Joint Learning for Biomedical NER and Entity Normalization: Encoding Schemes, Counterfactual Examples, and Zero-Shot Evaluation

Jiho Noh
jiho.noh@uky.edu
Department of Computer Science
University of Kentucky
Lexington, Kentucky, USA

Ramakanth Kavuluru
ramakanth.kavuluru@uky.edu
Division of Biomedical Informatics (Internal Medicine)
University of Kentucky
Lexington, Kentucky, USA

ABSTRACT

Named entity recognition (NER) and normalization (EN) form an indispensable first step to many biomedical natural language processing applications. In biomedical information science, recognizing entities (e.g., genes, diseases, or drugs) and normalizing them to concepts in standard terminologies or thesauri (e.g., Entrez, ICD-10, or RxNorm) is crucial for identifying more informative relations among them that drive disease etiology, progression, and treatment. In this effort we pursue two high level strategies to improve biomedical ER and EN. The first is to decouple standard entity encoding tags (e.g., "B-Drug" for the beginning of a drug) into type tags (e.g., "Drug") and positional tags (e.g., "B"). A second strategy is to use additional counterfactual training examples to handle the issue of models learning spurious correlations between surrounding context and normalized concepts in training data. We conduct elaborate experiments using the MedMentions dataset, the largest dataset of its kind for ER and EN in biomedicine. We find that our first strategy performs better in entity normalization when compared with the standard coding scheme. The second data augmentation strategy uniformly improves performance in span detection, typing, and normalization. The gains from counterfactual examples are more prominent when evaluating in zero-shot settings, for concepts that have never been encountered during training.

KEYWORDS

named entity recognition, entity normalization, information extraction, deep neural networks, biomedical natural language processing

ACM Reference Format:
Jiho Noh and Ramakanth Kavuluru. 2021. Joint Learning for Biomedical NER and Entity Normalization: Encoding Schemes, Counterfactual Examples, and Zero-Shot Evaluation. In *12th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics (BCB '21), August 1–4, 2021, Gainesville, FL, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459930.3469533

BCB '21, August 1–4, 2021, Gainesville, FL, USA

ACM ISBN 978-1-4503-8450-6/21/08…$15.00
https://doi.org/10.1145/3459930.3469533

1 INTRODUCTION

Biomedical information extraction (BIE) from free text is at the heart of many downstream biomedical natural language processing (BioNLP) applications including knowledge discovery, search systems, and question answering (QA) models. Niche applications such as automatic clinical cohort selection and evidence based medicine through patient similarity computing may also rely on the output of BIE systems. Given an input text (sentence or paragraph), at a high level BIE consists of two mains steps: (1). spotting biomedical entities (e.g., genes, diseases, and drugs) in text and linking them to standardized concepts in ontologies, terminologies, or other thesauri (e.g., Entrez, ICD-10, and RxNorm). (2). identifying any relations between concepts identified in step (1) as asserted in the text. Once these inter-concept relations are identified, they can be stored in structured databases as knowledge graphs. As more and more concepts and inter-concept relations are being discussed in scientific literature, clinical text, and even social media these days, BIE is the only scalable way of curating relational information being presented in textual data. There are obvious caveats regarding BIE methods given any NLP method has associated accuracy issues. However, if the same relation is obtained from multiple research articles, risks associated with imperfect methods can be alleviated.

1.1 Components of NER and EN

Our current effort concerns step (1) of BIE discussed in the previous paragraph. This step is actually composed of two different but related tasks:

(1) First is to identify spans of text in the input representing an entity of interest. This means determining the exact location where the span starts and ends (via character offsets) and then assigning an entity type (e.g., drug, disease, or gene) that typically comes from a set of predetermined fixed types. This *mention detection* and *entity typing* subtasks are together typically called named entity recognition (NER or just ER).
(2) Often the same entity is referred by multiple aliases (text strings) that essentially point to the same biomedical concept. Abbreviations or other short forms and synonyms are obvious sources of aliases. Mapping equivalent aliases to a unique biomedical entity, concept, or code in a standard terminology (e.g., RxNorm, NDC, Multum, Micromedex for drugs) is often called entity normalization (EN), entity linking, or concept mapping.

To summarize, NER consists of mention detection (MD) and entity typing (ET); and EN consists of mapping the span detected to an

actual concept in a standardized terminology. We note MD and ET go together in the sense that identifying a span as representing an entity without figuring out the type of such an entity is mostly not meaningful. The EN step is also crucial because just knowing something is a drug may not be enough and any downstream task can only operate in a concrete way if we also identify the exact drug, by mapping to a standard terminology. Henceforth we refer to the standardized entities (unique codes in a terminology) as *concepts* and typed spans of text as just *entities*. As such, our overall task in this paper can be simply stated as identifying entities and mapping them to concepts in an input text.

1.2 Challenges in biomedical NER/EN

Conventionally, BIE systems handle the NER and EN tasks independently and sequentially in a pipeline setup. That is, entity mentions (spans) are detected first along with their types. Subsequently, those spans are mapped to concepts in a terminology. Given the type is already known before EN happens in this setup, one needs to typically look for concepts that satisfy the type constraint set by the outcome of the preceding ER task. A well-known issue of this pipeline approach is the error propagation over the series of tasks. That is, errors made in mention detection or entity typing will automatically snowball to create errors in the EN step. Another missed opportunity in such a pipeline setup is more effective learning of features (and associated weights) that may be shared and tuned more effectively across multiple tasks simultaneously (e.g., using a single objective function or shared parameters). Pipelines inherently involve separate models for each constituent task and hence operate in disparate feature spaces that do not share any predictive signal. Some recent BIE efforts still seem to rely on this pipeline setup. However, multi-task models and joint approaches are also gaining popularity in the general NLP community and more recently in BioNLP too.

Another limiting factor is the target terminology size for the EN task. As the number of concepts increases, it becomes prohibitive to create high coverage training datasets. This is to be expected as manual efforts in biomedicine are more complex needing expert time, when compared with similar tasks in general domains where crowdsourcing is popular. Also, in terms of methodology, the *softmax* computation for the simpler multi-class modeling becomes very expensive with large target concept spaces. For example, the Unified Medical Language System (UMLS) Metathesaurus, one of the commonly used biomedical terminologies for entity annotation tasks, contains over 4.4 million concepts (as of 2020). The MedMentions [15] NER and EN dataset used in our effort is considered the largest resource in biomedicine for this task and has 352,496 annotated mentions; still that only covers $< 1\%$ $(= 34,724)$ of the full UMLS concept space. So on average there are around 10 instances for each of the unique concepts covered in the dataset. But we also notice that the test data split of MedMentions has concepts that are never encountered during training, leading to inevitable zero-shot scenarios. The sparsity of having very few or no training examples for a large portion of the target concept space can lead to overfitting outcomes with complex nonlinear models where spurious correlations between concepts and surrounding textual artifacts are sort of *memorized* by the model .

We handle error snowballing issues with a joint modeling approach that uses both shared parameter spaces and combined objective function. We address the sparsity concerns with two different strategies. The first is to experiment with a decoupled tagging scheme for representing training data for NER where type tags and positional tags are treated separately. The other strategy uses additional counterfactual training examples derived from the original training dataset to break spurious correlations between concepts and contextual artifacts. Next, we provide some related work pertinent to our contribution.

2 RELATED WORK

Before the prevalent use of neural methods for NER, most prior approaches relied on feature engineering with rule-based heuristic decision models. NER features include word-level patterns (e.g., punctuation, presence of different special characters such as digits or capital letters, part-of-speech, or prefixes/suffixes of tokens) [2, 3], list look-up features (e.g., gazetteer, lexicon, or dictionary) [14, 19]. *TaggerOne* [8] which is widely used in biomedical NER utilizes a semi-Markov model with carefully designed NER features such as the ones described earlier. *Dictionary matching*, based on the string-matching methods, was another popular choice [5, 13, 23, 24] for the EN task. The matching scores are computed between a mention and entities in the controlled vocabulary by leveraging the character n-grams or tf-idf features.

Since 2016, researchers have been proposing neural network-based approaches for EN. Kolitsas et al. [7] prepared fixed dense vector representations of concepts using the pre-trained *Word2vec* embeddings, which were then compared with a mention representation. Yamada et al. [28] also proposed a method to jointly learn the embeddings of words and concepts using a knowledge graph; they are then used in the named entity disambiguation process.

Recently, approaches utilizing neural language models for EN are burgeoning. Liu et al. [11] trained a simple neural language model on the next word prediction task, taking the character sequence as inputs. They used this language model for encoding input sequence before passing to an LSTM-CRF framework for the sequence labeling task. Wu et al.'s idea [26], which is conceptually similar to ours, uses the popular BERT transformer [4] to model a concept's representation using its title and short description, whereas our model represents each concept by its alias and categorical entity type. For EN, similarity scores are computed across the concept representations of k nearest neighbors of the mention representation.

Several neural network-based models for MD have also been proposed. A common approach is to consider all possible spans in a document as potential mentions and compute the mention scores using a feed-forward neural network layer [9, 30]. Because of quadratic complexity $(\mathbb{O}(T^2))$ in the number of tokens T, they had to rely on a heuristic rule to prune out certain unlikely mention candidates during both inference and training.

At least two previous neural NER models are evaluated on the MedMentions dataset, which can be considered state-of-the-art approaches as of now. Loureiro et al.'s model [13] uses a pre-trained neural language model (a BERT variant) to encode the input sentence. A BiLSTM-CRF module follows to identify a candidate entity

span. Once the span is specified, the language model's hidden outputs for the span are pooled to construct its contextual representation. This representation provides the contextual matching feature for ET/EN. They use pre-trained categorical entity embeddings (i.e., 21 semantic types and 18,425 CUI entities) for matching. Also, they used SimString [18] to compare the spans to concepts as in dictionary matching. Wiatrak et al.'s model [25] adopts a similar architecture, a pre-trained language model followed by BiLSTM. They explore the use of hierarchical multi-task learning using ER as an auxiliary task, whereby they aim for a joint learning objective similar to ours. Our proposed models differ in how the entity representations are structured and computed. For entity type classification, we do not use a vector space model. Instead, we adopt the multinomial linear classification for entity type in computing the semantic similarity of entities. Consequently, we consider the possibilities of entities being lexically similar but different in the semantic category. Also, our proposed models aim for a joint objectives in a single model by not using additional post-processing steps, such as dictionary matching based on string comparison.

3 HIGH LEVEL STRATEGIES

Before we elaborate on specifics of our models, we describe our strategies to convey high level intuition.

3.1 Decoupled labeling scheme for NER

Sequence tagging problems are ubiquitous in NLP, especially for part-of-speech tagging and NER. Unlike for classification where class labels are assigned to each document, for NER (and other tagging problems), one needs to generate a label for each token in the input. We need a simple way to capture entity spans for NER. To this end, entity spans are typically indicated by tags (one per token) that correspond to the beginning of an entity and the rest of it. Tokens that are not part of any entity typically have a "other" or "outside" tag. There are some variants of this tagging scheme but they all have a set of entity related tags and an "outside" tag. NER training data is represented in this fashion to directly build models that learn to assign tags, which can be easily used to infer entity spans. The most popular one among such tagging schemes is the IOB (Inside-Outside-Beginning) format. The `B-` prefix indicates that the token is the beginning of an entity, and an `I-` prefix indicates that the token is inside an entity span. Other tokens are labeled with the `O` tag. An entity that is represented by a single token can be labeled with either `B-` or `I-` tags depending on the scheme used.

The IOB prefixes are typically combined with the entity type suffixes. For example, the `B-LOC` tag indicates the beginning token of a "location" entity. This is a natural way to model the learning process because tokens that indicate the beginning of location span may have different characteristics (e.g., different casing, prefixes) compared with tokens that represent the inside tokens of such an entity or even the beginning of another entity type, say, a disease. Combining the positional information ("B") and the type information ("LOC") will help the model capture these differences. However, sparsity issues with not enough training examples for a specific position/type combination tag may cause performance issues. The IOB prefix tags are specifically for determining span boundaries, while the following type suffixes (e.g., "LOC") are for entity typing. A sparsity related issue is the possibility of the model learning spurious associations between the lexical units and specific tags. We are not aware of any efforts that decouple MD (IOB tags) with ET (type tags) and hence we explores the use of decoupled IOB prefixes (i.e., B, I, O) compared against the conventional IOB tagging scheme (i.e., B-type*, I-type*, O).

3.2 Counterfactual training examples

Although MedMentions dataset we use provides a large number of annotated examples compared to the previously available datasets, $\sim$ $200k$ is considered "small" for training a language model for NER targeting a terminology of several million unique concepts. To overcome this limitation, we augment the observable examples by creating counterfactual examples as proposed by Zeng et al. [29]. As illustrated in Figure 1, for each sentence example, we randomly choose one of the entity mentions and replace it with another entity that has the same semantic type of the original entity. The motivation of using this method is to eliminate the spurious correlations that may be established in a highly nonlinear model between the entity and its surrounding context.

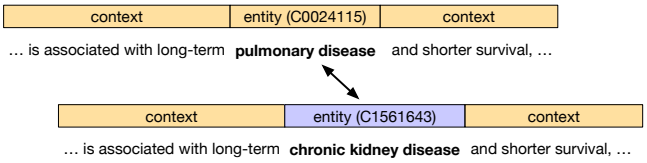

Figure 1: Data augmentation with counterfactual examples

While data augmentation allows us to train the models with more examples, the increased computing needed with the extended name space of new concepts should be carefully managed. We do this with an augmentation factor (ξ), a model hyperparameter, that represents the additional number of counterfactual examples generated per each original example.

4 METHODOLOGY

We denote an input sequence as $\mathcal{X} = (x_1, x_2, \ldots, x_n)$, where x_i is the i-th token of a length n sequence. In our proposed approach, we have three different multiclass tag assignment problems one each for mention detection (MD), entity typing (ET), and entity normalization (EN). The sequence labeling task is to predict a tag y_i for each token x_i where $y_i \in \mathcal{T}_*$, where $*$ can be $\mathcal{IOB}$ for MD, *type* for the semantic type, and *concept* for EN. Here $\mathcal{T}_{IOB} = \{I, O, B\}$ has just three tags indicating mention boundaries. $\mathcal{T}_{type}$ has as many elements as there are semantic types (e.g., disease) and $\mathcal{T}_{concept}$ has as many elements as there are unique concepts in the target terminology. There is also an "other" class for $\mathcal{T}_{type}$ and $\mathcal{T}_{concept}$ for certain tokens that are not part of a named entity and hence not needing a type or concept. All three classification tasks are done at the token-level whereby each token is labeled with $y \in \mathcal{T}_*$. For an identified entity span (via IOB tags), a single entity type in $\mathcal{T}_{type}$ and a unique normalized concept in $\mathcal{T}_{concept}$ are chosen by majority vote across the corresponding per-token type and concept assignments in that span, respectively. A sample gold annotation for MD, ET, and EN are shown in Figure 2 where two unique concepts

from the UMLS are annotated along with their semantic types in parentheses. The "O" tag is used for "other" annotations for type and concept tagging. We note that for the conventional NER scheme where MD and ET are combined, the annotation would naturally combine the IOB tag with the semantic type (e.g., B-t_a for the first word of the sentence).

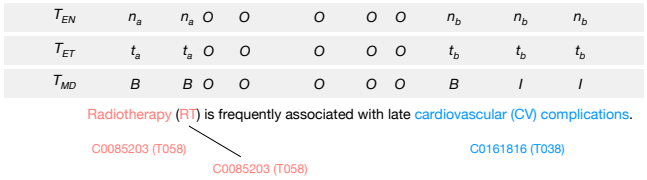

Figure 2: Example annotations for MD, ET, and EN ($n_a = C0085203$, $t_a = T058$, $n_b = C0161816$, $t_b = T038$)

4.1 Models

We devise two different joint neural models: IOBHI and ONETAG (Figure 3). Both architectures use a pre-trained language model, SciBERT [1], for encoding a sentence. To use a transformer based pre-trained language model as the base of a neural architecture is mostly standard practice at this time. Here we use SciBERT which is trained on scientific literature both from biomedicine and computer science. At a high level, we pre-compute vector representations of concepts in the target terminology using their names (different synonymous aliases) and semantic types with the pre-trained SciBERT model (covered in Section 5.1). These vectors are then compared with vector representations of SciBERT hidden outputs for each token in an input sentence. Best matched concept from the target concept space is then chosen for every detected span. As may be expected, additional nuts and bolts level details are incorporated to ensure dimensions are reshaped as needed through feed forward layers as the input is passed through the network. IOBHI and ONETAG differ in details of how these representations are derived. With this basic setup in mind, we will move on to specific details.

In the IOBHI model (left section in Figure 3), we consider the decoupled IOB and type tagging task as discussed in Section 3.1. The IOB classifier is placed at the end of the network such that it can read in the processed name and type representations for identifying mention segmentations. In ONETAG (right section in Figure 3), we use the conventional IOB tagging scheme, where we have $2n + 1$ target classes with n entity types (I and B tags for each semantic type). Note that the two networks are drawn in the same figure for the sake of brevity, but only the left or the right block is active at a time resulting in two different architectures.

Most of the BERT variants use WordPiece tokenization whereby the input sequence is split into subword units, which is expected to enhance the representations of rare words and morphological variations. The use of WordPiece demands additional pre-processing for annotation labels in subword units, which is further explained in Section 5.2. In IOBHI, *name* and *type projection* are dense layers that collect and transform the features from SciBERT's hidden outputs for each token into their representations. For the type representations, we apply the softmax function to obtain the semantic type probability distribution which can be directly compared with the one-hot vectors of the pre-computed name embeddings for semantic type. The name projection layer essentially transforms tokens in a span to the same space as the pre-computed segments of concept names. The concatenated vector of name and type representations are fed to the following biLSTM+CRF [6] for IOB tagging. The biLSTM+CRF module comprises of two bidirectional LSTM layers and a CRF layer whose target tag set size is three for the IOB tags. In the upper section of the figure, the same concatenated vector goes through a fully connected layer (*name matching*) followed by a dot product across the pre-computed concept name embeddings (details in Section 5.1). This process computes the bilinear interactions between the token-level feature representation and the concept name embeddings. Then, we assign the corresponding concept code to each token's normalized name representation using the pre-defined mapping between the concept name indices and actual concept codes.

In the right section of Figure 3, we have the ONETAG model where the biLSTM+CRF module replaces the semantic type classifier, which predicts the IOB-prefixes and entity type simultaneously. Following that is the 1D average pooling layer with kernel size 2 to construct the semantic type probability distribution the same way as in the IOBHI model. The rest of the name projection and bilinear mapping components share the same design. The name ONETAG derives from using a single tag to represent both positional tags (IOB) and semantic types that were decoupled in the IOBHI model.

4.2 Optimization

We take three objective functions in this model as depicted in Figure 3. $\mathcal{L}^{nll}_{MD}$ is the negative log-likelihood (NLL) loss from the CRF layer and $\mathcal{L}^{fl}_{ET}$ and $\mathcal{L}^{fl}_{EN}$ are the focal losses for the tasks of entity typing and normalization, respectively. For the purpose of joint learning, we use the weighted sum of the three objective functions where λ was empirically chosen to 1.0. The joint objectives for training the IOBHI and ONETAG models are

$$\mathcal{L}_{IOBHI}(\mathcal{X}, \mathcal{Y}; \theta) = \mathcal{L}^{nll}_{MD} + \lambda(\mathcal{L}^{fl}_{ET} + \mathcal{L}^{fl}_{EN}) \quad \text{and} \tag{1}$$

$$\mathcal{L}_{ONETAG}(\mathcal{X}, \mathcal{Y}; \theta) = \mathcal{L}^{nll}_{MD/ET} + \lambda\mathcal{L}^{fl}_{EN}. \tag{2}$$

Next, we provide some background and rationale for choosing the focal loss function $\mathcal{L}^{fl}$.

Focal loss [10] is a popular choice for the object detection tasks in computer vision. The motivation of this function is to minimize the gradient norms of easily classified examples (e.g., the pixels of background image in an object detection task). This function is especially effective with class imbalanced data such as in object detection where most of the pixels belong to the non-object class. As shown in Equation (3), the loss depends on the predicted probability distribution $\hat{p}$ on i-th individual sample where γ is a user-defined hyperparameter:

$$\mathcal{L}^{fl}(\mathcal{X}, \mathcal{Y}; \theta) = -\frac{1}{|\mathcal{Y}|} \sum_{y_i \in \mathcal{Y}} (1 - \hat{p}_{i,y_i})^{\gamma} \log \hat{p}_{i,y_i}, \tag{3}$$

where θ are network parameters.

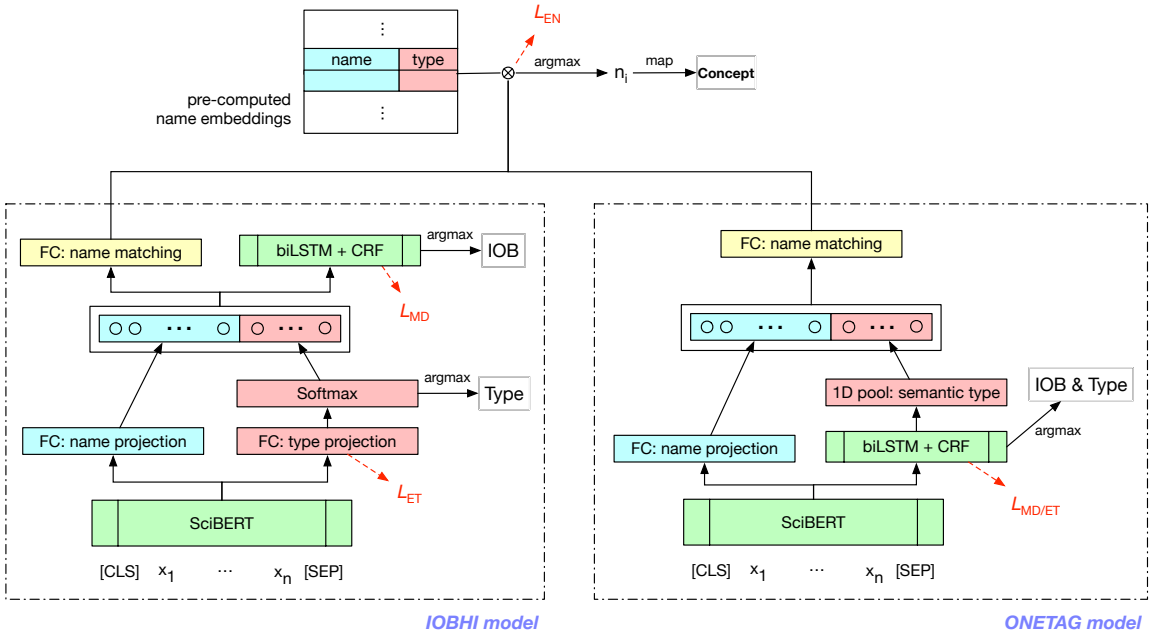

Figure 3: Model design of *IOBHI* (left) and *ONETAG* (right): In *IOBHI*, IOB classification is deferred to the end of the network (FC: fully connected layer, ⊗: dot product; note that IOBHI and ONETAG models are independent but are drawn in one figure for brevity.)

As observed by Mukhoti et al. [16], focal loss forms an upper bound on the regularized KL-divergence between the target distribution q and the predicted distribution $\hat{p}$, where the regulariser is the negative entropy of $\hat{p}$ (proof in [16]):

$$\mathcal{L}^{fl}(\mathcal{X}, \mathcal{Y}; \theta) \geq \mathrm{KL}(q||\hat{p}) - \gamma \mathbb{H}(\hat{p}). \tag{4}$$

The optimization using focal loss, hence, minimizes KL divergence while increasing the entropy of the predicted distribution $\hat{p}$, whereas cross-entropy only minimizes KL divergence. In our case where the majority class is the unlabeled (*other*) class, a prediction with a higher confidence, such as an "obvious" *other* class token, decreases the gradient norms in updating model parameters. As recommended by Mukhoti et al. [16], we choose γ dynamically with the threshold of the predicted probability, such that $\gamma = 5$ if $\hat{p}_{i,y_i} < 0.3$, else $\gamma = 3$.

5 DATA PREPARATION

Recently, Mohan and Li introduced the *MedMentions* dataset with an extensive set of biomedical entity annotations targeting the UMLS concepts. UMLS is the metathesaurus that combines concepts from over 200 medical vocabularies — 4.4 million unique concepts in the 2020 AB release — making it one of the most comprehensive biomedical terminologies. MedMentions provides 352,496 annotated examples from 4,392 PubMed abstracts prepared by human experts in biomedical content curation. The authors of MedMentions selectively chose 21 semantic types of UMLS that are considered most useful for semantic indexing. They created the annotated corpus *MedMentions-ST21pv* with the entities of the 21 semantic types and their descendent types. This corpus contains 203,282 mentions with 25,419 unique concepts from 4,392 documents. In this study, we use *MedMentions-ST21pv* as a benchmark.

5.1 Concept name embeddings

For the EN task, as indicated earlier, we look for the contextually most similar name from the pre-defined concept name embeddings given the encoded name representation. We independently compute these embeddings ahead of training time using the same pre-trained BERT model (i.e., SciBERT). Figure 4 illustrates these pre-processing steps listed as follows:

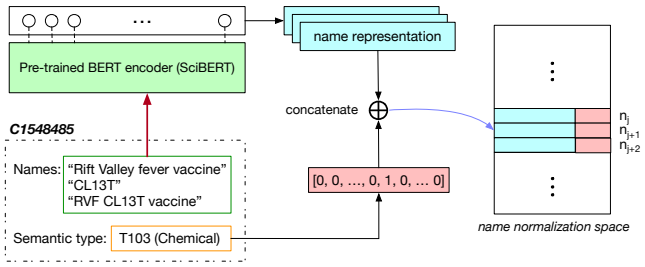

Figure 4: Pre-computed concept name embeddings for EN

(Step 1). We collect names from the UMLS definitions with the following constraints of the UMLS concept properties and add all the aliases mentioned in the MedMentions corpus together.

- the entity belongs to the 21 semantic types (T005, T007, T017, T022, T031, T033, T037, T038, T058, T062, T074, T082, T091, T092, T097, T098, T103, T168, T170, T201, T204) or the descendent types of those.
- LAT (language of term) is EN (English)
- TS (term status) is P (preferred)
- STT (string type) is PF (preferred form)

- SAB (source name) is in {CPT, FMA, GO, HGNC, HPO, ICD10, ICD10CM, ICD9CM, MDR, MSH, MTH, NCBI, NCI, NDDF, NDFRT, OMIM, RXNORM, SNOMEDCT_US}
- SUPRESS (supressible flag) is N (none)

(Step 2). We encode up to three names (typically multi-token phrases) for each concept using the BERT encoder. We take the encoder's last layer outputs and use the mean vectors as the name embeddings. We also append the concept's semantic type representation (i.e., the one-hot vector of the type) in order for the model to consider both the name and semantic type of the entity.

With the specified constraints, we can build 9,538,297 name embeddings for the entire UMLS concept set. For training and testing purposes, we build name embeddings only for the concepts that appear at least once in the MedMentions-ST21pv corpus. During training, we use the training subset with 42,836 name embeddings corresponding to concepts seen in the training examples; this number varies by the data augmentation factor (75,865 with $\xi = 1$, 106,561 with $\xi = 2$). At test time, we use the full set of 56,893 name embeddings for the entire set of concepts in the MedMentions corpus. The methodology for using a large scale name normalization space (such as the full 9.5+ million UMLS embeddings) is out of the scope in this current study.

5.2 Subword-level tokens to word-level labels

BERT models use the WordPiece tokenization [27] method, which is a subword segmentation algorithm. The vocabulary is constructed iteratively from the characters in the language by adding the most frequent combinations of entries in the current vocabulary. In our effort, the sequence labeling is modeled at the word level, which creates a discrepancy with the sequence tokenized using WordPiece given its subword focus. So, we use WordPiece to tokenize text and assign the given label to all the constituent subword tokens. In the inference step, we use the majority vote to determine the label of a word. For example, WordPiece would tokenize the word "hydrocodone" into ('hydro', '##cod', '##one'). If the model predicted the semantic types for the sequence as (T103, T168, T103), then we assign T103 to the full word.

6 EXPERIMENTS

6.1 Datasets and baseline models

We use MedMentions-ST21pv to evaluate the performance of our models on the NER and EN tasks. Table 1 shows the statistics of the dataset. We experiment with the same *train-validation-test* splits (60-20-20%) provided by the creators of the MedMentions datasets.

Table 1: Statistics of the MedMentions-ST21pv dataset

	Training	Dev	Test
# of documents	2,635	878	879
# of mentions	122,241	40,884	40,157
# of unique concepts mentioned	18,520	8,643	8,457

We compare our models with the publicly available biomedical NER tools and some previous efforts. We outline four different models from this effort, given two variables: model design and data augmentation factor.

- **ONETAG**: Model using conventional IOB format (i.e., combined IOB prefixes and type suffixes)
- **ONETAG ($\xi = n$)**: ONETAG trained with data augmentation (n is the augmentation factor)
- **IOBHI**: Model with decoupled IOB format (i.e., separate IOB tags and semantic types)
- **IOBHI ($\xi = n$)**: IOBHI trained with data augmentation

Followings are the tools and the state-of-the-art models used for comparisons:

- *TaggerOne* [8] has been a popular choice for the biomedical NER, which uses carefully designed rule-based algorithms.
- *QuickUMLS* [21] utilizes an approximate dictionary matching algorithm, which outperforms other biomedical text processing tools such as MetaMap and cTAKES.
- *SciSpacy* [17] is a package of specially designed tools for biomedical and scientific text processing leveraging the spaCy library. SciSpacy has shown superior results to QuickUMLS and MetaMap on the biomedical NER tasks.

Due to the recency of the MedMentions release, there are not many end-to-end models for NER and EN on this particular dataset. We identified two recent peer-reviewed efforts with similar evaluation setup as ours for comparison purposes.

- Loureiro et al. [13] presented a model of a BERT-biLSTM-CRF framework with an approximate dictionary matching method.
- Wiatrak et al. [25] proposed a model of a BERT-BiLSTM-MLP framework with a hierarchical structure of multiple tasks.

6.2 Training details

We adopt the SciBERT uncased model [1] as the sentence encoder whose model dimension is 768. The maximum sentence length of inputs is 256. The biLSTM+CRF consists of two layers of biLSTM networks with model dimension 256. All models are optimized using AdamW [12] controlled by a linear scheduler with warmup steps. The learning rate starts with 0, increases up to 3×10^{-5} during the warmup steps, and linearly decreases to 0 until the specified number of training steps. We apply dropout to biLSTM hidden states with a rate of 0.1. Training is done for 8 epochs with the batch size of 8.

6.3 Evaluation metrics

Whereas the objective functions are computed at the token level during the training and validation steps, we measure the model's performance using the mention-level metrics. Following the well-known CONLL conventions for NER [20], we use the *exact-match evaluation* system, where the metrics are micro-averaged strict precision, recall, and F1 scores. That is, a predicted text span (MD step) is a true positive (*tp*) only if the starting position and the length exactly match the ground truth. For a predicted concept (or its type) to be a *tp*, its span should be an exact match with the corresponding class of the ground truth. Hence, the semantic type classification and entity normalization performances are upper-bounded by the performance of mention detection. The constraints

Table 2: MD, ET, and EN performances on MedMentions-ST21pv dataset. The results marked with † are obtained from [13].

	Mention Detection (MD)			Entity Typing (ET)			Entity Norm. (EN)		
Model	P	R	F_1	P	R	F_1	P	R	F_1
TaggerOne	n/a	n/a	n/a	n/a	n/a	n/a	0.471	0.436	0.453
QuickUMLS†	n/a	n/a	n/a	0.145	0.169	0.156	0.180	0.261	0.213
ScispaCy†	n/a	n/a	n/a	0.101	0.317	0.154	0.252	0.535	0.342
Loureiro et al.'s (CLF)	0.694	0.718	**0.706**	0.586	0.646	0.615	0.322	0.527	0.400
Loureiro et al.'s (STR_CLF+)	0.694	0.718	0.706	0.631	0.637	**0.634**	0.484	0.501	0.492
Wiatrak et al.'s	0.742	0.593	0.659	0.594	0.553	0.573	0.431	0.401	0.415
ONETAG	0.709	0.671	0.690	0.636	0.602	0.619	0.503	0.476	0.489
ONETAG ($\xi = 1$)	0.701	0.679	0.690	0.625	0.605	0.615	0.509	0.493	0.501
ONETAG ($\xi = 2$)	0.696	0.674	0.685	0.620	0.601	0.611	0.512	0.496	0.504
IOBHI	0.701	0.682	0.691	0.614	0.597	0.605	0.500	0.487	0.494
IOBHI ($\xi = 1$)	0.706	0.675	0.690	0.620	0.593	0.606	0.522	0.499	0.510
IOBHI ($\xi = 2$)	0.705	0.673	0.689	0.617	0.589	0.602	0.524	0.499	**0.511**

Table 3: Zero-shot evaluation of IOBHI and ONTAG models for NER and EN

	Mention Detection (MD)			Entity Typing (ET)			Entity Norm. (EN)		
Model	P	R	F_1	P	R	F_1	P	R	F_1
ONETAG	0.877	0.671	0.760	0.707	0.541	0.613	0.531	0.406	0.460
ONETAG ($\xi = 1$)	0.866	0.694	0.770	0.695	0.557	0.618	0.581	0.466	0.517
ONETAG ($\xi = 2$)	0.864	0.694	0.770	0.697	0.560	**0.621**	0.617	0.495	0.549
IOBHI	0.868	0.688	0.768	0.685	0.542	0.605	0.545	0.432	0.482
IOBHI ($\xi = 1$)	0.861	0.692	0.767	0.677	0.544	0.603	0.605	0.485	0.539
IOBHI ($\xi = 2$)	0.868	0.699	**0.775**	0.679	0.547	0.606	0.624	0.493	**0.551**

are highest for the final EN step because the concept code, its type, and exact span in the input text ought to match the ground truth for us to consider it a *tp*.

To clarify our counting system, we define metrics

$$\text{precision} = \frac{\#tp}{\#tp + \#fp} \quad \textit{and} \quad \text{recall} = \frac{\#tp}{\#tp + \#fn},$$

where $(\#tp + \#fp)$ is the number of predictions and $(\#tp + \#fn)$ is the number of ground truth occurrences using the counts of false positives (*fp*) and false negatives (*fn*). To hold these properties, we consider the cases where the model predicts correctly-bounded text spans with wrong labels as both *false positives* and *false negatives* (e.g., a mention that should have been labeled as "A" but was predicted as "B" is a *fn* for "A" and a *fp* for "B"). In the zero-shot evaluation setup (details in Section 6.4.1), we do not consider a prediction as an *fp* if it overlaps ground truth mentions of concepts seen in the training dataset; this is a standard practice when assessing zero-shot capabilities. The model should identify the mentions regardless of whether the classes are seen or unseen in any dataset.

6.4 Results and discussion

Table 2 shows our models' performance against previous models and the biomedical NER tools on three different tasks: mention detection (MD), entity typing (ET, UMLS semantic type classification), and entity normalization (EN, UMLS concept code normalization) task. Our models outperform previous results on the end-to-end strictest EN task. But when restricted to the less strict tasks, MD and ET, the Loureiro et al. result seems superior. However, the ET objective of their models is inherently different from ours. Their models require pre-computed embeddings for entity types, which are then used exclusively for the ET task. This method prevents the models from learning features that may be shared in both ET and EN tasks. We can compensate the marginal loss of ET performance with a significant improvement in EN results. Also, given that the Loureiro et al.'s *STR_CLF+* model is an ensemble method using string comparison for approximate dictionary matching, we believe that another of their proposed models (*CLF*, row four of Table 2) is more suitable for evaluating the end-to-end neural approaches against our models.

In particular, for the ET task, we see ONETAG models perform better than IOBHI models. We conclude that if the eventual goal is

just NER (that is, MD and ET), the conventional tagging scheme is superior. Given there are only 21 semantic types in the dataset, with over 122K mentions in the training dataset, there could be enough signal for the combined tags (position plus type) to render the models effective for NER compared with the decoupled setup. However, when it comes to EN performance, the decoupled approach (IOBHI) that delegates IOB tagging to the end performs better than the conventional ONETAG approach. IOBHI uses type prediction before matching with pre-computed concept name embeddings and this type-only signal (without the IOB hints) seems to better help the model match correct concepts for the EN part. On the other hand, the IOB tagging in the IOBHI model performs well enough to spot the boundaries to an extent that renders the end-to-end EN evaluation superior for the overall architecture.

All our models achieved higher scores than the previous efforts on the EN task. The consistent increase in F1 score for the EN task with models using data augmentation supports its efficacy of providing counterfactual examples that break spurious correlations of concepts with surrounding textual context. However, the gains by increasing the augmentation factor from 1 to 2 is minimal. We deduce that using an adequate amount of augmented examples improves learning for entity normalization but not for other tasks.

6.4.1 Zero-shot evaluation . We also wanted to analyze what happens to our performance metrics when we look at zero-shot (ZS) scenarios. That is, what happens if we evaluate performances over those concepts that only occur in the test set and never show up in the training and development subsets. We find that there are 3,247 such unique concepts and 8,180 mentions of them in the test set. These ZS concepts account for 38% of all concepts in the test set (but only around 20% of all test mentions, which is reasonable since these concepts are expected to be rare).

Table 3 outlines the model performance in the ZS setting for concepts that exclusively occur only in the test dataset. We see similar patterns to the overall results (from Table 2) on the test set: (1). ONETAG models perform better on ET, and IOBHI models perform better on EN, (2). the data augmentation techniques enhance the performance on EN but not others. It is important to recall that in this setting an *fp* can only arise out of mistakes made for ZS concepts. That is, an *fp* for a predicted ZS concept occurs if (a). either its type or span is incorrect or (b). the ground truth concept is a different ZS concept. Note that the ZS concept set size is relatively smaller (only 20% of test set mentions) and the universe of possibilities for false positives for a ZS concept arises out of only other ZS concepts. Hence, the precision values in the ZS setting in Table 3 are higher than those in the larger full test dataset (Table 2). Recall values for MD and EN are similar to general results but ET recall is markedly lower compared to general results[1]. We also observe that counterfactual data augmentation only increases F1-score by around 2% in overall results but in the ZS settings the gains are from 7–9%. This is not surprising since breaking the spurious correlations between concepts and surrounding contexts during training is expected to help with ZS concepts that were never encountered in that process. Intuitively, we suspect, without augmentation some ZS concepts would be mismatched to potentially similar concepts seen during training.

6.4.2 Probing for semantic type affinities using the name matching layer parameters. Recall that a concept name embedding in our model is the concatenated vector of a dense vector from the name projection layer and a softmax probability distribution from the type projection layer. The following name matching layer computes the similarity scores with the pre-computed concept name embeddings, which are constructed in the same way via SciBERT. We deliberately designed the structure of a concept's name embedding in this manner to incorporate features corresponding to its name and also its type (in an explicit manner). This allows us to analyze the model regarding how it interprets the semantic type features from the hidden outputs on the EN task.

We particularly look at the parameter matrix of the name matching fully connected (FC) component (yellow boxes in Figure 3). Let the vector $u = \langle u_n, u_t \rangle$ be the hidden output from the model, which is the input vector of the FC component, expressed as a concatenation of vectors u_n and u_t, where u_n is for the name space in $\mathbb{R}^p$ and u_t is for the entity type space in $\mathbb{R}^q$. Let $v = \langle v_n, v_t \rangle$ be the pre-computed concept name embedding with the same structure as u. With W as the bilinear transformation function of the FC component, for u and v as defined earlier, we have $sim(u, v) = vWu$. Let's denote $\widetilde{u}$ be Wu, the transformed vector of u. Thus we rearrange the similarity measure of an input token and a pre-computed concept name embedding as the dot product of v and $\widetilde{u}$:

$$v \cdot \widetilde{u} = \sum_{k=1}^{p+q} v_k \widetilde{u}_k = \sum_{k=1}^{p} v_k \widetilde{u}_k + \sum_{k=p+1}^{p+q} v_k \widetilde{u}_k \tag{5}$$

$$= \sum_{k=1}^{p} \sum_{i=1}^{p+q} v_k W_{k,i} u_i + \sum_{k=p+1}^{p+q} \sum_{i=1}^{p+q} v_k W_{k,i} u_i \tag{6}$$

The second term in equation 6 clearly influences the type segment similarity in the end and is parametrized by a submatrix of W, specifically, $W_{type} = W[p+1, p+q; 1, p+q]$. Thus, the multiplication of W_{type} and its transpose gives us the affinity matrix among entity type representations that the model learned. Figure 5 is the cluster map of the 21 semantic types obtained from the correlation matrix $W_{type}(W_{type})^\top$ where rectangular blocks of (shades of) red indicating clusters; this was derived from hierarchical clustering using the *Scipy* [22] cluster package. Below we display some of the interesting type clusters our probing has surfaced.

- Cluster 1: T058: Health Care Activity, T091: Biomedical Occupation or Discipline
- Cluster 2: T092: Organization, T062: Research Activity, T170: Intellectual Product
- Cluster 3: T097: Professional or Occupational Group, T098: Population Group
- Cluster 4: T005: Virus, T007: Bacterium, T204: Eukaryote
- Cluster 6: T082: Spatial Concept, T017: Anatomical Structure, T022: Body System

These clusters appear to indicate that W has nicely converged to capture similarities among types. For example Cluster 4 seems to be grouping different types of organisms and Cluster 6 appears to capture anatomical locations. This probing provides a peek into the

[1]Please note that type prediction is independent of concept prediction although parameter sharing for the two tasks is in place. So this possibility of differences in recall is plausible although not expected.

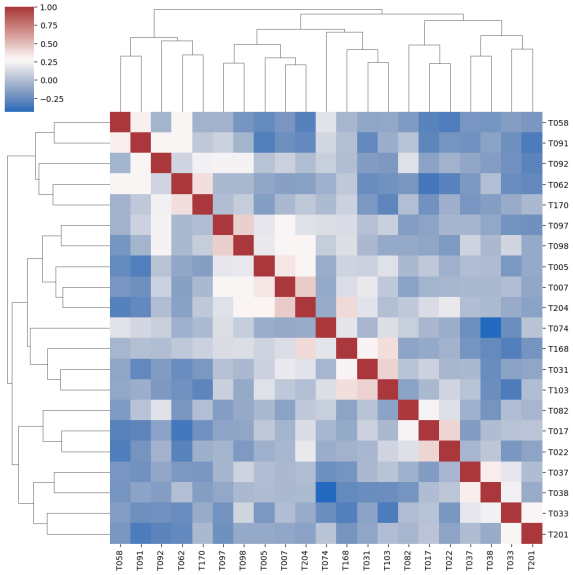

Figure 5: Type affinity matrix derived from the *name matching* layer's bilinear function

inner workings of what the model is learning and provides additional confidence that it is teasing out reasonable representations of concepts.

7 ERROR ANALYSIS

We manually analyzed randomly selected error causing instances to track different types of errors. We highlight a few high level classes of errors we found. We first focus on partial matches of spans that cause both *fp* and *fn* errors.

- Adjective or noun compound modifiers that our models predicted to be part of a span turned out to be incorrect in several error causing examples. The following examples contain *italicized* ground truth spans and the full and incorrect spans we predicted.
  - activating *mutations*
  - benign *parathyroid adenoma*
  - familial isolated *hyperparathyroidism*
  - leptin *gene promoters*
  - ZFX *oncogenes*

  Considering these examples, it does not appear that our spans are blatantly wrong because the adjectives and compounds we included in the predictions (e.g, benign, familial, leptin) seem pertinent to the ground truth spans. They appear to bring about more specificity to the concepts being tagged compared with ground truth annotations. We are not sure if these are errors in the MedMentions dataset or if this is really a nuanced phenomenon that our models are unable to capture.
- While the earlier examples indicated that we erroneously made spans more specific, we also encountered errors where less specific spans are somehow mapped to more specific concepts in MedMentions. Consider these examples:
  - (enzyme) *activity*
  - (bacterial culture) *medium*
  - *benign* (thyroid) *nodules*
  - *clinical* . . . findings
  - *persistent* . . . asthma

  The first three examples show the italicized spans in the ground truth but were mapped at the EN level to ground truth concepts that are more specific (as indicated with the corresponding full preferred name elements in parentheses). In the last two examples, terms that render more preciseness to concepts (e.g., findings, asthma) were actually present in the context but appeared with a gap (that included other tokens) from the ground truth spans. Maybe in these cases, during the EN task, our models were unable to latch on to the implied/latent signal present in the surrounding context.

We also accrued several errors when we missed or erroneously tagged broad themed concepts. For example, concepts with preferred names *men, women, results, predictors, group, trials* are sometimes mapped to specific concepts and sometimes not in the MedMentions dataset. The models had trouble figuring out which contexts warrant a mapping. Additionally *fp* errors also occurred with several abbreviations were deemed incorrectly mapped as per ground truth but seemed appropriate upon manual examination. Our analysis revealed that at times, only the first occurrence of an abbreviation was annotated in the ground truth with some subsequent mentions left untagged. This particular scenario does not seem to be an outcome of model's issues but due to inconsistencies in MedMentions test set.

8 CONCLUSION

In this effort, we evaluated two high level strategies in the context of a multi-task learning framework for named entity recognition and entity normalization for biomedicine. First, we explored the effect of decoupling IOB-prefixes from the type tags in the combined conventional tagging scheme. Results show that separating the task of identifying the boundaries of mentions from the entity type classification enhance the entity normalization performance but not for entity typing and mention detection. We also demonstrate that using an adequate number of counterfactual training examples helps in the EN task, more so in the zero-shot evaluation setting. Parameter probing showed meaningful clusters of semantic types and error analyses surfaced interesting issues that warrant deeper exploration of the MedMentions dataset and more advanced strategies that better exploit the context.

The focus of this effort was to assess specific strategies in the context of a joint modeling framework for biomedical NER and EN. Hence, we kept the model design fairly simple without resorting to more sophisticated methods such as transfer learning or domain adaptation. We did, however, leverage latest pre-trained language models (SciBERT) for biomedicine. More innovative methods to better capture context and implied intent of the writers are necessary to make additional progress. Our effort only uses concept aliases and semantic types for the EN task; well established knowledge bases that include explicit relationships among concepts or topic distributions across documents can be utilized in future efforts in NER and EN. We have not fully addressed the scalability aspect in

this effort. Although zero-shot performance is decent in the MedMentions name space, more realistic systems would need to search in the target space of 4.4 million UMLS concepts and 11 million corresponding English names. This problem demands parallel and distributed deep learning techniques and very fast nearest neighbor search (e.g., locality-sensitive hashing), aspects we intend to explore in the near future.

ACKNOWLEDGEMENTS

Research reported in this publication was supported by the National Library Of Medicine of the National Institutes of Health under Award Number R01LM013240. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

REFERENCES

[1] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3606–3611, 2019.

[2] Eckhard Bick. A named entity recognizer for danish. In *Proceedings of the 4th International Conference on Language Resources and Evaluation, LREC2000*, pages 305–308, 2004.

[3] Michael Collins. Ranking algorithms for named entity extraction: Boosting and the votedperceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 489–496, 2002.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[5] Jörg Hakenberg, Martin Gerner, Maximilian Haeussler, Illés Solt, Conrad Plake, Michael Schroeder, Graciela Gonzalez, Goran Nenadic, and Casey M Bergman. The gnat library for local and remote gene mention normalization. *Bioinformatics*, 27(19):2769–2771, 2011.

[6] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[7] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, 2018.

[8] R Leaman and Z Lu. TaggerOne: joint named entity recognition and normalization with semi-markov models. *Bioinformatics (Oxford, England)*, 32(18):2839–2846, 2016.

[9] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, 2017.

[10] T Lin, P Goyal, R Girshick, K He, and P Dollar. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017.

[11] Liyuan Liu, Jingbo Shang, Xiang Ren, Frank Xu, Huan Gui, Jian Peng, and Jiawei Han. Empower sequence labeling with task-aware neural language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[13] Daniel Loureiro and Alípio Mário Jorge. MedLinker: Medical entity linking with neural representations and dictionary matching. In *European Conference on Information Retrieval*, pages 230–237. Springer, 2020.

[14] David D McDonald. Internal and external evidence in the identification and semantic categorization of proper names. In *Acquisition of Lexical Knowledge from Text*, 1993.

[15] Sunil Mohan and Donghui Li. Medmentions: A large biomedical corpus annotated with umls concepts. In *Automated Knowledge Base Construction (AKBC)*, 2018.

[16] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip HS Torr, and Puneet K Dokania. Calibrating deep neural networks using focal loss. *arXiv preprint arXiv:2002.09437*, 2020.

[17] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. ScispaCy: Fast and robust models for biomedical natural language processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, 2019.

[18] Naoaki Okazaki and Jun'ichi Tsujii. Simple and efficient algorithm for approximate dictionary matching. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 851–859, 2010.

[19] Lisa F Rau. Extracting company names from text. In *Proceedings the Seventh IEEE Conference on Artificial Intelligence Application*, pages 29–30. IEEE Computer Society, 1991.

[20] Erik Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[21] Luca Soldaini and Nazli Goharian. QuickUMLS: a fast, unsupervised approach for medical concept extraction. In *Proceedings of the MedIR Workshop at SIGIR 2016*.

[22] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[23] Xi Wang, Jiagao Lyu, Li Dong, and Ke Xu. Multitask learning for biomedical named entity recognition with cross-sharing structure. *BMC bioinformatics*, 20 (1):427, 2019.

[24] Chih-Hsuan Wei and Hung-Yu Kao. Cross-species gene normalization by species inference. *BMC bioinformatics*, 12(S8):S5, 2011.

[25] Maciej Wiatrak and Juha Iso-Sipila. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 12–17, 2020.

[26] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. Scalable zero-shot entity linking with dense entity retrieval. *arXiv preprint arXiv:1911.03814*, 2019.

[27] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[28] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. Joint learning of the embedding of words and entities for named entity disambiguation. In *20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016*, pages 250–259. Association for Computational Linguistics (ACL), 2016.

[29] Xiangji Zeng, Yunliang Li, Yuchen Zhai, and Yin Zhang. Counterfactual generator: A weakly-supervised method for named entity recognition. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7270–7280, 2020.

[30] Rui Zhang, Cicero dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir Radev. Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107, 2018.