculates these terms of TP, FP, FN and then compute the precision, recall, and F1-score. This evaluation was used to get the exact accuracy for each token, but there are other, potentially more realistic, schemas to evaluate these models that consider partial matches such as the PPs. The Message Understanding Conference (MUC) (39) has introduced some metrics for evaluation considering the different categories of misclassified values against the strict classification, with the International Workshop on Semantic Evaluation (SemEval) resulting in the measurement to calculate corresponding precision, recall, and F1-score based on the MUC metrics. However, while these metrics give a better representation of the errors and performance, they are not used in the evaluation of modern NER models and as such would not make it possible for us to compare our results with models such as BERN2.

### Model Application to Improve Metabolite NER

Our prior work (32) showed that some entities recognised as metabolites are actually part of enzyme names and hence ought to be considered as false positives when evaluating these in the context of metabolite NER. Our work here on enzyme NER can be used to address this limitation by filtering out any metabolite entities that overlaps with an enzyme entity.

We show in Table 5 several examples of applying both TaBoLiSTM and SciBERT-BiLSTM to the same sentence. We have repurposed the TaBoLiSTM metabolomics corpus (over 1,000 publications in that corpus contain at least 1 enzyme, see Table 1) in this work, therefore in order to evaluate how SciBERT-BiLSTM can improve TaBoLiSTM we

<table>
<thead>
<tr>
<th>Model Evaluation</th>
<th>Precision</th>
<th>Recall</th>
<th>F1-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Annotation pipeline - Excl. abbreviations</td>
<td>0.996</td>
<td>0.762</td>
<td>0.863</td>
</tr>
<tr>
<td>BioBERT-BiLSTM BioBERT (12) All</td>
<td>0.968</td>
<td>0.940</td>
<td>0.950</td>
</tr>
<tr>
<td>SciBERT-BiLSTM SciBERT (11) All</td>
<td>0.966</td>
<td>0.951</td>
<td>0.956</td>
</tr>
<tr>
<td>BERN2 (15) BioBERT (12) All</td>
<td>0.134</td>
<td>0.856</td>
<td>0.231</td>
</tr>
</tbody>
</table>

| Annotation pipeline - Excl. abbreviations | 0.981 | 0.937 | 0.955 |
| BioBERT-BiLSTM BioBERT (12) Excl. abbreviations | 0.981 | 0.954 | 0.965 |
| SciBERT-BiLSTM SciBERT (11) Excl. abbreviations | 0.148 | 0.824 | 0.250 |

Table 4. Performance metrics for different models evaluated on the manually annotated test set (n = 526 full text articles).
only give examples from articles that were part of either test set. Table 5A contains examples where we identified no benefit of using SciBERT-BiLSTM to improve the output of TABoLiSTM as no entities overlap, hence TABoLiSTM correctly did not identify ‘tyrosine’ as metabolite as it was part of an enzyme name (it was not trained with this purpose). In Table 5B there are several examples where TABoLiSTM reports a metabolite that is actually part of an enzyme (SciBERT-BiLSTM output) such as identifying ‘ceramide’ twice in a sentence where only one of these is a metabolite and the other is part of ‘ceramide synthases’, or identifying ‘deoxycytidine’ as metabolite when SciBERT-BiLSTM identifies it as ‘deoxycytidine kinase’. Moreover, TABoLiSTM (32) may also have an application to our enzyme NER work here as it can be used to identify whether there is a metabolite name directly in front of a root term (in cases where only a root term is identified). This demonstrates that these models can be used together to improve each other, while this can be done in the context of employing these algorithms in parallel, other work (15) has shown that multi-task learning will yield both better performance and small model sizes.

Conclusions
In conclusion, this work contributed a novel dictionary- and rule-based automatic pipeline for enzyme NER with almost perfect precision and which runs in a matter of seconds (on a standard laptop) on an entire full-text article because of the hierarchical dictionary. We used this pipeline to create the first-of-its-kind corpus for enzyme NER to facilitate text-mining. Our best performing model was the SciBERT word embedding and tokenisation combined with a BiLSTM model and this deep learning model achieved a state-of-the-art performance (F1-score>0.95) for enzyme NER. Our proposed pipeline with the DL models can improve more effective enzyme text-mining and information extraction research for literature review and be used to improve other biomedical NER tasks.

The code for automatic annotation, model training and inference, as well as the data for training and testing are available at https://github.com/omicsNLP/enzymeNER.

AUTHOR CONTRIBUTIONS
Conceptualization, J.M.P. and T.B.; Methodology, M.W., J.M.P. and T.B.; Formal Analysis and Investigation, M.W., A.V. and J.M.P.; Writing – Original Draft: M.W.; Writing – Review Editing: J.M.P., M.W., T.B. and A.V. All authors have read and agreed to the published version of the manuscript. The authors thank Cheng Su Yeung and Nazarim Faghri Mirzai for their support and help during the code development.

A.V. is supported by a UK Research and Innovation (UKRI) Centre for Doctoral Training in AI for Healthcare PhD studentship (EPS/S023283/1 (EPSRC)). This research was funded by Health Data Research (HDR) UK and the Medical Research Council (MRC) via an UKRI Innovation Fellowship to T.B. (MR/S003703/1) and a Rutherford Fund Fellowship to J.M.P. (MR/S004033/1). The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

ORCID: 0000-0002-4971-9003 (JMP).
A. No metabolites annotated within enzymes

PMC3406255: “We also determined that pyruvate dehydrogenase, turnover of the TCA cycle, anaplerosis and de novo glutamine and glycine synthesis contributed significantly to the ultimate disposition of glucose carbon.”

PMC3406255: “Detection of glycine in the breast metastasis is of particular interest since phosphoglycerate dehydrogenase, an enzyme in the serine/glycine biosynthesis pathway, is commonly over-expressed in human breast adenocarcinoma, is amplified at the genomic level in a subset of these tumors, and is essential for tumor growth in a human breast cancer xenograft model [29].”

PMC3406255: “For example, mutations in isocitrate dehydrogenase isoforms 1 and 2 are commonly found in low-grade gliomas and influence intermediary metabolism, including some of the pathways analyzed in this work [44–46].”

PMC4525767: “Mesformin has previously been shown to reduce gluconeogenesis and hepatic glucose production [56] and this effect may be at least partially mediated via reduction in glutamic acid/glutamine concentration, although gluconeogenesis is highly regulated by the rate-limiting enzyme, phosphoenolpyruvate carboxykinase.”

PMC4525767: “There is also recent evidence demonstrating that mesformin suppresses gluconeogenesis by inhibiting mitochondrial glycerol phosphate dehydrogenase [57].”

PMC5287439: “N1-methylinosine is found 3’ adjacent to the anticodon at position 37 of eukaryotic tRNAs and is formed from inosine by a specific S-adenosylmethionine-dependent methylase. 25”

PMC6224486: “Moreover, a tryptophan catabolic enzyme, indoleamine 2,3-dioxygenase, has been reported as a central driver of malignant development and progression (39).”

B. Annotation of metabolite within an enzyme, false positive metabolite annotations can be filtered out

PMC3406255: “The labeling patterns in glutamate and glutamine were similar at each carbon, demonstrating that glutamate was converted to glutamine by the enzyme glutamine synthetase (GS).”

PMC3406255: “Pyruvate dehydrogenase (PDH), citrate synthase (CS), complete turnover of the TCA cycle, anaplerosis, and synthesis of glutamate and glutamine from glucose were evident in all eleven tumors.”

PMC3406255: “Anaplerosis is defined as the anaplerotic flux relative to citrate synthase activity.”

PMC3406255: “As measured by 13C NMR isotopomer analysis, the anaplerotic flux in every tumor approached or exceeded entry of carbon into the TCA cycle via citrate synthase, with a total anaplerotic flux of 1.25 ± 0.43 (mean and SD) relative to citrate synthase for the five GBMs infused at 8hr (Table 1), indicating robust anaplerosis in these tumors.”

PMC6346730: “Recently, Nagahashi and colleagues showed the importance of sphingosine kinase 1-produced S1P to breast cancer-induced hemangiogenesis and lymphangiogenesis in a mouse model [44].”

PMC5073280: “The specific activity of glutathione peroxidase (GPx) was expressed as μmoles of NADPH oxidized/min/mg protein as explained by Mills 1959 16.”

PMC5073280: “Superoxide dismutase (SOD) activity was determined following standard method 18.”

PMC5073280: “Similar trend was also observed in non-enzymatic glutathione peroxidase activity (Fig. 2D).”

PMC5073280: “Levels of Protein (A), Lipid Peroxidase (B), Catalase (C), Glutathione Peroxidase (D), Superoxide Dismutase (E), and Reactive Oxygen Species (ROS) (F) in study groups are presented in box and whisker plots (***, **p value < 0.01, and < 0.005, respectively, of Wilcoxon-Mann-Whitney test).”

PMC5073280: “AROs not only showed the highest levels of oxidative stress in terms of ROS and lipid peroxidation activity but also had the lowest proportion of catalase, superoxide dismutase and glutathione peroxidase activity.”

PMC6150202: “For enhancing therapeutic effect, thiopurines are very often used in co-medication with other therapeutics, such as cyclosporin (prednisone, PRE, strong anti-inflammatory agent), nonsteroidal anti-inflammatory drugs (mesalazine, 5-amino-saliclyc acid, MSL, potent scavenger of reactive oxygen species), and enzyme inhibitors (allopropurinol, ALP, inhibitor of xanthine oxidase).”

PMC6224486: “In the present study, the level of monocytylglyceride was low probably due to the high level of monocytylglycerol lipase, which has been shown to promote hepatocellular carcinoma and colorectal cancer (24, 25).”

Bibliography


7. A. No metabolites annotated within enzymes


13. Haoming Jiang, Daniqing Zhang, Tianyu Cao, Bing Yin, and Tuo Zhao. Named entity recognition with small strongly labeled and large weakly labeled data. In Proceedings of the...