Fast entity recognition in biomedical text

Amy Siu
Max Planck Institute
for Informatics
Saarbrücken, Germany
siu@mpi-inf.mpg.de

Dat Ba Nguyen
Max Planck Institute
for Informatics
Saarbrücken, Germany
datnb@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute
for Informatics
Saarbrücken, Germany
weikum@mpi-inf.mpg.de

ABSTRACT

In biomedical text mining, entity recognition is often an early task in the pipeline of analyzing free text. MetaMap, the *de facto* standard software tool for this task, employs much Natural Language Processing (NLP) machinery to recognize entities in UMLS (Unified Medical Language System), the largest metathesaurus. Knowing that the NLP machinery is time-consuming, and that UMLS is rich in lexical variations, this work investigates whether a fast, string similarity-based method can achieve results comparable to those of MetaMap. We implemented an NLP-light method that performs fast MinHash lookups via character trigram features. Starting with UMLS as the dictionary of entities, we select a subset whose entity names are short and thus amenable to a string similarity-based approach. We applied the method to both scientific literature and layman-oriented texts from Internet health portals. Our proposed method achieved up to 83% precision and 78% coverage under a strict rating scheme that penalizes failure in Word Sense Disambiguation (WSD), at a throughput of 1,720 PubMed abstracts or 175 web pages per minute. When compared to MetaMap, our proposed method achieved comparable precision and 13% less coverage using less than 1% of the time.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Medical Information Systems; I.2.7 [**Natural Language Processing**]: Text Analysis

Keywords

Biomedical text mining, entity recognition

1. INTRODUCTION

In recent years, the amount of biomedical information has grown tremendously. Much of this information is published as free texts in scientific literature as well as layman-oriented health portals on the Internet. The Biomedical Natural Language Processing (BioNLP) community has responded by developing and applying various text mining techniques in order to extract this information buried in free texts.

In a BioNLP text mining pipeline, recognizing entities in these free texts often serves as an early task, upon which other downstream tasks depend. Knowledge harvesting, for instance, requires text mentions to be mapped to entities in a dictionary before relations between them can be analyzed. Downstream tasks not only depend their success on the quality of the entities recognized; they may not even begin before the entity recognition task is completed. Therefore, it is crucial that a biomedical entity recognition method provides both high quality and high throughput.

Entity recognition in the biomedical domain presents unique challenges uncommon to the general domain. Biomedical texts, especially scientific literature geared towards professionals, are steep in specialized jargons. Biomedical concepts are often expressed in long phrases with a large number of variations. Given a text mention, there is often a high degree of ambiguity for the text mention-entity mapping. In fact, there is often inherent vagueness in the text regarding the precise type of the entity; whether *high blood pressure* is a disease, a symptom, or a risk factor to a disease is not easily distinguishable even given the context. Word Sense Disambiguation (WSD) is therefore a closely related research topic, whose contribution can greatly refine the output of an entity recognition method.

To counter these challenges, entity recognition has been a major area of research within the BioNLP community. Over the past decade, MetaMap [1] has become the *de facto* standard software tool for general-purpose biomedical entity recognition. Employing much Natural Language Processing (NLP) machinery, MetaMap trades off higher quality with lower throughput. As text collections grow in size, this lower throughput gradually becomes the bottleneck of a BioNLP text mining pipeline. In our experience, without parallelization, processing 600k PubMed[1] abstracts – a small portion of over 16m English abstracts in the entire collection – takes 26 days (3.8s per abstract using a single instance of MetaMap). On the other hand, we observe that MetaMap takes UMLS (Unified Medical Language System)[2] as its dictionary of entities. UMLS is the largest metathesaurus, rich in synonyms and lexical variations. This observation spurs us to investigate an alternative entity recognition method that is fast and exploits this lexical richness.

KDD–DMH'13, August 11, 2013, Chicago, Illinois, USA.
Copyright ©2013 ACM 978-1-4503-2174-7/13/08 ...$15.00.

[1]www.ncbi.nlm.nih.gov/pubmed
[2]www.nlm.nih.gov/research/umls

This work presents a string similarity-based method for biomedical entity recognition aimed at minimizing the use of NLP machinery, and thus processing time. The key ingredient of the method is MinHash [2], a variant of the Locality Sensitive Hashing (LSH) [4] algorithm that transforms a dictionary lookup into a hash lookup with high probability of success. Together with judicious selection of a subset of the UMLS dictionary and simple heuristics in selecting which text mentions to perform lookups for, our method achieves up to 83% precision, under a strict rating scheme that penalizes failure in WSD, at 35ms per abstract. By comparing our results to those of MetaMap, we hope to demystify the contributions of NLP components in an entity recognition machine.

2. RELATED WORK

Entity recognition in the biomedical domain often focuses on a specific sub-domain. Proteins and genes are the most popular sub-domain to date. BioCreative has been driving the community with the gene mention normalization task [8], that is, to map a text mention of a gene or a gene product to biological databases. Out of a large body of works, there are a number of software tools publicly available, where Gimli [3] and ABNER [12] are two notable ones. Besides recognizing proteins and genes, there are also works focusing on chemical entities [11], disease names [7], and organisms [9], to name only a few. Other efforts contribute to providing linguistic resources for the community, of which BioLexicon [13] is a recent addition with over 2.2m lexical entries. For general-purpose biomedical entity recognition, however, MetaMap and UMLS remain the most widely used resources.

String similarity-based methods are frequently employed to perform various text-oriented tasks in the biomedical domain. Yamaguchi *et al.* [16] compare the performances of four different string similarity metrics for the task of clustering chemical and non-chemical abbreviations. Wellner *et al.* [15] combine an adaptive string similarity model with conditional random fields to pick up protein names in free text. String similarity metrics can be cast as a machine learning problem, as Tsuruoka *et al.* [17] propose – given protein names, learn the metric via logistic regression. The resulting metric is later used to look strings up from a dictionary of protein names.

LSH is a proven technique used in numerous applications, especially when one requires speed in working with large data sets. Ravichandran *et al.* [10] present an NLP example, where nouns from a web corpus are clustered based on cosine similarity. Chum *et al.* [5] present another, prominent example in the area of computer graphics. The authors extend the hashing algorithm with weighted set similarity measures. The resulting algorithm is capable of detecting near duplicate images and videos, and is highly scalable.

3. METHOD

3.1 Dictionary Lookup via MinHash

MinHash algorithm. LSH [4] is a probabilistic method that reduces the dimensions of a high-dimensional data set. Intuitively speaking, similar items in the data set are hashed with high probability to the same bucket. MinHash (min-wise independent permutations) [2], a variant of LSH, has the additional property that the probability of two sets S_1 and S_2 being hashed to the same bucket is exactly their Jaccard similarity:

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$$

In our implementation of MinHash, we encode a string as the set S of character trigrams of the string. Let π_1, π_2, $\ldots \pi_k$ be hash functions from k independent min-wise permutations, such that each π maps trigrams to integers. Then $\pi(S)$ is the set of integers thus mapped, and let $min(\pi(S))$ be the smallest integer in this set. The hash value, or signature, of the string is a concatenation of these smallest integers from each permutation:

$$min(\pi_1(S)) \oplus min(\pi_2(S)) \oplus \cdots \oplus min(\pi_k(S))$$

The concatenation operator is implemented as simple arithmetic summation. [2] shows that, for each permutation π:

$$P[min(\pi(S_1)) = min(\pi(S_2))] = J(S_1, S_2)$$

To tune the MinHash parameters, we performed preliminary experiments. Besides reviewing precision and recall, we also wanted to minimize lookup time and bucket collisions, or false positives. We applied MinHash to UMLS dictionary entities, and looked up 500 random entity names. The optimal parameters that yielded the best results were as follows: choose k=30 from 14k permutations, project the data set into 12m dimensions, and use 2 MinHash tables to increase recall.

Dictionary construction. UMLS is made of up entities, called *concepts* in UMLS documentation, where one entity is represented by one or frequently multiple entity names bearing the lexical variations from different thesauri sources. Being a potpourri of different thesauri with heterogeneous norms, UMLS contains entity names unsuitable for a string similarity-based method. Specifically, entity names featuring long strings are unlikely to appear verbatim in free text. Including such entities in MinHash also clogs the hash tables with unproductive signatures and increases collisions. Therefore, we take only those entity names in UMLS that are 5 words or shorter, and 100 characters or shorter. Using the freely available (category 0) portion of the 2012AB data set, the subset of UMLS thus obtained features 2.7m unique entity-entity name pairs.

Many entities in UMLS already provide ample lexical variations such as singular and plural forms (for example in entity C0020974 *immunoglobulin injection*, *immunoglobulin injections*, and *immunoglobulins injection*), verb conjugations (C0021107 *implant*, *implanted*, and *implanting*), and different word orders (C0021943 *chromosome inversion* and *inversion chromosome*). Some entities, however, do not contain such information. Since the MinHash implementation relies heavily on the completeness of lexical variations, we augment our subset of UMLS by generating missing variations.

The first augmentation concerns plural forms of nouns. We use WordNet [6] to detect entity names that end with an English noun, and then use MorphAdorner[3] to generate

[3]morphadorner.northwestern.edu

Corpus	Genre	Number of articles selected	Average number of words in one article
PubMed abstracts from 2011	Scientific literature	5,000	181.47
PubMed Central full-length articles from 2011		500	3,038.01
MayoClinic	Health portal on the Internet	500	1,793.47
UpToDate		500	2,570.94
Wikipedia health portal		500	510.05

Table 1: Composition of test articles

the noun's plural form. The entity C0751248 *M'Naghten rule*, for instance, is augmented with the additional entity name *M'Naghten rules.* This procedure generates 439k entity names.

The second augmentation concerns verb conjugations. Starting from entities featuring a single word, we first use WordNet to check that it is an English verb. Then we verify that the corresponding entity belongs to UMLS-defined semantic groups that feature verbs. We choose *Activities & Behaviors*, *Concepts & Ideas*, *Phenomena*, *Physiology*, and *Procedures* as the qualifying semantic groups, such that, for instance, a gene named *CASH* is disqualified. After passing these tests, we apply MorphAdorner to conjugate the verbs. Care is taken not to incorporate a generated variation when such an entity name already exists in UMLS, because a pre-existing entity may well represent a semantically different entity. For instance, although from C0175735 *shear* the medical device we could generate *shearing*, that entity name already exists as entity C0205013 the therapeutic procedure. In this case, *shearing* is not used to augment the medical device entity. This procedure generates 4,614 entity names.

False positive pruning. Due to the probabilistic nature of MinHash, collisions of different entity names being hashed to the same bucket are inevitable, leading to false positives. We detect false positives by comparing the Jaccard similarity of character trigram sets between a text mention and a dictionary entity name. When the Jaccard similarity scores below the threshold of 0.8, the entity name is discarded as a false positive.

3.2 Selecting Text Mentions for Lookup

Fast and robust dictionary lookups contribute to only half of the success. The other half comes from the module that selects text mentions for lookups. We present two strategies that adhere to the theme of minimizing NLP machinery.

Consecutive words. This strategy is nearly NLP-free: simply take consecutive words as text mentions, and use heuristics to trim the text mentions or discard undesirable ones. We start with the word length of 1, or every single word. Discard the word when it is a stop word, or when the word has only 3 or fewer characters. When looking at word lengths of 2 or more, remove any leading stop words. Discard the text mention when there is a punctuation dividing the words, as this indicates the text mention would not represent a coherent entity. Since the dictionary only contains entity names up to 5 words long, we also limit the length of text mentions to 5 words. Despite the large number of text mentions generated, this strategy is viable because it is fast, and the accuracy of mapped entities is taken care of by the dictionary lookup module.

Noun phrases. This strategy is NLP-light: take only noun phrases as text mentions. We use the Stanford CoreNLP tool [14] to assign part-of-speech tags, and then use OpenNLP[4] to perform noun phrase chunking. We further identify complex noun phrases in the form of:

noun group – preposition – noun group

where a noun group is in the form of:

[[optional adverb] – optional adjective] – noun

Complex noun phrases allow us to capture text mentions like *shortness of breath* and *left lower lobe of lung*, as well as *lobe of lung* when the optional adjectives *left lower* are omitted. Notice that this strategy generates strictly a subset of those text mentions from the consecutive words strategy. This distinction addresses the question regarding the balance between precision and recall, as we want to investigate whether using more selective text mentions improves precision.

4. EVALUATION

We programmed the aforementioned MinHash method in Java, and ran the experiments in standard Linux machines with 8 Intel Xeon CPUs at 2.4GHz and 48Gb of main memory. To maximize lookup speed, the program loads the precomputed MinHash tables in main memory, at a one-off cost of 20s when the program starts up. The collection of test articles are randomly selected from both biomedical scientific literature and layman-oriented health portals on the Internet (see Table 1).

To ensure that MetaMap achieved the best possible performance, we loaded the MetaMap program and all of its associated data files into shared memory (shm). We cut MetaMap's runtime by half by issuing one request per article rather than one per sentence. In addition, MetaMap used the 2012AB base data set, which corresponded to the same portion of UMLS we constructed our dictionary from. Finally, MetaMap provides scored entities for each text mention. We only used the top-scoring entity in our evaluation; where multiple entities shared the same top score, all of those entities were taken into consideration.

4.1 Precision

UMLS entity names sharing the same lexical form often represent semantically different entities. The text mention

[4]opennlp.apache.org

	UMLS		UMLS + P		UMLS + V		UMLS + PV		MetaMap	
	sci	web	sci	web	sci	web	sci	web	sci	web
Consecutive words	0.94	0.98	0.97	0.96	0.94	0.99	0.96	0.99	0.94	0.96
Noun phrases	0.91	0.96	0.94	0.99	0.94	0.97	0.92	0.96		

(a) Lenient rating scheme

	UMLS		UMLS + P		UMLS + V		UMLS + PV		MetaMap	
	sci	web	sci	web	sci	web	sci	web	sci	web
Consecutive words	0.71	0.81	0.74	0.71	0.67	0.80	0.75	0.83	0.79	0.81
Noun phrases	0.73	0.78	0.78	0.83	0.64	0.79	0.74	0.81		

(b) Strict rating scheme

Table 2: Precision of the MinHash method and MetaMap

medicine can be mapped to entity C0013227 the pharmacologic substance, and to entity C0025118 the occupational discipline. While our string similarity-based method lacks the power to discern between these semantic differences, MetaMap has a WSD module that removes incorrect entities. In order to assess how much WSD contributes to the final mappings, we evaluated precision using two rating schemes. In the lenient rating scheme, as long as a text mention is mapped to at least one correct entity, we rated this text mention as correct. In the strict rating scheme, the presence of any incorrect entity would rate the text mention as incorrect. In other words, the strict rating scheme penalizes failure in WSD.

Table 2 shows the precision results of our program under various combinations of experimental setups. In the table's headers, *UMLS* denotes the UMLS-subset dictionary; *+P* and *+V* denote augmenting it with the plural nouns and verb conjugations respectively. *sci* and *web* denote the genres of the test articles, namely scientific literature and health portal on the Internet respectively. Each cell in the table presents the precision evaluated from 100 randomly sampled text mentions. Overall, the MinHash method trails behind MetaMap in precision, though in a few settings their precision results are comparable (between 4% worse to 2% better). A few trends are observed:

Lenient vs. strict rating scheme. Under the lenient rating scheme, the MinHash method scores consistently over 90% in precision. This result is expected, as MinHash finds, for a text mention, all entity names in the dictionary spelled similarly. Almost all the time, at least one of these entity names would be the entity expressed in the text mention. In fact, MinHash fails when the text mention is spelled similarly to unrelated entity names *and* when the correct entity does not offer a lexical variation similar to the text mention. For instance, the word *architecture* in the phrase *interfere[nce] with sleep architecture* is mapped to entity C0003737 the occupation, the only entity name in the dictionary with that spelling.

Naturally, both the MinHash method and MetaMap lose precision under the strict rating scheme. While the MinHash method loses between 15% to 30%, MetaMap only loses 15% across all settings. MinHash'es heavier loss can be attributed to its lack of WSD. A more precise diagnosis, however, requires more in-depth investigation into the distribution of text mentions requiring WSD, together with the distribution of UMLS entity names causing ambiguity.

Consecutive words vs. noun phrases. The precision of text mentions selected via the noun phrases strategy is generally lower than that via the consecutive words strategy. We find this result surprising, as one would expect noun phrases to be text mentions whose lexical variations are likely to be found in the dictionary. Upon closer examination, it turns out that the root of the problem lies in the noun chunks identified by the chunking tool. Many such chunks are acronyms, numbers spelled out as English words, and single words high in ambiguity such as *form* and *system* – precisely the types of text mentions a simple string similarity-based approach does not handle well. Compared to the consecutive words strategy, the noun phrases strategy uses a higher proportion of such problematic text mentions, dragging the precision down.

Corpus genre. Precision observed in layman-oriented articles generally outperforms that in scientific ones. One contributing factor is the lack of acronym detection across multiple sentences, as scientific literature features acronyms more frequently. More importantly, we observe that sentences in scientific articles – especially abstracts – are often long with convoluted sentence structures. As soon as a text mention does not adequately express the full nature of the corresponding entity, a simple string similarity-based lookup would fail. A common example is the listing of multiple items, as in *cell proliferation, differentiation and migration*, where *differentiation* and *migration* are incomplete text mentions, and only *cell proliferation* fully expresses the entity.

One key observation here is that noun phrase chunking does not rectify this situation. We conjecture that a better solution lies in leveraging the sentences' dependency parse trees, such that text mentions may be properly constructed before looking up the dictionary.

4.2 Coverage

To the best of our knowledge, although there are corpora annotated for highly focused sub-domains such as proteins and their interactions, there is none annotated with all types of biomedical entities. To provide an indication of recall, then, we rated every text mention in 30 random PubMed abstracts from the test articles. We took the union of all correct text mentions mapped by either the MinHash method or MetaMap, and let this larger set of text mentions be an estimation of complete coverage. Using the lenient and strict rating schemes, the 30 abstracts covered a total of 2,481 and 2,401 text mentions respectively.

	MinHash con	MinHash NP	MetaMap
Lenient	0.8420	0.2048	0.9105
Strict	0.7839	0.1930	0.9138

Table 3: Coverage of the MinHash method and MetaMap

	Number of PubMed abstracts per minute	Number of other articles per minute	Number of words per minute
MinHash con	1,720.19	175.15	339,508
MinHash NP	863.22	85.42	166,533
MetaMap	15.26	1.41	2,786

Table 4: Throughput of the MinHash method and MetaMap

Table 3 shows the coverage of the MinHash method and MetaMap for these 30 abstracts. Since augmenting the UMLS subset dictionary with verb and noun variations yielded the best precision, here we used only this UMLS + PV dictionary for the MinHash method. *con* and *NP* denote the consecutive words and noun phrases strategies respectively. Again, some trends are observed:

Low coverage of noun phrases strategy. The most glaring observation in Table 3 is the low coverage of the noun phrases strategy. Although disappointing, the numbers are not surprising. In the 30 abstracts, only 13% of all words are chunked as noun phrases, and thus further taken as text mentions for lookups by the MinHash method. Compare this with the consecutive words strategy, where all words regardless of parts-of-speech are considered, and verbs in particular contribute to many text mentions. Under the consecutive words strategy, 65% of all words are eventually included in text mentions with (both correctly and incorrectly) mapped entity names.

MinHash and MetaMap complement each other. Regardless of the lenient or strict rating scheme, MetaMap achieves a stable coverage at 91%. As with precision, the MinHash method with consecutive words strategy also trails behind MetaMap in coverage, but is no rival here due to a gap of up to 13%. Notice that neither the MinHash method nor MetaMap finds every text mention in our estimated complete coverage. Let us visit some notable patterns that allude to the strengths and weaknesses of both programs, which may shed some light on why both programs pick up entities the other does not.

MetaMap is capable of analyzing text mentions syntactically, such that "less important" words within a text mention may be skipped. Consider the text mentions *aerobic anoxygenic phototrophic bacteria* and *drug-endogenous substance interaction*. They are mapped to *aerobic bacteria* and *drug interactions* respectively, which are indeed correct entities despite losing some specificity. The MinHash method only considers a sequence of words in its entirety, and would never have found such coarser-grained entity names.

MetaMap's syntactic analysis is not accurate all the time, however. Where it makes a mistake is where the MinHash method may prove complementary. Perhaps due to chunking errors, text mentions like *shortness of breath* and *pain breakthrough* do not always remain intact; MetaMap may split the text mention into shorter text mentions of single words. Consequently, single words are mapped to their own, separate entities, causing the original, longer text mention as a whole to miss out on getting mapped to more applicable entities. (Using the "term processing" option, one can force MetaMap to take a text mention as-is without splitting it, but this requires the user to provide the text mentions, which in turn requires the user to precompute some linguistic analysis.) The MinHash method's consecutive words strategy, being blind to syntactic analysis, would always attempt to lookup all text mentions consisting of a sequence of words.

Finally, MetaMap has built-in support for acronym detection, a feature that the MinHash method does not provide. An acronym such as *BAC* represents four different entities, and the correct entity can only be inferred from the spelled out entity name usually appearing prior to the acronym in the same article. Similar cases contribute to text mentions the MinHash method misses but are picked up by MetaMap.

4.3 Throughput

We recorded the time required to apply the MinHash method and MetaMap to all the test articles. Both text mention selection strategies, consecutive words and noun phrases, were performed using the UMLS + PV dictionary. We report the average processing time from 5 repeated runs in Table 4.

Employing almost no NLP machinery, the consecutive words strategy is the fastest. The noun phrases strategy, which uses light NLP machinery, takes twice as long as the consecutive words strategy. Given that the consecutive words strategy performs better in precision *and* coverage, let us use this setting to revisit the scenario presented in the Introduction. Instead of 26 days, processing 600k PubMed abstracts now with the MinHash method will take less than 6 hours.

5. CONCLUSIONS AND FUTURE WORK

MetaMap is the *de facto* standard biomedical entity recognition software tool that uses much NLP machinery. At the cost of higher quality, however, is its lower throughput. We presented an alternative, fast biomedical entity recognition method, with the aim of achieving near-MetaMap quality at a fraction of the time. This alternative is a string similarity-based method built upon the MinHash algorithm, operating over a carefully constructed dictionary of entity names based on UMLS. Using the consecutive words strategy to select text mentions, the MinHash method achieves a precision of up to 83% and a coverage of up to 78% under the strict rating scheme. Our method's precision is comparable to that of MetaMap (between 4% worse to 2% better), though our coverage trails behind MetaMap by 13%. It appears that while heavy NLP machinery does boost precision and coverage, only a minority of text mentions benefit from it; the majority of text mentions can be accurately mapped to entities using the MinHash method alone. And since the Java implementation of our method runs at two magnitudes faster than MetaMap, we intend to utilize the time saved to address that minority with appropriate NLP techniques.

Going forward, we would like to incorporate Word Sense Disambiguation (WSD), as we believe it will have the most impact on bridging the gap in quality. Secondly, to improve the selection of text mentions, we would like to investigate the use of dependency parsing, which will hopefully prove more beneficial than noun phrase chunking. Finally, we would also like to leverage existing work on biomedical acronym detection, and incorporate such a module into our implementation.

6. REFERENCES

[1] A. Aronson and F. Lang. An overview of MetaMap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229–236, 2010.

[2] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, STOC '98, pages 327–336, 1998.

[3] D. Campos, S. Matos, and J. Oliveira. Gimli: open source and high-performance biomedical name recognition. *BMC Bioinformatics*, 14(1):54, 2013.

[4] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, 2002.

[5] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *Proceedings of the British Machine Vision Conference*, BMVC '02, volume 3, pages 4–13, 2008.

[6] C. Fellbaum. *WordNet: An Electronic Lexical Database.* Springer, 2010.

[7] R. Islamaj Dogan and Z. Lu. An inference method for disease name normalization. In *Proceedings of 2012 AAAI Fall Symposium Series*, pages 8–13, 2012.

[8] A. Morgan, Z. Lu, X. Wang, A. Cohen, J. Fluck, P. Ruch, A. Divoli, K. Fundel, R. Leaman, J. Hakenberg, C. Sun, H.-H. Liu, R. Torres, M. Krauthammer, W. Lau, H. Liu, C.-N. Hsu, M. Schuemie, K. B. Cohen, and L. Hirschman. Overview of BioCreative II gene normalization. *Genome Biology*, 9(Suppl 2):S3, 2008.

[9] N. Naderi, T. Kappler, C. J. Baker, and R. Witte. OrganismTagger: detection, normalization, and grounding of organism entities in biomedical documents. *Bioinformatics*, 27(19):2721–2729, 2011.

[10] D. Ravichandran, P. Pantel, and E. Hovy. Randomized algorithms and NLP: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, ACL '05, pages 622–629, 2005.

[11] T. Rocktäschel, M. Weidlich, and U. Leser. ChemSpot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640, 2012.

[12] B. Settles. ABNER: an open source tool for automatically tagging genes, proteins and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

[13] P. Thompson, J. McNaught, S. Montemagni, N. Calzolari, R. del Gratta, V. Lee, S. Marchi, M. Monachini, P. Pezik, V. Quochi, C. Rupp, Y. Sasaki, G. Venturi, D. Rebholz-Schuhmann, and S. Ananiadou. The BioLexicon: a large-scale terminological resource for biomedical text mining. *BMC Bioinformatics*, 12(1):397, 2011.

[14] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL '03, volume 1, pages 173–180, 2003.

[15] B. Wellner, J. Castaño, and J. Pustejovsky. Adaptive string similarity metrics for biomedical reference resolution. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 9–16, 2005.

[16] A. Yamaguchi, Y. Yamamoto, J.-D. Kim, T. Takagi, and A. Yonezawa. Discriminative application of string similarity methods to chemical and non-chemical names for biomedical abbreviation clustering. *BMC Genomics*, 13(Suppl 3):S8, 2012.

[17] Y. Tsuruoka, J. McNaught, J. Tsujii, and S. Ananiadou. Learning string similarity measures for gene/protein name dictionary look-up using logistic regression. *Bioinformatics*, 23(20):2768–2774, 2007.