Bioinformatics, 2024, **40**, ii45–ii52
https://doi.org/10.1093/bioinformatics/btae402
Proceedings of ECCB2024

Improving dictionary-based named entity recognition with deep learning

Katerina Nastou [1], Mikaela Koutrouli[1], Sampo Pyysalo[2], Lars Juhl Jensen[1,*]

[1]Novo Nordisk Foundation Center for Protein Research, Faculty of Health and Medical Sciences, University of Copenhagen, Blegdamsvej 3, Copenhagen, 2200, Denmark
[2]TurkuNLP Group, Department of Computing, University of Turku, Turku, 20014, Finland

*Corresponding author. Novo Nordisk Foundation Center for Protein Research, Faculty of Health and Medical Sciences, University of Copenhagen, Blegdamsvej 3, Copenhagen, 2200, Denmark; E-mail: lars.juhl.jensen@cpr.ku.dk (L.J.J.)

Abstract

Motivation: Dictionary-based named entity recognition (NER) allows terms to be detected in a corpus and normalized to biomedical databases and ontologies. However, adaptation to different entity types requires new high-quality dictionaries and associated lists of blocked names for each type. The latter are so far created by identifying cases that cause many false positives through manual inspection of individual names, a process that scales poorly.

Results: In this work, we aim to improve block list s by automatically identifying names to block, based on the context in which they appear. By comparing results of three well-established biomedical NER methods, we generated a dataset of over 12.5 million text spans where the methods agree on the boundaries and type of entity tagged. These were used to generate positive and negative examples of contexts for four entity types (genes, diseases, species, and chemicals), which were used to train a Transformer-based model (BioBERT) to perform entity type classification. Application of the best model (F1-score = 96.7%) allowed us to generate a list of problematic names that should be blocked. Introducing this into our system doubled the size of the previous list of corpus-wide blocked names. In addition, we generated a document-specific list that allows ambiguous names to be blocked in specific documents. These changes boosted text mining precision by ∼5.5% on average, and over 8.5% for chemical and 7.5% for gene names, positively affecting several biological databases utilizing this NER system, like the STRING database, with only a minor drop in recall (0.6%).

Availability and implementation: All resources are available through Zenodo https://doi.org/10.5281/zenodo.11243139 and GitHub https://doi.org/10.5281/zenodo.10289360.

1 Introduction

Named entity recognition (NER) is an important task in text mining that allows the identification of domain-specific terms or phrases—also known as named entities—in text and their classification into entity types (Nadeau and Sekine 2007). In the biomedical domain, named entities include genes, chemicals, and diseases among others. NER is a prerequisite for the performance of other text mining tasks that allow the extraction of valuable information from text, such as relation extraction (RE) (Perera *et al.* 2020).

Dictionary-based methods, which leverage pre-defined dictionaries compiled from extensive collections of terms for each entity type, have been a prominent approach in NER (Leser and Hakenberg 2005), as they simultaneously allow both the recognition and normalization of identified names in text. However, constructing dictionaries tailored for biomedical text mining demands significant time and expert knowledge, posing challenges in achieving optimal results (Wang *et al.* 2018).

We have in the past developed a robust, fast dictionary-based tagging engine, the JensenLab `tagger`, which serves as the core of an NER system that is applicable to many biomedical problems (Jensen 2016). The software can process thousands of PubMed abstracts per second and this high speed makes it well-suited for both real-time text mining (Pafilis *et al.* 2016) and NER in massive text corpora (Westergaard *et al.* 2018). Combining this tagger with other dictionaries forms the basis for extraction of associations among genes/proteins (Szklarczyk *et al.* 2023), small molecule compounds (Szklarczyk *et al.* 2016), cellular components (Binder *et al.* 2014), tissues (Palasca *et al.* 2018), and diseases (Grissa *et al.* 2022), among others. Text-mining results obtained from these applications are normalized to identifiers from suitable databases and ontologies (Maglott *et al.* 2000, Hubbard *et al.* 2002, Wang *et al.* 2009, Schriml *et al.* 2012), and are used in several biological databases, which are freely available as a suite of web resources. These databases include the STRING database of protein interactions (Szklarczyk *et al.* 2023).

Despite its simplicity, dictionary-based NER is a powerful method that can perform surprisingly well in the biomedical domain. While software does matter, the most important part of any dictionary-based NER system is the dictionary itself. Adaptation of the NER system to a different domain requires the construction of a new high-quality dictionary and an associated block list of names. The manual creation of block list s for `tagger` was so far done through manual inspection of the names that occur most frequently in a large text corpus to identify those that would produce numerous false positives and should be disregarded. This process scales poorly, due to the manual labor involved.

The majority of the biomedical text mining community has been using deep learning-based and specifically Transformer-based methods (Miranda-Escalada *et al.* 2023) to perform

several tasks, including NER, RE and Question-Answering. Moreover, Large Language Models have recently emerged as an alternative also for tasks other than Question-Answering (Wang *et al.* 2023). We wanted to explore how we could take advantage of deep learning-based methods to improve the JensenLab suite of web resources. The intrinsic nature of these resources necessitates named entity normalization in conjunction with NER, and a fast and lightweight tagging system allowing weekly updates. It is prohibiting in terms of cost to perform such updates using deep-learning methods. Moreover, given the lack of suitable corpora to train deep learning-based methods for named entity recognition and normalization across all entity types, our objective did not center around devising an entirely new method. Rather, we could see a prospect in enhancing the precision of our dictionary-based NER system through the incorporation of state-of-the-art methods.

In this study, we present an alternative method for block list generation, which can bring many of the benefits of deep learning to dictionary-based NER, without sacrificing the advantages of the latter. To address the limitations of manual block list creation, we automated this crucial step by unifying and comparing results of three biomedical NER methods. We sought to create a large comprehensive dataset of text spans where different NER methods agreed on entity boundaries and types. This consensus dataset served as the foundation for generating both positive and negative examples for four entity types: genes or gene products, diseases, species, and chemicals. We then trained a Transformer-based model, specifically BioBERT (Lee *et al.* 2019), to perform entity type classification using that dataset. Ultimately we employed this model to accurately identify problematic names for blocking, resulting in a substantial expansion of the existing block list, ensuring a precision enhancement without compromising recall. In addition, we created document-specific lists to address ambiguous name occurrences within specific documents. The successful implementation of these strategies significantly enhanced the quality of all web resources populated with associations generated by `tagger`. Moreover, this work serves as a guide for automating the identification of names to be blocked in various dictionary-based text mining systems, setting a precedent for more efficient and accurate NER in the biomedical domain.

2 Materials and methods

2.1 The dictionary-based `tagger`

As mentioned above text mining is integral to a suite of web resources with associations among biomedical entities (Jensen 2016). The initial step in extracting these associations from the literature involves the recognition of biomedical entities by the `tagger`, laying the foundation for relation extraction (RE). It is evident that the quality of the NER system directly impacts the quality of the generated associations.

Blocking and allowing names for the `tagger` is a key step to ensuring good quality. It operates across two levels: global (i.e. corpus-wide) and local (i.e. document-specific), utilizing two distinct methods for incorporating terms: automatic and manual additions. The generation of manual lists involves meticulous inspection of the most frequent names in the literature before addition, while that of automatic lists does not involve any human intervention. Both lists are combined to create a final list, favoring the manually curated list in case of conflicting overlaps between the two.

Operations on the global level involve both global blocking to identify names generally unsuitable for text mining, such as “Bad” or “ask,” which should be altogether removed for increased precision, and global allowing to safeguard names verified and accurately labeled as biomedical entity names of the correct type (e.g. “ATM”). Furthermore, the globally curated allowing functionality acts as a protection against potential inaccuracies from an auto-generated global block list. In local contexts, blocking or allowing decisions are tailored to the document at hand. For instance, “wingless” predominantly signifies a protein, yet in select instances, it refers to apterous insects, and local blocking in these instances is preferred to incorrect tagging. Similarly, a name like “toy,” which should be blocked globally, might be allowed in a specific document where it refers to the transcription factor “toy.” Despite the potential for both manual and automatic approaches in local contexts, considering the amount of effort these require, only automatic additions of names to the local list make sense.

The `tagger` functions in a case-insensitive manner and allows for arbitrary insertion or deletion of spaces and hyphens in the names and punctuation characters before or after them, both important for capturing the orthographic variation of biomedical named entities (Pafilis *et al.* 2013). By contrast, all lists mentioned above function in a case-sensitive manner, affecting only specific versions of a name when added therein (e.g. adding “bad” will not affect the tagging of “BAD”).

During dictionary generation, clashes may arise, such as “NO” existing as a synonym in both the gene and chemical dictionaries. To resolve these clashes, a preference hierarchy based on source quality is implemented during the generation of the dictionaries for `tagger`. Diseases hold precedence over species, followed by genes or gene products, and finally chemicals. This hierarchy dictates that in the example above, “NO” will be tagged as a gene unless otherwise filtered. The resolution of such conflicts necessitates specific entity type filtering during dictionary creation. By employing filters, conflicts like overlapping gene and chemical names for “NO” can be addressed, ensuring accurate tagging.

The trust hierarchy in dictionaries and the inherent complexity of manually integrating terms for resolving dictionary clashes emphasize the significance of automatically blocking names after dictionaries have been built. This automatic process stands as the primary approach for enhancing the precision of `tagger` results, presenting a streamlined and effective means for optimizing accuracy for both NER and RE. For more details on dictionary generation for the `tagger`, please refer to (Jensen 2016).

2.2 Consensus dataset

The first step toward creating a method for automatically detecting problematic names in a biomedical dictionary is to identify high-quality examples to train the method on. These come in the form of a labeled dataset that allows supervised training for the task of biomedical entity type classification. To ensure that such a dataset has high coverage, a very large pool of both positive and negative examples is required. Since obtaining enough examples of properly labeled named entities via manual annotation is prohibitive—due to the time and resource demands of the effort—we opted for an

automatic approach, which can provide a dataset that is simultaneously large and of high quality. To this end, we compared the NER results of PubTator (Wei *et al.* 2019), EVEX (Van Landeghem *et al.* 2013), and `tagger` (Jensen 2016), thus identifying millions of text spans where all three methods agree on the boundaries and the type of entity tagged.

As mentioned, `tagger` is a dictionary-based method, with a dictionary based on Ensembl (Hubbard *et al.* 2002), Refseq (Maglott *et al.* 2000), PubChem (Wang *et al.* 2009), NCBI taxonomy (Schoch *et al.* 2020), and Disease Ontology (Schriml *et al.* 2012). On the other hand, PubTator encompasses different text-mining tools (Huang *et al.* 2011, Wei *et al.* 2012, Leaman *et al.* 2013) to detect biomedical entities following different nomenclatures than `tagger`, while EVEX uses the Turku Event Extraction System (Björne *et al.* 2010) together with BANNER (Leaman and Gonzalez 2008) and the McClosky-Charniak Parser (McClosky and Charniak 2008). The selection of the three systems is appropriate for creating a highly precise consensus dataset of training examples for genes or gene product (ggp), disease (dis), species (org), and chemical (che) named entities, as these methods rely on different architectures and/or nomenclatures to produce NER results for the four entity types they have in common.

To train a model for classifying entities, we designed a dataset consisting of biomedical entities within their textual context, all labeled with the biomedical entity's type. To achieve this, we constructed positive examples using the consensus results of all three aforementioned NER methods, i.e. matches where all three methods agree on boundaries and entity type, comprising 200-word contexts around instances of the four designated entity types (100 words to the left and right of each entity mention).

At the same time, we compiled a negative set from contexts around nouns and noun phrases where none of the methods produced a match corresponding to any of the targeted named entity types (neg class). Specifically, for the neg class, we only required the three NER methods to agree on either tagging something different from the entity types used to create the positive dataset, without requiring them to agree on the type, or not to tag anything at all. As a result of this approach, we also excluded all matches where either only one or two of the NER methods tagged an entity of the positive named entity types (ggp, dis, org, and che). When creating the neg class set using these criteria, it is almost certain that the nouns and noun phrases therein are not names of any of the positive entity types. Nonetheless, they can still refer to an entity of one of those types, and the contexts of names and non-name references can be identical. Therefore, we filtered our negatives to remove various likely confusing cases. We implemented simple noun phrase head detection heuristics and compiled a filter list of potentially confusing head terms. In brief, noun phrases headed by pronouns (e.g. "it," "that," "these"), numbers (e.g. "one," "two"), and top- or intermediate-level terms for target entity types (e.g. "protein," "enzyme," "cytokine") or known inflected forms were removed from the negative set of contexts. The filter list we used can be found on Zenodo.

2.3 Experimental setup

The task of entity type classification has clear analogies to NER, an area where the current state-of-the-art methods predominantly utilize models based on the Transformer architecture (Vaswani *et al.* 2017). These models are initially pre-trained on large collections of text to produce a general language model, and can later be fine-tuned to perform specific tasks, such as NER. We have selected the cased BioBERT base model v1.1 (Lee *et al.* 2019), a biomedical domain-specific pre-trained deep language representation model based on BERT (Devlin *et al.* 2018), to train a multi-class entity type classifier on the consensus dataset described above.

BERT-based models are pre-trained using masked language modeling (Devlin *et al.* 2018), i.e. hiding certain words with a special mask token and asking the model to fill in the gaps. We have leveraged this feature of BioBERT to train our classifier to learn only how the contexts around each biomedical entity type look, without access to the text of the named entity mentions themselves. Specifically, we masked all entity mentions of interest in the consensus set, and set the label for each example based on the masked entity, e.g. "*Steroid excretion in patients receiving [MASK] compounds.*" would get a *chemical* label, while "*When [MASK] becomes mutated, it loses its function.*" would get a *gene or gene product* label. We have generated two datasets with the format described above: a 125k dataset with 125 000 training examples and a 12.5M dataset of 12 500 000 training examples.

During fine-tuning, we updated the weights of the pre-trained BioBERT model to classify the examples described above as belonging to one of the five training classes (ggp, org, dis, che, neg) with the output softmax layer configured to transform the contextualized representations generated by the BERT model into segment-level probabilities for each class. These probabilities were derived by aggregating information from tokens within each segment, enabling the prediction of the most likely class for the entire segment. To select the best model, we used the 125k consensus dataset described above and assigned 125 000 examples to the training set and 62 500 to the development set. Using this dataset we tuned the hyperparameters of the BioBERT model through a grid search to find the best combination of values for the learning rate (5e−5, 3e−5, 2e−5, 1e−5, and 5e−6), mini batch size (32 and 64), number of epochs (2, 3, and 4), maximum sequence length (256), and context size (200). The values in parentheses for each parameter were those suggested in the BioBERT publication and some extensions to those, based on our initial experiments. For more details on the experimental setup please refer to our code base, available through GitHub. The experiments were repeated four times and the results are expressed as a mean and standard deviation of the F1-score (micro-averaged over all labels in the sets). The hyperparameters producing the best total mean F1-score on the development set were selected for training a model on the 12.5M consensus dataset—12.5 million examples were used for training and 62 500 to calculate the test set performance on this set.

2.4 Block list generation and prediction

To generate automated block list s, we used all PubMed abstracts (as of August 2022) and all full-texts available in the PMC BioC text mining collection (Comeau *et al.* 2019) (as of April 2022). We then ran `tagger` using the manually curated block list s only. We generated examples for all contexts surrounding the `tagger` matches of the four positive classes, and we used the best-performing model for entity type prediction on these examples.

As explained above, for each example, the classifier produces a probabilistic score of belonging to each class (including

the negative class) and, based on these, we can calculate the score of not belonging to the tagger-assigned class P(¬C), where C stands for class. To generate a corpus-wide block list, we do an unweighted score average of all P(¬C) within each document for each name. We then average these document-level scores across the entire corpus–considering that matches in different documents are independent–and generate a single score per name of not belonging to the tagger -assigned class across the entire literature. To generate the global block list we had an additional requirement, that there are at least two documents with one mention for each name.

Afterwards, we manually inspected the lists to identify the thresholds that we would use to add names to the global block list, since those could be different for each class. The fact that we had a probabilistic score for each match allowed us to explore the use of local (document-level) predictions to resolve ambiguous names, by aggregating within-document scores for each name. This allowed us to test both the generation of lists of terms that could be locally blocked or allowed—the latter only for terms that are automatically blocked globally. Once again, the scores that would serve as thresholds for inclusion in these lists had to be manually identified by inspecting random samples of the lists. To identify names that would be locally allowed we calculated a ratio based on the formula below:

$$ratio = \frac{P(C|N) \cdot P(C|L)}{P(\neg C|N) \cdot P(\neg C|L)} \quad (1)$$

where C stands for class and corresponds to the five entity types used to train the classifier (gene or gene product, disease, species, chemical and negative), P(C|N) is the global probabilistic score of a term belonging to that specific class, P(C|L) is the local probabilistic score estimate for the term belonging to that class, and ¬C is "notC," so $P(\neg C|N) = (1 - P(C|N))$ and $P(\neg C|L) = (1 - P(C|L))$. This formula allows us to override even very high global confidence scores when trying to add names to the local allow list. The calculation of the ratio for identifying names to be locally blocked is analogous to Equation (1).

2.5 Evaluation

To evaluate the effects of block list s we generated four sets of results by performing four different tagger runs:

1) run with a combined automatically generated and manually curated block list (***curated+auto***): This run involved using a combination of both automatically generated and manually curated block list s for tagger and resembles what is currently done for JensenLab web resources.
2) run with manually curated block list s (***curated_only***): in this run, only the manually curated block list was used.
3) run with automatically generated block list s (***auto_only***): in this run we only used block list s that were automatically generated from the fine-tuned BioBERT model. To generate these lists, we had to run tagger without any block list (only the built-in tagger regular expressions were used to block names) and we repeated the process for block list generation, using the thresholds that we determined from the process described in the previous section.
4) run without any block list (***no_block_list***): this involved running tagger without utilizing any block list. We also adapted the tagger code to exclude the use of built-in regular expressions. The adapted code is provided through Zenodo.

We conducted evaluations using two distinct approaches. Firstly, since the main incentive for updating the block list s is improving the JensenLab suite of web resources, we aimed to assess the impact of block list s in co-occurrence-based relation extraction. Specifically, we evaluated the results from the four runs above against two datasets: the KEGG database (Kanehisa and Goto 2000) gold standard used for evaluations by the STRING database (Szklarczyk *et al.* 2023), and the gold standard of protein-disease associations used by the DISEASES database (Grissa *et al.* 2022).

Secondly, to evaluate the direct impact of automatically adding names in the block lists on NER, we manually checked 500 random matches of each class from the ***curated_only*** run and calculated tagger 's precision for each class using the formula below:

$$prec_{curated} = \frac{TP_{curated}}{TP_{curated} + FP_{curated}} \quad (2)$$

where $TP_{curated}$ is the number of true positive matches, i.e. those belonging to the class assigned by the tagger, and $FP_{curated}$ is the number of false positive matches.

To assess the effect of the addition of the automatic block list, we checked 200 matches per class that got removed as a result of incorporating the automatically generated block list. For this purpose, we compared the results of the ***curated_only*** run and the ***curated+auto*** run and identified the set of matches that got removed between these two runs. Then we randomly selected a subset of these removed matches to inspect, and calculated the precision of automatic blocking using the formula below:

$$prec_{blocking} = \frac{FP_{blocked}}{TP_{blocked} + FP_{blocked}} \quad (3)$$

The $prec_{blocking}$ is the fraction of automatically blocked names that were correctly blocked, i.e. former FPs that were removed by blocking.

To calculate the effect of automatic blocking on precision and recall, we need to introduce weighted TPs and FPs:

$$TP_w = A \cdot x \cdot (1 - prec_{blocking}) \quad (4)$$

$$FP_w = A \cdot x \cdot prec_{blocking} \quad (5)$$

where $A = 500$ ($TP_{curated} + FP_{curated}$) and x is the class-specific automatic block rate (calculated from the difference in the number of matches between the ***curated_only*** and ***curated+auto*** runs). From these, the precision using both curated and automatic block list ($prec_{c+a}$) and the relative difference in recall (rec_{diff}) for a given class can be calculated as:

$$prec_{c+a} = \frac{TP_{curated} - TP_w}{(TP_{curated} - TP_w) + (FP_{curated} - FP_w)} \quad (6)$$

$$rec_{diff} = -\frac{TP_w}{TP_{curated}} \quad (7)$$

3 Results and discussion

3.1 Dataset generation and model training

By obtaining the consensus results of tagger, PubTator, and EVEX, we generated a dataset with millions of text spans of positive and negative examples, which we split into a 125k and a 12.5M set as described in. The number of examples was balanced between the five classes, as shown in Table 1 for the training datasets. A manual inspection of 500 names (100 of each class) from this dataset allowed us to calculate its precision, which was found to be 99.8% (Supplementary Table S1, available through Zenodo), showcasing the high quality of this dataset.

Our hyperparameter grid search on the 125k set revealed that the $BioBERT_{base}$ model with a learning rate of 2^{-5} and a batch size of 32 was the best model trained on the set of 125 000 examples, with a mean F-score of 94.83% (SD = 0.0356%) on the development set. All models in this grid had a mean F-score of 94% or above. Detailed results are presented in Supplementary Table S2. We used the set of hyperparameters above to train a model on the 12.5M set for one epoch. This model achieved an accuracy of 96.67% (SD = 0%) on the test set of 62 500 examples and is the one that is further used for all prediction runs. The fine-tuned Tensorflow model on the 12.5M set is available through Zenodo.

3.2 Block list generation

The tagger was executed using the full tagger dictionary and only manually curated lists on a corpus of the entire biomedical scientific literature as of August 2022, which consisted of 34 420 049 documents. The dictionary and block list files, as well as the input documents used in this run are available via Zenodo. There were a total of 75 419 774 gene or gene product, 78 615 064 chemical, 39 995 175 disease, and 62 921 879 species matches as a result of that run. For each match, a 200-word context was generated from the input document and classified using the fine-tuned model for entity type prediction.

Using the process detailed in, we generated an initial set of global block list s to identify the probabilistic score threshold to use for each entity type. These were chosen empirically based on the relative densities of true and false positives at threshold values for P(¬C) in steps of 0.05 starting from 0.5. A spreadsheet with the lists of names checked during this process is available through Zenodo. The best threshold choice for genes or gene products, species, and chemicals was 0.5, while for diseases the value was 0.85.

Table 1. Number of examples for each class in the 12.5M and 125k training sets.

Entity type	Number of examples	
	12.5M set	125k set
Gene or gene product	2 292 475	23 105
Chemical	3 705 213	37 205
Disease	2 083 848	20 795
Species	1 918 464	19 402
Negative	2 500 000	24 493
Total	12 500 000	125 000

We also had to manually determine the ratios for local allow/block list s, by manually inspecting a random set of blocked matches within their textual context. This inspection showed that the generation of good quality lists for diseases and species was not possible. So we generated lists only for genes or gene products and chemicals, using a ratio [Equation (1)] of 1000 as the threshold of inclusion. As for the local allow lists genes or gene products was the only class with good quality lists and the ratio for those was set to a very high value (10^{15}).

Introducing the lists above into the tagger dictionary approximately doubled the size of the global list from 148 143 names to 350 862 names. The local (document-specific) allow/block list was introduced for the first time in the text-mining system and contains 155 660 names blocked and 9611 names allowed in 144 585 and 9458 documents, respectively. This led to a block rate of 7.4%, leading to a total reduction of 51 675 498 unique tagger matches in the four targeted classes due to the additional names that were blocked (from 693 282 180 matches to 641 606 682 matches). Specifically, for chemicals the block rate is ∼11% (24 026 228 matches removed), ∼9% for genes or gene products (20 551 927 matches removed), ∼4% for species (4 897 488 matches removed), and ∼2% for diseases (2 199 855 matches removed).

3.3 Evaluation

To measure the effect of introducing automatically generated block lists on tagger 's performance, we run tagger using four different setups as described in Methods. We detected all co-occurring protein–protein and protein–disease pairs in the literature resulting from these four runs and evaluated the performance of co-occurrence-based relation extraction using a functional protein–protein association gold standard from KEGG (Kanehisa and Goto 2000) and a protein–disease gold standard from DISEASES (Grissa *et al.* 2022). The results from this evaluation are shown in Fig. 1.

Figure 1 shows that the incorporation of any block list in tagger markedly enhances the quality of its generated results. Notably, best results are achieved with our newly proposed method, i.e. by the combination of the automatically generated and the curated block list s. However, the differences between the three block list-including runs are not particularly pronounced. Even when focusing on the initial segments of the curves where more cumulative true positives are expected, the differences are still slim. A potential explanation for this result is that proteins present in the two gold standard datasets tend to be well-studied. This means that they will frequently show up in the literature and that problematic names are thus relatively easy to find both by manual curation (green line) and by the automatic approach (orange line). Consequently, these datasets may not accurately reflect the potential benefits derived from the automated blocking of names, as manual curation will almost certainly give worse coverage of the less studied proteins, which are not in these gold standards. We thus opted to employ a second strategy.

For the second strategy, we assessed the quality of automatically blocked names by manually inspecting them. Table 2 shows the precision of tagger per class when using only the curated block list and the effect of automatic blocking of names, allowing us to see the changes in the dictionary-based system's precision and recall. To calculate the initial precision of tagger for each class we have selected

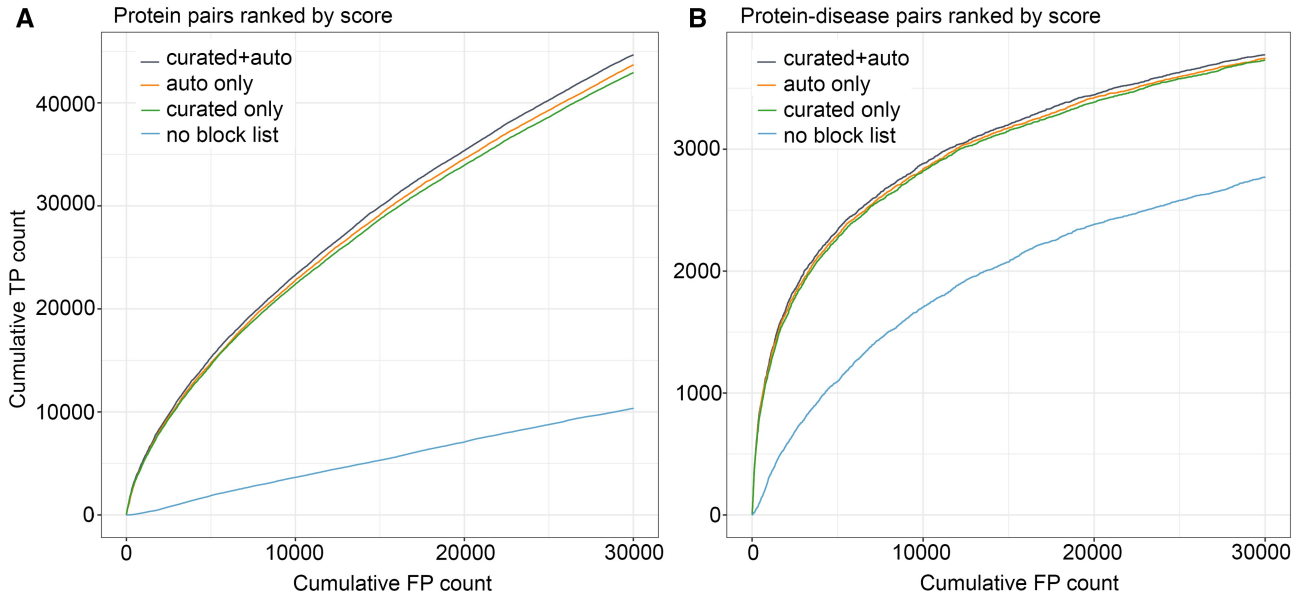

Figure 1. Cumulative true positive (TP)/false positive (FP) curves to evaluate human (A) functional protein–protein associations and (B) protein–disease associations. The pairs are ranked based on their significance score. For details on the calculations of this score please refer to Franceschini *et al.* (2012). The curves are showing only up to Cumulative FP count of 30 000 pairs.

Table 2. Precision and relative recall calculations for `tagger`.

Entity type	Precision curated (%)	Precision curated+auto (%)	Precision difference (%)	Recall difference (%)
Gene or gene product	89.8	97.6	+7.8	−1.1
Chemical	76.2	85.1	+8.9	−0.6
Disease	98.4	99.8	+1.4	−0.6
Species	88.4	91.8	+3.4	−0.3
Average	88.2	93.6	+5.4	−0.6

500 matches from the **curated_only** run for each of the four targeted entity types, did a manual evaluation of the correct assignment to this class by `tagger` and assigned a TP or FP label (Supplementary Tables S3–S6, available through Zenodo). To assess if a match is a TP or FP, we manually check the names within the specified context (i.e. within the documents in which the matches appear), to assess if they belong to their assigned class. Correct normalization was not a requirement to assign a TP label. The precision of `tagger` when run using only the curated list is calculated using formula (2) and is presented in the first column of Table 2.

Then we randomly selected 200 matches for each class, that were removed due to the introduction of the automatically generated block list and manually checked to evaluate which ones were TPs and which were FPs (Supplementary Tables S7–S10). We used formula (6) to calculate the precision after the introduction of this block list. Moreover, we have used formula (7) to calculate the relative difference in recall due to the addition of the automatically generated block list.

Overall, one can see that while the introduction of the automatically generated block list s will inadvertently result in a recall reduction–due to the blocking of some good names–the precision gain clearly overshadows the former. The effect is more evident for classes where dictionary-based NER faces more problems, like chemicals, and very slight in cases where dictionary-based text mining already performs very well, like diseases. These results support our decision to introduce the automatically generated block list s in `tagger` 's dictionaries and have a positive effect for JensenLab web resources, including the latest version of the STRING database. In addition, for evaluating the effect of the new block list on dictionary-based species NER, we used the recently published S1000 corpus (Luoma *et al.* 2023). The F-score increased by 1% (from 84.7% to 85.7%) in this dataset, which was a result of a 2.6% increase in precision and a 0.4% decrease in recall, showcasing an overall positive effect.

4 Conclusions

In this work, we propose a workflow designed to automate the process of adding names in a block list for dictionary-based NER systems and use the `tagger` to assess the results of this process. We have generated and used a very large dataset of 12.5 million text spans, to generate positive and negative examples of contexts for four entity types, namely genes or gene products, diseases, species and chemicals, and a negative class. By training a highly accurate (F-score = 96.7%) Transformer-based entity type classifier on this dataset, and using it to do entity type prediction on all `tagger` matches resulting from a run with a curated block list, we have managed to effectively double the size of the global block list that `tagger` uses, which would not have been feasible by manual curation. At the same time, we have incorporated for the first time an extensive automated local allow/block list which supports the document-specific resolution of ambiguous names.

Evaluation of the newly generated lists demonstrated positive effects on both dictionary-based NER and RE.

Subsequently, these enhanced lists have been directly incorporated into `tagger`, without altering how the tool works or adversely affecting its speed. The improved dictionaries and annotation results are thus made publicly available immediately, through the RESTful APIs that serve the web resources with text mining results produced by the `tagger`, including the widely used STRING database. This integration ensures immediate access to improved resources, advancing the utility and accuracy of `tagger` in several biomedical text mining applications.

Acknowledgements

We thank the CSC—IT Center for Science for generous computational resources. We would like to thank Rebecca Kirsch for providing the code to generate Fig. 1.

Supplementary data

Supplementary data are available at *Bioinformatics* online.

Conflict of interest

None declared.

Funding

This work was supported by Novo Nordisk Foundation [grant number NNF14CC0001] and from the Academy of Finland [grant number 332844]. K.N. has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie [grant number 101023676]. M.K. has received funding from Novo Nordisk Foundation [grant number NNF20SA0035590]. This paper was published as part of a supplement financially supported by ECCB2024.

Data availability

The data underlying this article are available in Zenodo https://doi.org/10.5281/zenodo.11243139 and GitHub https://doi.org/10.5281/zenodo.10289360.

References

Binder JX, Pletscher-Frankild S, Tsafou K *et al.* COMPARTMENTS: unification and visualization of protein subcellular localization evidence. *Database* 2014;**2014**:bau012.

Björne J, Ginter F, Pyysalo S *et al.* Complex event extraction at Pubmed scale. *Bioinformatics* 2010;**26**:i382–90.

Comeau D, Wei C, Islamaj Doğan R *et al.* PMC text mining subset in bioc: about three million full-text articles and growing. *Bioinformatics* 2019;**35**:3533–5.

Devlin J, Chang M-W, Lee K *et al.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol 1 (Long and Short Papers)*, pp. 4171–86.

Franceschini A, Szklarczyk D, Frankild S *et al.* STRING v9.1: protein–protein interaction networks, with increased coverage and integration. *Nucleic Acids Res* 2012;**41**:D808–15.

Grissa D, Junge A, Oprea TI *et al.* DISEASES 2.0: a weekly updated database of disease–gene associations from text mining and data integration. *Database* 2022;**2022**:baac019.

Huang M, Liu J, Zhu X. Genetukit: a software for document-level gene normalization. *Bioinformatics* 2011;**27**:1032–3.

Hubbard T, Barker D, Birney E *et al.* The Ensembl genome database project. *Nucleic Acids Res* 2002;**30**:38–41.

Jensen LJ. One tagger, many uses: illustrating the power of ontologies in dictionary-based named entity recognition. bioRxiv, https://doi.org/10.1101/067132, 2016, preprint: not peer reviewed.

Kanehisa M, Goto S. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res* 2000;**28**:27–30.

Leaman R, Gonzalez G. BANNER: an executable survey of advances in biomedical named entity recognition. *Pac Symp Biocomput* 2008: 652–63. https://pubmed.ncbi.nlm.nih.gov/18229723/.

Leaman R, Islamaj Dogan R, Lu Z. Dnorm: disease name normalization with pairwise learning to rank. *Bioinformatics* 2013;**29**:2909–17.

Lee J, Yoon W, Kim S *et al.* BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 2019;**36**:1234–40.

Leser U, Hakenberg J. What makes a gene name? named entity recognition in the biomedical literature. *Brief Bioinform* 2005;**6**:357–69.

Luoma J, Nastou K, Ohta T *et al.* S1000: a better taxonomic name corpus for biomedical information extraction. *Bioinformatics* 2023; **39**:btad369.

Maglott D, Katz K, Sicotte H *et al.* Ncbi's locuslink and refseq. *Nucleic Acids Res* 2000;**28**:126–8.

McClosky D, Charniak E. Self-training for biomedical parsing. In: *Annual Meeting of the Association for Computational Linguistics*, 2008, 101–4. https://www.semanticscholar.org/paper/Self-Training-for-Biomedical-Parsing-McClosky-Charniak/170479ac946ef294455005cc6fb6adb1d2df7a4f.

Miranda-Escalada A, Mehryary F, Luoma J *et al.* Overview of DrugProt task at BioCreative VII: data and methods for large-scale text mining and knowledge graph generation of heterogenous chemical–protein relations. *Database* 2023;**2023**:baad080.

Nadeau D, Sekine S. A survey of named entity recognition and classification. *LI* 2007;**30**:3–26.

Pafilis E, Frankild SP, Fanini L *et al.* The species and organisms resources for fast and accurate identification of taxonomic names in text. *PLoS One* 2013;**8**:e65390.

Pafilis E, Buttigieg PL, Ferrell B *et al.* Extract: interactive extraction of environment metadata and term suggestion for metagenomic sample annotation. *Database* 2016;**2016**:baw005.

Palasca O, Santos A, Stolte C *et al.* Tissues 2.0: an integrative web resource on mammalian tissue expression. *Database* 2018; **2018**:bay003.

Perera N, Dehmer M, Emmert-Streib F. Named entity recognition and relation detection for biomedical information extraction. *Front Cell Dev Biol* 2020;**8**:673.

Schoch CL, Ciufo S, Domrachev M *et al.* NCBI taxonomy: a comprehensive update on curation, resources and tools. *Database* 2020; **2020**:baaa062.

Schriml LM, Arze C, Nadendla S *et al.* Disease ontology: a backbone for disease semantic integration. *Nucleic Acids Res* 2012; **40**:D940–6.

Szklarczyk D, Santos A, Von Mering C *et al.* STITCH 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic Acids Res* 2016;**44**:D380–4.

Szklarczyk D, Kirsch R, Koutrouli M *et al.* The STRING database in 2023: protein–protein association networks and functional enrichment analyses for any sequenced genome of interest. *Nucleic Acids Res* 2023;**51**:D638–46.

Van Landeghem S, Björne J, Wei C-H *et al.* Large-scale event extraction from literature with multi-level gene normalization. *PLoS One* 2013;**8**:e55814.

Vaswani A, Shazeer N, Parmar N *et al.* Attention is all you need. In: *NIPS'17*. Red Hook, NY, USA: Curran Associates Inc., 2017, 6000–6010.

Wang S, Sun X, Li X *et al.* Gpt-ner: named entity recognition via large language models. arXiv, arXiv:2304.10428, 2023, preprint: not peer reviewed.

Wang X, Yang C, Guan R. A comparative study for biomedical named entity recognition. *Int J Mach Learn Cyber* 2018;**9**:373–82.
Wang Y, Xiao J, Suzek TO *et al.* PubChem: a public information system for analyzing bioactivities of small molecules. *Nucleic Acids Res* 2009;**37**:W623–33.
Wei C, Kao H, Lu Z. SR4GN: a species recognition software tool for gene normalization. *PLoS One* 2012;**7**:e38460.
Wei C, Allot A, Leaman R *et al.* PubTator Central: automated concept annotation for biomedical full text articles. *Nucleic Acids Res* 2019;**47**:587–93.
Westergaard D, Stærfeldt H-H, Tønsberg C *et al.* A comprehensive and quantitative comparison of text-mining in 15 million full-text articles versus their corresponding abstracts. *PLoS Comput Biol* 2018;**14**:e1005962.

Bioinformatics, 2024, 40, 45–52
https://doi.org/10.1093/bioinformatics/btae402
Proceedings of ECCB2024