入れ子になっている固有表現に対する Distant Supervision

芝原 隆善 [1,2], 山田 育矢 [2,3], 西田 典起 [2], Shanshan Liu[2], 古崎 晃司 [2,4], 渡辺 太郎 [1], 松本 裕治 [2]

[1] 奈良先端科学技術大学院大学 {shibahara.takayoshi.sk4, taro}@is.naist.jp
[2] 理化学研究所 {takayoshi.shibahara, ikuya.yamada, noriki.nishida, shanshan.liu, kouji.kozaki, yuji.matsumoto}@riken.jp
[3]Studio Ousia ikuya@ousia.jp
[4] 大阪電気通信大学 kozaki@osakac.ac.jp

1 はじめに

本研究では Nested NER と呼ばれるタスクに取り組む。Nested NER とは、入れ子になっている語句をも対象にする固有表現抽出のタスクである。このタスクでは例えば、“. . . in the human immunoglobulin heavy-chain gene enhancer .” という文を入力として、“human immunoglobulin heavy-chain” が protein であり “human immunoglobulin heavy-chain gene enhancer” が DNA であることを特定する。このタスクは近年様々な手法で取り組まれており、特にスパンを全て列挙して分類する枠組みで取り組めることから、多くのスパン分類のモデルでの研究が行われている [1, 2, 3, 4, 5]。

一方で入れ子でない固有表現抽出に対して、従来からそのアノテーションコストが問題視されてきた。そこで既知の語句集合: 辞書 を利用した擬似データ作成に基づく、教師なし設定の固有表現抽出: Distant Supervision NER が行われてきた。Distant Supervision NER には 辞書拡張を活用する手法 [6] 、Self-training や Partial CRF などの辞書マッチを正しいとしてそのラベルを伝搬させる手法 [7, 8] など様々な手法が提案されてきたが、辞書拡張を活用する手法の中には Nested NER のようにスパン分類モデルを利用する手法 [9] も知られている。

本研究では Nested NER に対して Distant Supervision の設定で問題に取り組む。つまり、辞書マッチによる擬似データ生成と教師あり学習による手法で、入れ子の固有表現を抽出する。この際、本研究では上記の二つの研究で利用されているスパン分類モデルを利用する。

本研究では GENIA [10] コーパスで評価を行った。辞書マッチによる F 値は 25 % とかなり低かったが、POS タグに基づく Chunker を利用したことで、39 % ほどの F 値へと改善した。同じ Chunker を活用することで、Distant Supervision において 48 % の精度が得られた。その一方で Distant Supervision による結果は、教師あり設定の 76 % と比べて 28 % 低い結果となった。

2 手法

2.1 辞書マッチによる擬似データ作成

今回の擬似データ作成ではカテゴリごとに分けられた語句: 辞書 を活用する。文中に現れる辞書 [1)] の語に対して、出現したスパンと辞書中でのカテゴリを擬似教師データとして活用する。この辞書マッチの際に重複を許すことで入れ子の固有表現を考慮できるようにした。[2)]辞書に含まれるものはそのカテゴリに分類し、辞書に含まれないスパンは “Others” というカテゴリに分類する。この “Others” カテゴリはテスト時には無視されるカテゴリである。

今回取り扱う辞書には タンパク質名の “Beta1-Tubulin” と遺伝子名の “beta1-tubulin” など 大文字・小文字 の差異で区別される語句が存在した。そのような事例に対して precision を上げ、かつ全体としては recall を向上させるために、このような文字種によって区別される複数の語句が存在する時には大文字と小文字を区別する辞書マッチを行い、そうでない場合には大文字と小文字を区別しない辞書マッチを行った。

1） より具体的には付録 B を参考されたい。
2） 複数のスパンの一部が互いに重なる場合も許容した。

2.1.1 Chunker を利用した辞書マッチ

辞書マッチを単純に行うだけでは、しばしばスパンを短く取ってしまうという問題が生じた。例えば、図 1 で示された例では、辞書によって “HB24” と “mRNA” がマッチしているが、正解のスパンは “HB24 mRNA” だけである。そこで、まず “HB24 mRNA” というスパンを Chunker によって獲得し、このスパンの末端に位置する語句 (“mRNA”) の辞書での分類 (RNA) を利用して、スパンのラベル付を行った。このような Chunker の利用は Disntant Supervision NER において以前から行われいる方法である [11, 9] 。今回は scispaCy [12] を利用して得られた POS タグを活用して、正規表現を利用した Chunker [3] を作成した。この Chunker は入れ子の Chunk を予測できない。そのため、入れ子の疑似データを作成できないが、入れ子の構造を考慮した Chunker は今後の研究課題とする

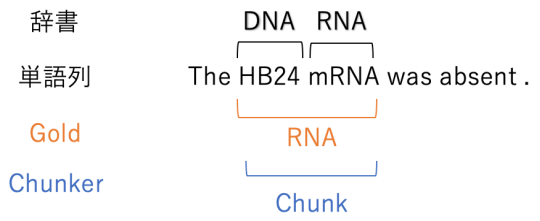

図 1　Chunker を利用した辞書マッチの例

2.1.2 Others の Undersampling

上記の辞書マッチの手順から明らかな通り、辞書に含まれないスパンに相当する “Others” カテゴリの個数が多くなってしまい、うまく学習ができないことが想定される。そこで “Others” 以外のカテゴリと “Others” が同じ頻度になるように “Others” のラベルをランダムにサンプリングした。

2.2 学習に利用したスパン分類モデル

今回利用するスパン分類モデルは [13] のモデルを参考にしている。特徴量抽出部分は BioBERT [14] を活用し、それぞれのスパンに対して、スパンのはじめと終わりの位置の最終隠れ層のベクトルを連結して利用する。この際に全ての[4]スパンを列挙して分類することで、入れ子になっている固有表現に対応する。分類器部分は Dropout、線形変換及び softmax を利用し、それぞれのスパンに対して確率値を計算する。それぞれのスパンに対する確率値から擬似正解との相互エントロピーをロスとして利用する。

事前実験によってこのスパン分類モデルが長いスパンを予測しがちであることがわかった。そのため、POS タグに基づく Chunker [3] の予測する Chunk の範囲内に絞って学習と予測を行った。つまりこの Chunk を最大幅とするスパン内部においてのみ入れ子構造を予測した。

3 データセット

3.1 擬似データ作成に利用した資源

疑似データの作成のため、辞書として UMLS を、辞書マッチの対象となるラベルなしコーパスとして PubMed を利用した。UMLS は 2020AA のバージョンを利用し、総計 290,706 個の語句からなる辞書 [1] を活用した。また、PubMed からは 100,000 文を利用し、train と development に 8 対 2 の割合で分割して利用した。

3.2 評価データセット

評価データセットとして Nested NER の先行研究でよく利用されている GENIA データセット [10] を利用した。GENIA データセットには並列句など複数の語句が含まれるスパンに対するアノテーションが振られている。Nested NER の先行研究では、このような並列句を含んだスパンを問題の対象に含めるものもあるが、本研究ではこれらのスパンを対象外とした。BioBERT に入力する前に単語分割するために scispaCy [12] を利用した。

GENIA データセットの train/development/test の各データ分割は、15,024、1,670、1,855 文である。教師あり設定の場合はこれらのすべてを利用し、Distant Supervision 設定の場合は test データのみ利用している。つまり Distant Supervision 設定では学習の際に擬似データのみしか利用しておらず、評価の際にのみアノテーションデータを利用している。GENIA データセットにおけるそれぞれのカテゴリの固有表現の数は表 1 の通りである。

3) この Chunker についてより詳しくは 付録の POS に基づく Chunker を参考されたい。
4) 実装上はスパンの最大長を設定している

カテゴリ	固有表現数
protein	34,097
DNA	9,933
RNA	933
cell type	6,987
cell line	3,790

表 1 GENIA データセットにおける各カテゴリの固有表現数

4 実験と結果

4.1 擬似データ作成

表 2 には擬似データ作成手順による精度を GENIA コーパス [10] のテストデータに対して評価した結果を示した。“nest inside prediction” の列には予測されたスパンのうち他のスパンの内側に入っているスパンの個数を示している。“nest inside R.” が入れ子の内側となっているスパン、“nest outside R.” が入れ子の外側になっているスパンに対する recall の値が計算されている。“Dictionary Match” は辞書マッチによるスパン同定とクラス分類を意味する。次に “Chunker + End Match” は POS タグを利用した Chunker によるスパン同定と、そのスパンの末端語に対して辞書を利用したクラス分類である。次の “Oracle Span + End Match” は、スパン同定には正解のデータを利用し、クラス分類にはスパンの末端語句の辞書マッチを利用した手法であり、“Gold” は正解のテストデータから分かる数値を記載している。

4.2 Distant Supervision

表 3 は Distant Supervision を適用した時の精度について述べている。この表 3 は表 2 と同様に、テストデータにおける固有表現抽出の精度と予測における入れ子の固有表現の数と、入れ子の内側の正解数を示している。左側の Model 部分には手法名を示している。上から大きく、擬似データ作成手法、Distant Supervision, 教師あり手法 の精度を表している。“Chunker + End Match” は擬似データ作成手法であり、POS タグベースの Chunker によるスパン同定とスパン末端語句によるスパン分類から構成されている。これは表 2 の “Chunker + End Match” と同じものである。“BioBERT (Distantly Supervised)” は “Chunker + End Match” の手法で作成した擬似データを元にスパン分類モデルを適用したものである。“+ Chunker” は “BioBERT (Distantly Supervised)” に追加して、訓練と予測の対象とするスパンを POS に基づいた Chunker で取得される Chunk の内部に限定したものである。“BioBERT (Supervised)” は “BioBERT (Distantly Supervised)” と同じスパン分類モデルを教師ありデータで学習したモデルを意味している。

5 考察

5.1 擬似データ作成

表 2 の結果から、辞書だけだと精度が出ず、Chunker を利用すると精度が改善することがわかる。つまり、辞書の示すスパンとテストデータにおけるスパンが一致していないことがわかる。また、正解データのスパンを利用した結果とも乖離があることから、この POS タグを利用した Chunker はスパン同定を完全にはできていないということがわかる。

入れ子の予測観点では、単純に辞書マッチをした場合に、Gold データよりも入れ子になることが多いということがわかった。POS タグベースの Chunker を利用した場合、基本的には外側のスパンを取るような設計になっている。そのため一部の例外的な事例を除き入れ子の予測はなく、入れ子の予測事例は 10 となっている。Oracle のスパンを利用した際には 241 個の内側のスパンしか取れておらず、Gold のものに比べて約 40 % ほどであり、辞書の被覆率の小ささが影響していると考えられる。

これらの実際の入れ子部分に対する予測精度を見てみるとデータセット全体での精度と比べて大きな違いがないことが全ての手法で見て取れる。ただし全ての場合で、外側のスパンを特定することが容易であることが分かる。基本的に外側のスパンを取得するように設計された Chunker に基づいているので、“Chunker + End Match” では入れ子の内外の精度に比較的大きなスコア差があることが見て取れる。この Chunker は外側の一番大きなスパンを取ることを目的とした正規表現に基づいているものの、例外的に一部入れ子のスパンを出力している。

5.1.1 エラー分析: 曖昧なスパン

エラー分析を行ったところ、スパン境界が曖昧と見える事例がいくつか存在した。特に修飾語がつくかどうかという曖昧性が多かった。例えば “two nuclear proteins”, “pure B cellline” などの修飾語が正解スパンに含まれる事例があった。その一方で、novel “TH protein”, purified “NF-kappa B” など、修飾語を含

Model	P.	R.	F.	nest inside prediction	nest inside R.	nest outisde R.
Dictionary Match	24.78	25.08	24.93	1464	23.66	24.35
Chunker + End Match	59.07	29.38	39.24	10	26.00	29.57
Oracle Span + End Match	87.92	46.54	60.87	241	43.49	46.39
Gold	100	100	100	646	100	100

表 2　擬似データ作成手順による精度

Model	P.	R.	F.	nest inside prediction	nest inside R.	nest outisde R.
Chunker + End Match	59.07	29.38	39.24	10	26.00	29.57
BioBERT (Distantly Supervised)	24.27	62.38	34.95	20449	60.46	65.74
+ Chunker	54.65	43.29	48.31	383	40.71	46.63
BioBERT (Supervised)	76.64	74.97	75.79	936	72.99	75.33
Gold	100	100	100	646	100	100

表 3　Distant Supervision 適用による精度

まないスパンが正解スパンとされている事例も存在した。このような事例に対して辞書に修飾語がつかない語句が含まれているならば、修飾語の含まれる場合・含まれない場合どちらが正解かを決めることは困難であると考えられる。

5.2 Distant Supervision

表 3 より “Chunker + End Match” と “BioBERT (Distantly Supervised)” の比較から、このノイズのあるデータに基づいて学習・予測させることは難しいということがわかる。具体的には並列句のような長いスパンを予測してしまうことが多かった。“+ Chunker” と予測範囲を Chunker によるスパン内部に限定しない “BioBERT (Distantly Supervised)” の比較から、精度が改善していることがわかる。これは、先ほど述べたような長いスパンを抑制できているためだと考えられる。一方で、教師あり手法を利用した “BioBERT (Supervised)” と比較すると 20 % ほどの差があることから、データノイズの影響や Chunker の制限により予測できなくなったスパンなどの影響で依然として教師あり手法にはかなり劣っていることがわかる。

入れ子になっている予測事例の数を確認すると、“BioBERT (Distantly Supervised)” が飛び抜けて多い。これは上記のような長いスパンの影響が考えられる。精度が低いものの “+ Chunker” では一定数の入れ子の事例を予測している。入れ子を直接訓練事例として与えなくても、構成する単語などから、入れ子の固有表現を予測しているのだと考えられる。興味深いのは教師あり学習の結果、アノテーションデータと比べて約 1.5 倍入れ子のスパンを予測している点である。教師あり学習の際にも、構成される単語などから過剰に固有表現を予測してしまう傾向がある可能性が考えられる。

実際の入れ子事例に対する精度を見てみると、データセット全体のスコアと大きな違いが出ていないことが分かる。これも擬似データ作成の場合と同様に、入れ子の外側と内側では入れ子の外側の特定が容易であることが見て取れる。

6 結論

本稿では擬似データ作成に基づく入れ子の可能性を考慮した固有表現抽出に取り組んだ。擬似データの作成がそもそも難しく、不十分な擬似データ手法によるノイズの多いデータに基づいた学習は難しいということがわかる。

Distant Supervision NER の先行研究の BOND [9] では、ノイズのあるデータに過学習させないことで、不十分な精度の擬似データに対処している。このように今後はデータ中のノイズを考慮した手法 [15] を試みる必要があると考えている。

また、5.1.1 に述べたが、そもそもどれだけ複雑な概念を取ってくるかには曖昧性がある。例えば辞書に “water” だけがある場合、“pure” を含めた “pure water” を抽出するかどうかは応用先の意図に依存し、辞書だけで判定するのは困難である。そのため Distant Supervision NER において、固有表現が取れている/取れていない を判定するには、別の評価尺度を併用することが必要なのではないだろうか。具体的には正解スパンの headword が取れているか [3] などの評価方法が考えられる。

参考文献

[1] Mingbin Xu, Hui Jiang, and Sedtawut Watcharawittayakul. A Local Detection Approach for Named Entity Recognition and Mention Detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1237–1247, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[2] Mohammad Golam Sohrab and Makoto Miwa. Deep Exhaustive Model for Nested Named Entity Recognition. In *EMNLP*, 2018.

[3] Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. A general framework for information extraction using dynamic span graphs. 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)*, pp. 3036–3046, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Hiroki Ouchi, Jun Suzuki, Sosuke Kobayashi, Sho Yokoi, Tatsuki Kuribayashi, Ryuto Konno, and Kentaro Inui. Instance-Based Learning of Span Representations: A Case Study through Named Entity Recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, July 2020.

[5] L. Sun, F. Ji, K. Zhang, and C. Wang. Multilayer ToI Detection Approach for Nested NER. *IEEE Access*, Vol. 7, pp. 186600–186608, 2019. Conference Name: IEEE Access.

[6] Jingbo Shang, Liyuan Liu, Xiaotao Gu, Xiang Ren, Teng Ren, and Jiawei Han. Learning Named Entity Tagger using Domain-Specific Dictionary. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2054–2064, Brussels, Belgium, 2018. Association for Computational Linguistics.

[7] Zhanming Jie, Pengjun Xie, Wei Lu, Ruixue Ding, and Linlin Li. Better Modeling of Incomplete Annotations for Named Entity Recognition. 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)*, pp. 729–734, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[8] Nathan Greenberg, Trapit Bansal, Patrick Verga, and Andrew McCallum. Marginal Likelihood Training of BiLSTM-CRF for Biomedical Named Entity Recognition from Disjoint Label Sets. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2824–2829, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[9] Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. BOND: BERT-Assisted Open-Domain Named Entity Recognition with Distant Supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 1054–1064, New York, NY, USA, August 2020. Association for Computing Machinery.

[10] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. GENIA corpus—a semantically annotated corpus for bio-textmining. *Bioinformatics*, Vol. 19, No. suppl_1, pp. i180–i182, July 2003.

[11] Jason Fries, Sen Wu, Alex Ratner, and Christopher Ré. SwellShark: A Generative Model for Biomedical Named Entity Recognition without Labeled Data. *arXiv:1704.06360 [cs]*, April 2017. arXiv: 1704.06360.

[12] Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *BioNLP@ACL*, 2019.

[13] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6442–6454, Online, November 2020. Association for Computational Linguistics.

[14] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, p. btz682, September 2019. arXiv: 1901.08746.

[15] Hwanjun Song, Minseok Kim, Dongmin Park, and Jae-Gil Lee. Learning from Noisy Labels with Deep Neural Networks: A Survey. *arXiv:2007.08199 [cs, stat]*, October 2020. arXiv: 2007.08199.

[16] Olivier Bodenreider. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, Vol. 32, No. Database issue, pp. D267–270, January 2004.

A　POS に基づく Chunker

本稿で利用した POS に基づく Chunker は POS タグを取得した後、正規表現を利用して Chunk を取得する。なお POS タグの検出には sciSpacy [12] を利用した。この正規表現は Chunk の左部分の検出と右部分の検出、修飾語及び語句末端の検出部分から構成されている。例えば、 “The United States is . . . ” という文に対して “DT NNP NNP VBZ . . . ” という POS タグ系列が得られたとする。この時、Chunk の左側として “DT” を、右側として “VBZ” を、修飾語として “NNP” を語句末端として “NNP” を検出する。

Chunk の左側は “DT”, “IN”, “CC”, “,”, “RB”, “RBR”, “RBS”, “RP”, “VB”, “VBD”, “VBZ”, “VBP”, “.”, “-LRB-”, “WDT”, “TO”, “:”, “NNS”, “-RRB-”, “PRP$” のいずれかの POS タグである。ただし、文頭の可能性もあるとする。Chunk の右側は “,”, “.”, “IN”, “VB”, “VBD”, “VBZ”, “VBN”, “VBP”, “CC”, “MD”, “DT”, “RB”, “RBR”, “RBS”, “JJ”, “JJR”, “JJS”, “RP”, “-LRB-”, “-RRB-”, “WDT”, “TO”, “:” のいずれかの POS タグである。ただし文末の可能性もあり、語句末端の語句が複数形の場合 “NN” もふくむとする。修飾語は “JJ”, “NN”, “NNP”, “CD”, “VBG”, “VBN”, “”” の POS タグのうちのいずれかの繰り返しである。

語句末端の POS タグを考慮するにあたって、複数形と単数形で対応を変えた。単数形の場合、語句末端は “NN” あるいは “VBG” であるとした。複数形の場合 “NNS” であるとした。また、“interleukin 2” のような事例に対応するために末端に数字: “CD” が来ても良いとした。

以上の説明を python の正規表現にまとめるの次のようにかける。まず単数形の場合、

```
(DT|IN|CC|,|RB|RBR|RBS|RP|VB|VBD|VBZ|VBP|
.|-LRB-|WDT|TO|:|NNS|-RRB-|PRP$|<s>)
((JJ|NN|NNP|CD|VBG|VBN|'')*(NN|VBG)(CD)?)
(,|.|IN|VB|VBD|VBZ|VBN|VBP|CC|MD|DT|RB|RBR|
RBS|JJ|JJR|JJS|RP|-LRB-|-RRB-|WDT|TO|:|</s>)
```

とかける。(スペースの都合上改行を入れている) 次に複数形の場合は

```
(DT|IN|CC|,|RB|RBR|RBS|RP|VB|VBD|VBZ|VBP|.|
-LRB-|WDT|TO|:|NNS|-RRB-|PRP$|<s>)
((JJ|NN|NNP|CD|VBG|VBN|'')*(NNS)(CD)?)
(,|.|IN|VB|VBD|VBZ|VBN|VBP|CC|MD|DT|RB|RBR|
RBS|JJ|JJR|JJS|RP|-LRB-|-RRB-|WDT|TO|:|</s>|NN)
```

となる。ただし、“<s>” を文頭、“</s>” を文末の意味として利用している。

B　UMLS からの辞書作成

今回テストデータとして利用した GENIA コーパスに直接対応する辞書は存在しない。そこで本研究では UMLS [16] の情報を変換し、今回の擬似データ生成に利用する辞書を作成した。利用した UMLS のバージョンは 2020AA であり、その知識グラフの全体を利用した。ただし、問題の多かった “HGNC”, “OMIM”, “NCI”, “SNOMEDCT_US”, “PDQ”, “CHV”, “LNC” の知識グラフについては除外した。

次に GENIA コーパスにおける五つのそれぞれのカテゴリ (protein, DNA, RNA, cell type, cell line) に対して、そのカテゴリの語句をどのように取得したかを述べる。まず “protein” に対しては、“Proteins” C0033684 及びその子孫を利用した。さらにそこから後述の “DNA”, “RNA” の語句として得られる UMLS Concept を取り除いた。ただし、ここで UMLS Sematic Type T116: “Amino Acid, Peptide, or Protein”, T087: “Amino Acid Sequence” に含まれる UMLS Concept だけを残した。“DNA” に関しては C0012854: “DNA”, C0162326: “DNA Sequence”, C0008633: “Chromosomes”, C0019652: “Histones” 及びその子孫を利用し、“DNA”, “DNA Sequence” 及び、“Histones” でない “Chromosomes” を “DNA” のカテゴリとして利用した。ただし、UMLS Sematic Type T028: “Gene or Genome”, T114: “Nucleic Acid, Nucleoside, or Nucleotide”, T086: “Nucleotide Sequence”, T026: “Cell Component” に含まれる UMLS Concept だけを残した。“RNA” に関しては C0035668: “RNA” 及びその子孫を利用した。ただし、UMLS Sematic Type T114: “Nucleic Acid, Nucleoside, or Nucleotide”, T086: “Nucleotide Sequence” に含まれる UMLS Concept のみに限定した。“cell type” に関しては UMLS Semantic Type が “T025” であるような UMLS Concept を利用した。“cell line” に関しては C0007600: “Cultured Cell Line” 及びその子孫を利用した。

その後、複数のカテゴリに含まれるような曖昧性のある語句を除去した。例えば “IL2” という文字列は、“Interleukin-2” というタンパク質名としても、“IL2 gene” という遺伝子名としても登録されている。この段階でのそれぞれのカテゴリの語句数は表 4 のとおりである。また、辞書マッチの recall を上げるために inflect [5] という python ライブラリを利用して、それぞれの語句に対して複数形を辞書に追加した。

カテゴリ	語句数
protein	169,760
DNA	100,544
RNA	6,335
cell type	13,542
cell line	525

表 4　UMLS を元にした辞書中の各カテゴリごとの語彙数

5)　https://github.com/jaraco/inflect