Journal of Biomedical Informatics 57 (2015) 320–332

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

PKDE4J: Entity and relation extraction for public knowledge discovery

Min Song *, Won Chul Kim, Dahee Lee, Go Eun Heo, Keun Young Kang

Department of Library and Information Science, Yonsei University, 50 Yonsei-ro, Seodaemun-gu, Seoul 120-749, Republic of Korea

ARTICLE INFO

Article history:
Received 26 February 2015
Revised 20 July 2015
Accepted 6 August 2015
Available online 12 August 2015

Keywords:
Text mining
Information extraction
Named entity recognition
Relation extraction

ABSTRACT

Due to an enormous number of scientific publications that cannot be handled manually, there is a rising interest in text-mining techniques for automated information extraction, especially in the biomedical field. Such techniques provide effective means of information search, knowledge discovery, and hypothesis generation. Most previous studies have primarily focused on the design and performance improvement of either named entity recognition or relation extraction. In this paper, we present PKDE4J, a comprehensive text-mining system that integrates dictionary-based entity extraction and rule-based relation extraction in a highly flexible and extensible framework. Starting with the Stanford CoreNLP, we developed the system to cope with multiple types of entities and relations. The system also has fairly good performance in terms of accuracy as well as the ability to configure text-processing components. We demonstrate its competitive performance by evaluating it on many corpora and found that it surpasses existing systems with average *F*-measures of 85% for entity extraction and 81% for relation extraction.

1. Introduction

Biomedical researchers are currently coping with an enormous amount of information, both in terms of raw data from experiments and a number of scientific publications describing their results. The sheer amount of information available in scientific literature already exceeds the ability of researchers to digest and is growing at an unprecedented rate. Thus the challenge is how to make effective use of these findings. Text-mining techniques have focused on how to better utilize the knowledge contained in biomedical scientific publications, accessible only in the form of natural human language. Automating the process of understanding the relevant parts of the scientific literature allows for effective searching, creates large-scale models of the relationships of biomedical entities, and enables inference of new information and hypothesis generation for biomedical research.

This paper presents an extensible, flexible text-mining system for public knowledge discovery, called PKDE4J. The main task of PKDE4J is to extract entities and their relations from unstructured text. PKDE4J extends the Stanford CoreNLP [1] and is publicly available at http://informatics.yonsei.ac.kr/pkde4j. PKDE4J differs from other text-mining techniques that are involved in entity and relation extraction in a couple of ways. First, PKDE4J is configurable so that various combinations of text-processing components can be plugged in for different tasks. For example, for the problem of gene–disease association, we use the Human Metabolome Database (HMDB) [2], UniProt [3] as gene dictionary; Medical Subject Headings (MeSH) produced by US National Library of Medicine, and KEGG Disease [4] as disease dictionary. Another layer of flexibility is that entities can be extracted either by exact or approximate match. In addition, entities can be extracted either by dictionary only or using a mixture of supervised learning and dictionary when the system is further extended. While dictionary-based entity extraction is a useful approach, it suffers from a high number of false positives, mainly caused by short names, which significantly degrade overall performance. Although this problem can be temporarily fixed by excluding short names from the dictionary, such a solution disallows for recognizing short entity names.

Second, PKDE4J provides an extensible framework for extraction. Portability is a major issue that impedes the widespread use of text-mining tools in online biological documents. Some systems extract protein–protein interactions (PPIs), others are designed to mine gene–disease relations, but none can extract both of these kinds of relations in a unified framework. Most current approaches are focused on a specific application to solve a specific kind of problem. Because there is a wide range of relation extraction tasks, a single optimized prediction model is only effective in a certain condition. For example, the simple relation extraction task of

* Corresponding author. Tel.: +82 2 2123 2416; fax: +82 2 393 8348.
E-mail addresses: min.song@yonsei.ac.kr (M. Song), krevas@yonsei.ac.kr (W.C. Kim), leedahee@yonsei.ac.kr (D. Lee), goeun.heo@yonsei.ac.kr (G.E. Heo), ky.kang@yonsei.ac.kr (K.Y. Kang).

http://dx.doi.org/10.1016/j.jbi.2015.08.008
1532-0464/

determining whether a sentence contains a relation or not requires a different model from the task of event extraction. To tackle this issue, we develop an extensible rule engine based on dependency parsing for relation extraction. A set of rules is applied to identify whether a relation exists in a sentence and to determine its relation type. Our proposed solution, a plug-and-play approach for building a rule engine, handles different extraction tasks in an efficient manner.

PKDE4J not only provides flexible and extensible extraction capability, but also achieves highly accurate performance. Our experimental results show that PKDE4J outperforms competing algorithms for both entity and relation extraction in most cases.

The remainder of this paper is organized as follows. We discuss some related works in the field of biomedical named entity recognition (NER) and relation extraction (RE). Next, we describe the architectures of NER and RE modules in the proposed system. Then we report and discuss the experimental results and compare the PKDE4J framework with other similar approaches. We conclude by summarizing the key points of our work and presenting future directions.

2. Background

2.1. Entity extraction

NER techniques applied in the biological fields can be categorized to three main approaches: rule-based, dictionary-based, and machine learning-based approaches.

Many of the rule-based systems were developed to recognize protein or gene names [5–7]. Early works use rules that are generated based on part-of-speech (POS) tags, simple dependencies [5], or grammatical or contextual features [6]. ProMiner uses regular expressions as well as contextual rules [7]. While those systems point out the over-fitting risk of their rules as a limitation, they achieve 96.7%, 92.9%, and 80.8% in *F*-measure, respectively. Recently, the rule-based approach has been in the spotlight to be adopted as a part of the whole framework to identify complex names [8].

Dictionary-based entity extraction is still the state-of-the-art approach for large-scale biomedical literature annotation and indexing. Its major advantage over the pattern-based approach is that it not only recognizes names, but also identifies unique concept identities. Among dictionary-based approaches, the exact dictionary lookup is the simplest one, but always achieves low extraction recall because a biological term often has many variants, which a single dictionary will fail to include [9]. Accordingly, some studies [10,11] have begun to implement fuzzy dictionary matching and elaborate post-processing algorithms. Use of the fuzzy match yields better results than the exact match by 3–9% of the recall, improving the *F*-measure to 53.7% for identifying names of proteins, RNAs, DNAs, cell lines, and cell types [10], and to 66.1% for recognizing protein names [11]. Despite the simplicity and straightforward nature of the dictionary-based and rule-based approaches, they still suffer from the inability to detect new terminology.

For this reason, machine learning-based approaches have recently been introduced and increasingly developed. Among a variety of machine learning algorithms, Conditional Random Fields (CRFs) are popular because they allow for the incorporation of various features that can be advantageous for the process of sequence labeling [12]. Hsu et al. merge forward and backward parsing CRF models [13], and Li et al. improve the performance of a CRF tagger with features based on a large-scale dictionary [14]. These groups achieve *F*-measures of 88.3% and 89.1%, respectively, for gene mention tagging. Campos et al. present Gimli based on CRF models with a rich set of features [15]. Gimli achieves an *F*-measure of 87.2% for recognizing gene names and 72.2% for proteins, DNAs, RNAs, cell lines, and cell types. Although machine learning techniques do fairly well, their performance can be skewed by the quality and composition of learning data. Thus, these techniques often fail to tag mentions of other biological types. In order to address this point, Kang et al. propose the use of knowledge bases such as Unified Medical Language System (UMLS) in the machine learning system, which enables a smaller set of training data for prediction [16].

2.2. Relation extraction

Three main techniques are used to discover relations between two entities in biomedical texts: co-occurrence [17,18], pattern or rule [19], and supervised learning-based approaches [20] that quantify the similarity between two entities and construct kernel functions such as an SVM classifier. In recent years, many studies have developed a variety of RE techniques such as a hybrid approach that combines two or more approaches to achieve more accurate performance by tackling the complex sentence structure of literature data. Chowdhury and Lavelli [21] propose a hybrid kernel-based approach that combines various features such as dependency pattern, trigger words, negative cues, walk features, and regular expression patterns. But *F*-measure values of the system vary depending on the corpus, indicating the approach's lack of stability and robustness.

RE systems are involved in a wide range of extraction tasks such as binary extraction and event extraction. Most event extraction systems are developed in context of the BioNLP'09 Shared Task on event extraction. Table 1 shows the list of relation and event extractor systems and their relation tasks.

RelEx [22] is an RE tool based on dependency parse trees. This approach uses simple rules with POS, noun-phrase-chunking, and dependency trees. In the relation filtering phase, it uses negation check, effector/effectee detection, enumeration resolution, and restriction to focus the domain. While RelEx takes advantages of dependency parse tree information, it only focuses on three rules: (1) effector-relation-effectee, (2) relation-of-effectee-by-effector, and (3) relation-between-effector-and-effectee, and it adapts one type of relation between genes and proteins. Befree [23] is a text-mining system to identify relations between drugs, genes, and their associated diseases. This system combines the shallow Linguistic Kernel approach with a new kernel, the Dependency Kernel, to detect relationships between two entities. The system deals with gene–disease associations by scoring them based on the public database. This in turn limits the number of relation types that can be extracted by the system.

With the increasing need to extract more detailed and complex relations between two entities, NLP techniques and methods for capturing delicate relations become more important. In addition, the dependency tree-based relation extraction technique, which is a backbone of our PKDE4J system, is applied in event extraction research where an event is characterized by normal or nominalized verbs. TEES [24] is an event extraction system, which focuses on the events between genes and proteins. For trigger and edge detection, the system defines dependency parse-based features and uses an SVM multiclass classifier. Rule-based post-processing is applied to refine the graph results. The developers recently released TEES 2.1 [25], which is based on an automated annotation scheme-learning system. The system emphasizes the unmerging classification of edge-detection process and unified site-argument representation. It shows improved performance and generalizability; however, its reliability when applied to the biomedical event extraction field is still a limitation. BExtract [26] is the rule-based event extraction system based on dependency paths between

Table 1
Relation extraction systems.

System	Domain	Publication year	Method & approach	Evaluation corpus	Relation type
RelEx [22]	Relation extraction	2007	Dependency parse-based rule	LLL, HPRD	PPI
Befree [23]	Relation extraction	2014	Kernel-based machine learning	EU-ADR, GAD	Gene–disease
TEES [24,25]	Event extraction	2009, 2013	Dependency parse-based feature & SVM	BioNLP'09, BioNLP'13	PPI
BExtract [26]	Event extraction	2009	Dependency parse-based rule	BioNLP'09	PPI
EventMine [27,28]	Event extraction	2010, 2012	Machine learning & rich feature	BioNLP'09	PPI
BioContext [29]	Event extraction	2012	Integrated event extraction method	BioNLP'09, GENIA	PPI
Bio-event meta service [30]	Event extraction	2011	Integrated event extraction method	BioNLP'09	PPI

trigger expression and entities in sentences. EventMine [27] is a pipeline system based on a machine learning approach for event extraction, which is composed of a trigger detector, an event edge detector, and an event detector. This system was recently improved [28] by adopting a new co-reference resolution technique. But it fails to overcome a key limitation of machine learning—that it inevitably learns what is dictated by the training data with inconsistent annotations. BioContext [29] is an integrated text-mining system performing entity recognition, biomolecular event extraction, and contextualization. It detects negation and speculation, and carries out event extraction by combining TEES and EventMine. Bio-event meta service [30] integrates several event extraction techniques, and the service is evaluated by users using the BioNLP'09 gold standard corpus. Most existing relation and event extraction systems appear to perform only a single type of extraction.

A comprehensive literature review of previous studies led us to choose a dictionary-based approach for NER module to target diverse entity types without worrying about the existence of training data. Unlike other dictionary-based NER systems, we further enhance both precision and recall by applying regular expressions to tokens. The RE module is designed to adopt a combination of dependency tree-based rules to overcome the limitation of the existing systems, which have restricted rules applicable only to specific situations. In other words, most of the previous entity and relation extraction systems allow a limited extent of application, while PKDE4J can extract multiple types of biomedical entities and relations comprehensibly with its extensible and flexible framework.

3. Methods

Our system consists of two major pipelines for public knowledge discovery (Fig. 1). In the first pipeline, it extracts target entities based on dictionaries by extending the Stanford CoreNLP [1]. The second pipeline applies dependency tree-based rules to sentences with two or more entities to extract relationships among those entities. Dependency tree, a well-received representation of syntactic information, encodes grammatical relations between words in a sentence with the words as nodes and dependency types as edges. An edge from one word to another represents a grammatical relation between the two. Every word in a dependency tree has exactly one parent except the root. The Stanford CoreNLP offers a variety of natural language processing tools, such as tokenization, sentence splitting, POS tagging, lemmatization, and dependency parser. These tools are executed in the predefined order of the pipeline.

3.1. Entity extraction module

We modify the Stanford CoreNLP pipeline to make it suitable for our advanced, flexible, dictionary-based entity extraction. Fig. 2 describes the steps of entity extraction.

As shown in Fig. 2, the entity extraction module of PKDE4J consists of four major sub-modules: dictionary loading, pre-processing, entity annotation, and post-processing.

3.1.1. Dictionary loading

We first configure the dictionaries, which the entity-extraction module then loads. There is no limit to the number of dictionaries that can be used, and it is possible to add more dictionaries at any time. The dictionary data are loaded with the Trie data structure, which allows for faster n-gram matching.

3.1.2. Pre-processing

3.1.2.1. Abbreviation resolution. If the input file consists of an abstract or a document with several paragraphs, not a single sentence, the module is designed to apply abbreviation resolution. For instance, if "Alzheimer disease (AD)" is written in the beginning of the input document, it changes all of the following ADs into "Alzheimer disease." This prevents the failure of n-gram matching caused by abbreviations.

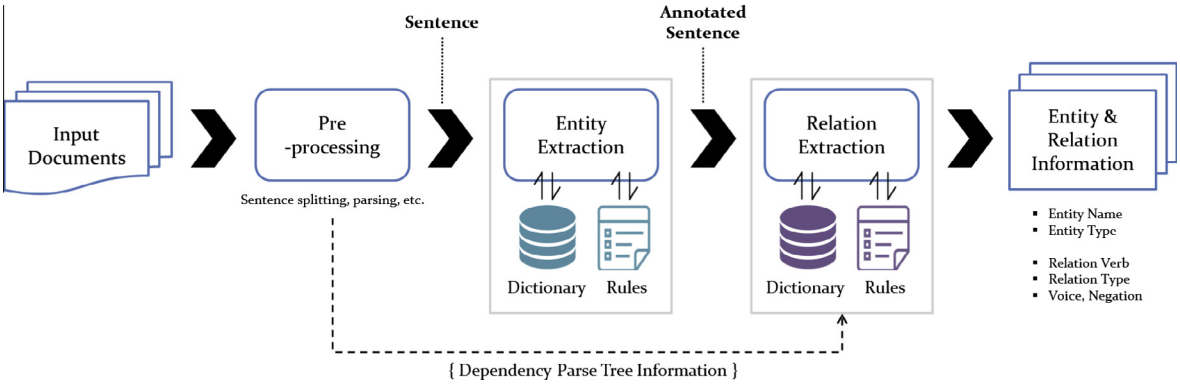

Fig. 1. Overview of system architecture.

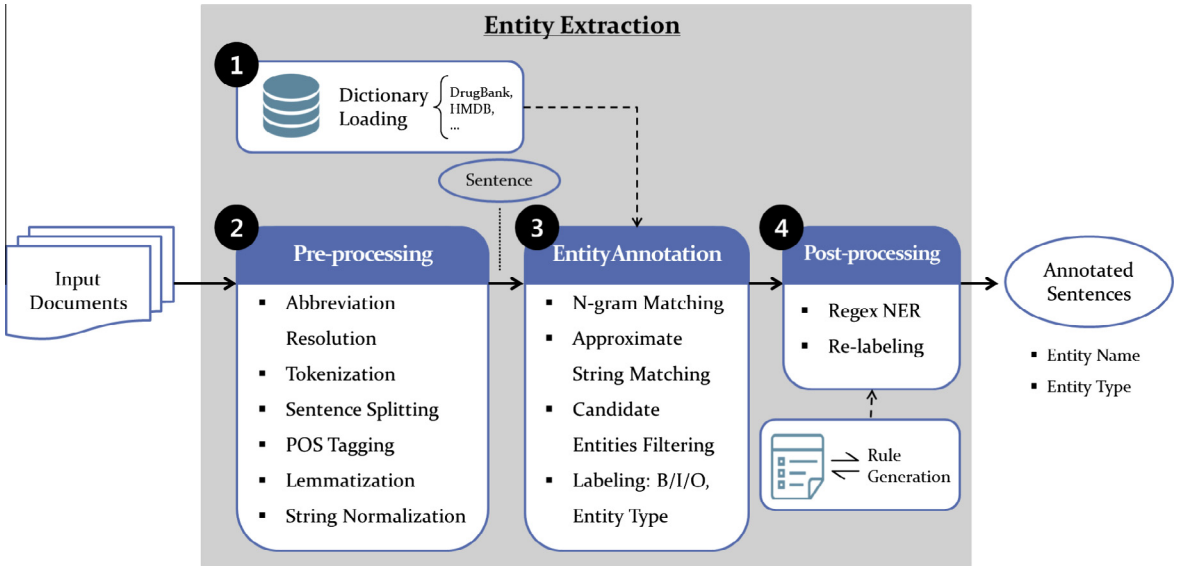

Fig. 2. System flow of entity-extraction module.

3.1.2.2. Tokenization. For tokenization, we use the Stanford PTBTokenizer implementing Penn Treebank 3 (PTB) tokenization [31]. PTBTokenizer is implemented as a finite automation by JFlex to make it an efficient, fast, deterministic tokenizer. Compared to a whitespace-based tokenizer and the statistical OpenNLP tokenizer, the Stanford tokenizer produces more fined-grained tokens. For example, it splits "(no)" into three tokens—(, no, and)—rather than treating it as a single token. Duplicated strings are merged to prevent redundant processing.

3.1.2.3. Sentence splitting. For sentence boundary detection (SBD), we use the Stanford CoreNLP's sentence splitting algorithm, which is based on a Maximum Entropy model. We trained the SBD model with the GENIA corpus [32].

3.1.2.4. Part-of-speech tagging. We use the Stanford POS tagging technique, which is based on a flexible statistical CRF model by using Gibbs sampling [33]. This strategy relaxes the requirement of exact inference, substituting approximate inference algorithms for training high-accuracy sequence models.

3.1.2.5. Lemmatization. We use the Stanford lemmatization technique that is available in the Stanford CoreNLP. This technique does full morphological analysis to accurately identify the lemma for each word. Lemmatization is similar to word stemming, but rather than yield the stem of the word, it replaces the suffix to get the normalized word form.

3.1.2.6. String normalization. We implement the string normalization method to reduce the string variation by case sensitivity and special characters like +, ∗, ;, and _. If necessary, strings with uppercase are changed to those with lowercase, and/or the appointed special characters can be removed from all the input texts and dictionary data. In the case of the special character -, it can be replaced by whitespace, allowing for the general entity name patterns.

3.1.3. Entity annotation

3.1.3.1. N-gram matching. For n-gram tokenization, we use Apache Lucene ShingleWrapper class that constructs shingles (token n-grams) from a token stream. In other words, it creates combinations of tokens as a single token. For example, the sentence "please divide this sentence into shingles" might be tokenized into shingles "please divide," "divide this," "this sentence," "sentence into," and "into shingles." The number of tokens is configurable, and we use seven grams for the experiments in consideration of the average length of entity names and the speed of entity extraction. Entities greater than seven grams are string matched after normalization.

3.1.3.2. Approximate string matching. Unlike the exact matching algorithm adopted by most dictionary-based approaches, the approximated matching technique is based on the weighted edit distance of strings from dictionary entries. We use Soft-TFIDF because it achieves highly accurate performance [34]. Soft-TFIDF is a hybrid similarity measure introduced by Cohen et al. [34] that is designed to take advantage of the good performance of TFIDF, without automatically discarding words that are not strictly identical. Soft-TFIDF combines TFIDF with Jaro–Winkler distance [35], a measure based on the number and order of the common characters between two strings.

3.1.3.3. Candidate entities filtering. To filter candidate entities, we apply POS filtering and stopword removal. During POS filtering, we remove the tokens determined as determiner (DT), adverb (RB), comparative adverb (RBR), and superlative adverb (RBS). Examples of stopwords include "one," "three," and "anyone."

3.1.3.4. Labeling. In the last stage, we choose BIO format as a labeling scheme, which indicates: B for the beginning of an entity, I for inside an entity, and O for outside an entity. Thus, tokens tagged with a letter other than O are judged as final named entities by the module. As the module recognizes multiple types of named entities, it attaches the corresponding entity type next to B and I tags as shown in Fig. 3.

3.1.4. Post-processing

For further improvement of extraction quality, we adopt entity mapping based on regular expressions to the types of entities [36] using Regex NER. It defines cascaded patterns over token sequences, providing a flexible extension of the traditional regular expression language defined over strings. We define a set of rules for each entity type that expresses several patterns of entity mentions by analyzing the corpora, and those patterns are described

Input Text: Acetylsalicylic acid can be…	
Acetylsalicylic	B-DR
acid	I-DR
can	O
be	O

Fig. 3. Example of entity-extraction result.

with BIO labels assigned in the previous stage. The rule set is then applied to the pipeline so that PKDE4J can relabel the tokens if any predefined rule is matched.

3.2. Relation extraction module

The core of the proposed relation extraction module is driven by a set of dependency parsing-based rules. Dependency-parse trees provide a useful structure for the sentences by annotating edges with dependency types, e.g., subject, auxiliary, modifier. Dependency-parse trees entail global dependencies within sentences, i.e., between words that are far apart in a sentence. Sentences of biomedical texts tend to be long and complicated and frequently mention various entities.

The relation extraction workflow (Fig. 4) extracts directed qualified relations starting from free-text sentences where two or more entities are extracted by the entity extraction module. The relation extraction module requires a list of verbs and nominalization terms that are used to describe relations of interest.

After extracting entities in a sentence, relation extraction procedures are executed to construct rules to extract relation triplets. After pre-processing, we traverse the resulting dependency tree to find relation triplets by using a predefined set of relation rules for a dependency tree. For efficiency and ease-of-use, we develop a relation rule engine extending the object-oriented strategy design pattern that is from object-oriented software engineering and enables the flexible behavior of algorithms. The object-oriented strategy design pattern is used to define a family of algorithms, encapsulate each one, and make them interchangeable within the family [37]. It is particularly useful for creating objects that represent various strategies and allowing them to be properly executed in a predefined order. In our case, a strategy indicates a dependency tree-based rule. By going through a predefined set of strategies, a sentence is examined, and a family of rules is interchangeably applied to the sentence. At the end, relation triplets are determined along with relation type, voice, and negation.

3.2.1. Relation dictionary

To extract relations, we identify a sentence's verb, which may be located between the two entities and contain relational characteristics. Then we use the classified bio-verb list to determine the relationship between entities specifically focused on the biomedical domain. We started with 398 biomedical verbs prepared by Sun and Korhonen [38] and finely tuned the list based on careful review by a biomedical expert. We classify verbs into four categories: Positive (68), Negative (54), Neutral (111), and Plain (165). Table 2 shows the bio-verb type classification. Within these categories, we sort the verbs into 10 types to extract more precise relations. If there is no verb in a sentence, we detect the nominalized form of verb that contains relationship between two entities in a sentence. There might be cases of no relation. We treat that kind of sentence as a juxtapose situation, which means the entities simply co-occur in a sentence.

3.2.2. Rule generation

In the present paper, rules for relation extraction rely mainly on syntactic deep parsing. Syntactic parsing aims to identify

Table 2
Bio-verb type classification.

Category	Number of verbs	Type	Verb example
Positive	68	Increase Transmit Substitute	Lead, Contribute, Rise Shift, Move, Migrate Supplement, Alter
Negative	54	Decrease Remove	Decline, Diffuse, Down-regulate Deplete, Abrogate, Disassociate
Neutral	111	Contain Modify Method Report	Possess, Constitute, Include Methylate, Modulate, Normalize Bleach, Centrifuge, Spin Evaluate, Analyze, Examine
Plain	165	Plain	Return, Switch, Balance

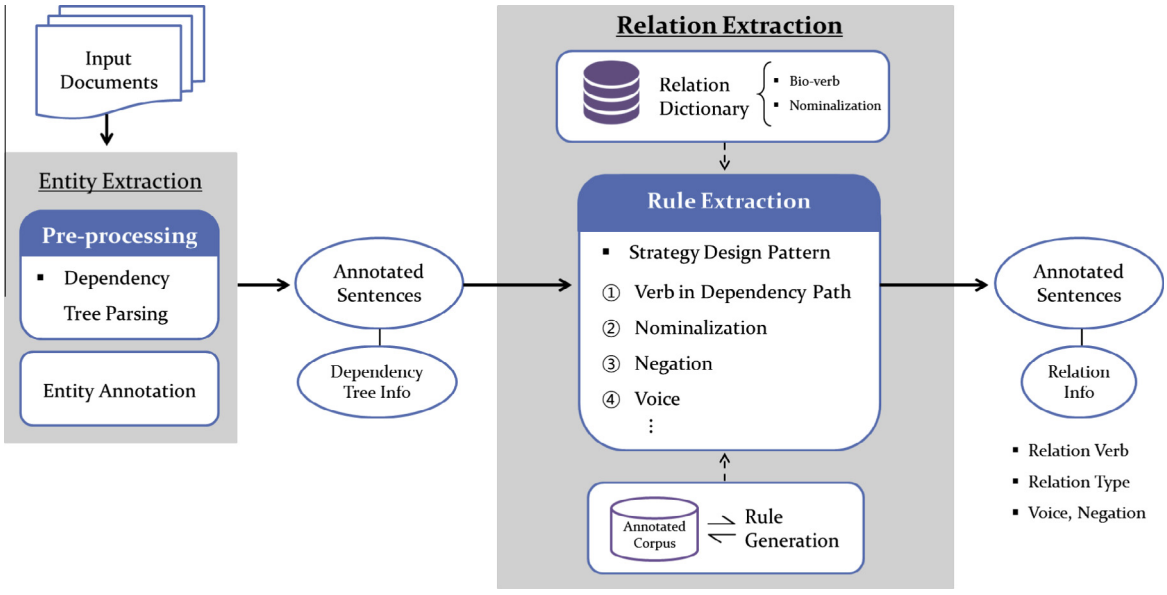

Fig. 4. System flow of relation-extraction module.

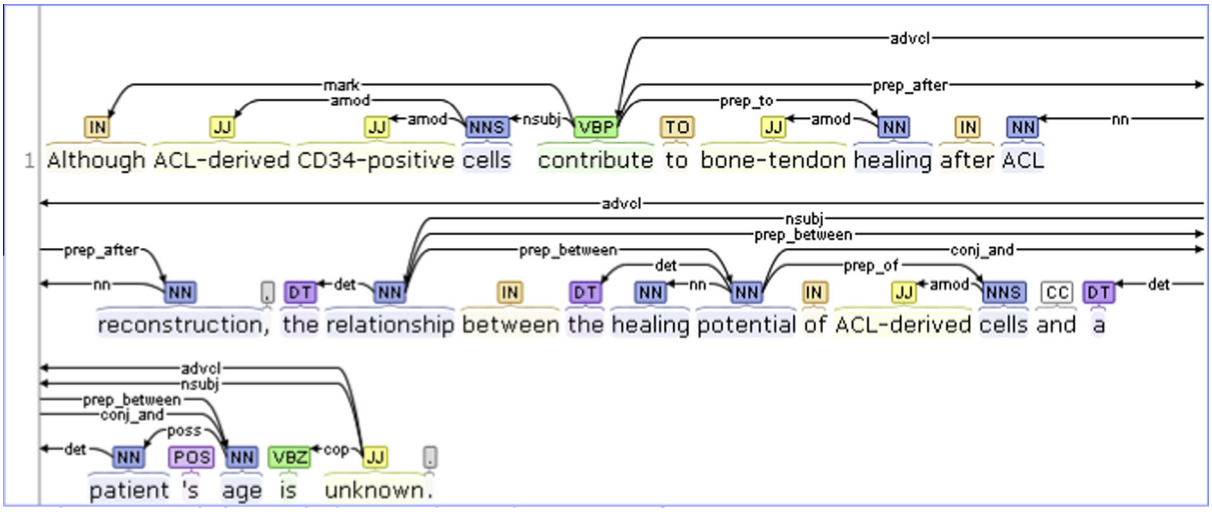

Fig. 5. Example of syntactic parsing.

grammatical structures in a sentence and captures the hidden hierarchy of the sentence for further processing. Fig. 5 shows an example of the dependency tree for the sentence: "Although ACL-derived CD34-positive cells contribute to bone-tendon healing after ACL reconstruction, the relationship between the healing potential of ACL-derived cells and a patient's age is unknown." (Collapsed CC-processed dependencies).

Given two entities, such as "ACL-derived CD34-positive cells" and "bone-tendon," the tree denotes the syntactic dependencies between them (Fig. 6). However, multiple entities and many relations are presented in a sentence. As a result, the use of the entire parse tree of the whole sentence adds unnecessary computational burden for entity extraction. Thus, we need to spot the portion of the parse tree that is pertinent to location of entities in a sentence.

For parse-tree construction, we adopt a Grammatical Relation (GR) tree, a variation of the path-enclosed tree (PT) [39]. A GR tree is the smallest common sub-tree including two or more entities and their relation.

3.2.3. Rule extraction

In our approach, a relation is typically represented as a pair of entities, linked by a directed arc. The arc is given a label usually corresponding to a semantic type. In the process of detecting relation type, we discover further information helpful for relation detection such as negation and voice, so that various meanings can be distinguished. The output of syntactic parsing includes predicate-argument relations among words. These relations are

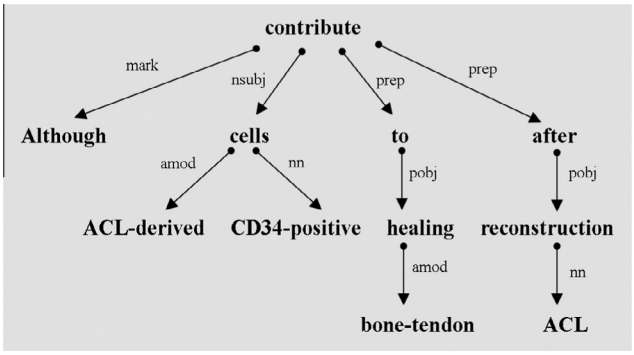

Fig. 6. Example of a dependency-parse tree.

especially useful for relation extraction when the meaning of a sentence plays a central role.

Using the dependency parser, we find the syntactic and grammatical structures of 1000 sentences in the manually annotated corpus that we create for relation extraction. Those sentences are obtained from PubMed records (biomedical literature from the MEDLINE database) by random sampling. We then analyze the results of the dependency parser and extract rules in a heuristic manner, for instance, examining common characteristics or structures that give useful tips for spotting possible relations in sentences. The validity of each rule is repeatedly tested on the same corpus. Based on the analysis result, we capture the final set of 17 rules (which we call "strategies," see Table 3) that can judge whether a sentence has a relation between two entities and how they are related. If the condition of the rule is met in an input sentence, the rule determines the relation type, voice, or negation of that sentence. We also set their combination and determine an order of consideration in the pipeline, taking their importance into account. The number of strategies is changeable per task because certain traits of each corpus can make some strategies unnecessary. Conversely, new strategies can be added to the relation extraction module due to its extensible framework.

3.2.3.1. Verb in dependency path between two entities. In the rule extraction process, a dependency path contains concatenation of relations in the path between entities in the dependency tree, including directions (e.g. "subj-> <-prep_in <-mod"). The verb-based features are designed on the assumption that verbs are often trigger words. For each verb in a dependency path, there exists a path to the left of the (lemmatized) verb, to the right, and both such as "subj-> interact <- prep_in <- mod."

Example: ... *MDA-7 elicited cell death in tumor cells* ...
{*MDA-7* [GE] -> *elicited* (POS_VB): nsubj} AND
{*elicited* (POS_VB)<- *cell death* [BP]: dobj}
⇒{Relation Verb = *associated*}

In the example above, the two entities, *MDA-7* (gene) and *cell death* (biological process), are respectively connected to the verb *elicited* in different directions in the sentence's dependency tree. So the system judges *elicited* as the relation verb for the relationship between those entities by the Verb in Dependency Path rule.

3.2.3.2. Nominalization. Nominalizations are pervasive in the molecular biology sublanguage to describe the highly nested and

Table 3
Relation strategy set.

1. Verb in dependency path
   - In the level of root (verb), show subordinate dependency relation types and directions
2. No verb in dependency path
   - Figure out whether the sentence has a verb or not between two entities
   - If not, detect nominalization strategy or weak nominalization strategy is processed
3. Detect nominalization
   - Existence of nominalized verb located in left/right position of the entity and distance from specific entity
4. Weak nominalization
   - Detect when the one entity is with preposition and the noun to which any biomedical verb is nominalized is located ahead of that preposition
5. Negation
   - Determine negation by checking existence of negative word (adjective, verb, etc.)
6. Voice (active/passive)
   - Distinguish the type of voice
7. Contain clause
   - Check if the sentence has any clauses
8. Clause distance
   - Distance between clause and entities on the left and right, the closest ones. The entities might be all to the right or to the left or divided
9. Negation clause
   - Detect negation clause (just find the clause)
10. Number of entities between entities
    - Number of recognizable entities located between two recognized entities
11. Entities in between
    - Show which entities are located in between the two main entities
12. Surface distance
    - Distance between the two recognized entities (including existing tokens and entity itself)
13. Entity counts
    - Number of entities
14. Same head
    - Check if two entities have same parent
15. Entity order
    - Order of entities
16. Full-tree path
    - Use it in dependency-parsing process
17. Path length
    - Path length from the parent node to child node

complex molecular interactions. But nominalizations are more difficult and complicated to process than verbs [34]. Current approaches provide a limited solution in terms of the nominalizations recognized and the patterns used to express their arguments. We use the dependency-tree structure to handle the extant alternations involving the argument structure of nominalizations. In particular, the following features are used to determine the relation existence and type between two entities. We identify nominalizations by templates such as [<NOMINALIZATION_TERM> <PREP (POS)> <ENTITY A> <PREP (POS)> <ENTITY B>].

3.2.3.3. Weak nominalization. Since there are many instances that do not match the strict rule of nominalization as defined above, we also set a nominalization rule that loosely defines the pattern of nominalization where only one entity meets the pattern of nominalization.

3.2.3.4. Negation. Unlike other negation techniques, our approach is not for sentence negation but focuses on negation of each pair of entities. To do that, we use the dependency relation NEG, which denotes the relation between a negation word and the word it modifies.

Example: ... *amantadine is associated negatively with Parkinson's disease* ...
{*associated* [Relation Verb] -> *negatively*: advmod} AND
{*negatively* $\in$ [List of Negative Adverbs]} AND
{*associated* $\notin$ [List of Semantically Negative Verbs]}
$\Rightarrow$ {Relation Type = Negative}

This example shows how the Negation rule decides the relation type of two entities. After *associated* is recognized as the relation verb for the relationship between *amantadine* (drug) and *Parkinson's disease* (disease) by the Verb in Dependency Path rule, the Negation rule first checks if that relation verb has the dependency of the adverb modifier (advmod). If so, the rule examines whether the modifier (*negatively* in this case) is included in the list of negative adverbs to set the relation type as negative. It finally sees if the relation verb is contained within the list of semantically negative verbs. Because *associated* is not on the list, the rule doesn't reverse the decision for the relation type and leaves it as negative.

3.2.3.5. Voice. We define the Voice rule by utilizing the dependency relation auxpass, which denotes passive auxiliary. A passive auxiliary of a clause is a non-main verb, which contains the passive information. (i.e., "DR3 was increased" auxpass (increased, was)).

3.2.3.6. Determination of clause. In determining whether there is relation between two entities, it is particularly important to identify subordinate clauses in the clause identification and the dependency-structure analyses. Clause identification is a task of recognizing the embeddedness of clauses, and of finding the starting and ending points of clauses. To this end, we use the chunk tag of "SBAR" from structural information, which denotes clauses by a subordinating conjunction. If one entity is located on the left side of the "SBAR" relation and the other entity is located on the right side, the relation type of the two entities is set to NONE as they are included in different clauses.

Example: ... *of the atrophin-1 protein, but cancer in* ...

{POS_parent node of entity = CC OR WP}

$\Rightarrow$ {Relation Type = NONE}

The above example explains that the relationship between *atrophin-1* (protein) and *cancer* (disease) is set to be NONE by the Determination of Clause rule because those entities are located in the different clauses distinguished by *cancer*'s parent node *but*, whose part of speech is recognized as the coordinating conjunction (CC).

In addition to those core relation rules described above, we provide a set of supplementary rules such as Number of entities between two entities, Surface distance, Entity counts and Same heads. Those supplementary rules are briefly explained in Table 3.

4. Results

To evaluate the PKDE4J system, we conduct a series of experiments for bio entity tagging and relation extraction and compare the performance of PKDE4J with other techniques, including Cocoa [40], Neji [41], BANNER [42], Gimli [18] for NER, RelEx [22], Hybrid [43], and BeFree [23] for RE.

4.1. Evaluation methods

To evaluate the performance of PKDE4J, we use well-accepted performance measures: precision (P), recall (R), and F-measure (F). Precision refers to the ability to avoid type I errors (false positives); recall is the ability to avoid type II errors (false negatives); and F-measure is defined as the harmonic mean of precision and recall. These indicators are presented as follows:

$$\mathbf{P} = \frac{TP}{TP + FP}, \quad \mathbf{R} = \frac{TP}{TP + FN}, \quad \mathbf{F} = \frac{2P \cdot R}{P + R}$$

where TP means true positive value, FP means false positive value, and FN means false negative value.

We evaluate PKDE4J on the test corpora for most entity or relation types. However, if the corpus does not offer separate training and test data, we randomly split the entire dataset, with 90% serving as training data and the remaining as test data. In such cases, we employ 10-fold cross-validation to obtain the valid mean values. We take the performance numbers of the comparison systems reported in the literature, or if not available, from other publications to which we already referred, where the system is included as a comparison system. The evaluation for NER and RE is carried out individually so that we can measure the performance of each module independent of the others. So when we evaluate the relation extraction module, we allow it to take a gold standard set of entity mentions to automatically extract their relationships.

4.2. Evaluation for entity extraction

4.2.1. Corpora

We evaluate the performance of entity extraction on a total of six corpora to address nine entity types (see Table 4). For cell, cellular component, species, and molecular function & biological process, we use CRAFT [44] which contains nine different biomedical concepts. CRAFT is composed of 67 full-text articles with more than 21,000 sentences. The experiments on the other five entity types are conducted with each corresponding corpus. GENETAG [45], which has annotations of genes or proteins, consists of 2000 sentences from MEDLINE abstracts. It was used for BioCreAtIvE I and II challenges. The AnEM corpus [46] involves annotations of anatomical concepts like organ, tissue, and organism from 500 biomedical documents. The NCBI Disease corpus [47] is focused on disease annotations and based on 793 PubMed abstracts with 6651 sentences. The DDI corpus [48], produced for the SemEval-2013 Task 9 DDI extraction task, is annotated with pharmacological substances from 792 DrugBank [49] texts and 232 MEDLINE abstracts. Lastly, the NaCTeM Metabolite and Enzyme corpus [50] consists of 296 MEDLINE abstracts with annotations of metabolites and enzymes.

We specifically use the test data of those corpora (if available) for a fair comparison with other NER systems. If any specific class is shown in Table 4, it means that we target that specific class for the experiments on the corresponding entity type. Otherwise, if the corpus contains annotations of one type, this type becomes a target. If it has annotations of multiple classes, we integrate all the classes included in each corpus into a single target class.

4.2.2. Dictionaries

The dictionary-based NER module of PKDE4J is designed to flexibly incorporate any dictionaries regardless of the file format. For the experiments, we combine the entity names from several dictionaries, ontologies, and freely available online databases as described in Table 5. For instance, we collect names from Cell Ontology [51], MeSH, and Gene Ontology [52] for anatomical concepts and from Lipid Maps [53] and MassBank [54] for metabolites. Through a number of preliminary experiments, we found that an unpredictable variation of performance occurs depending on the matching rate between the coverage of corpus terms and that of dictionary terms. We thus select some of the dictionaries if they were used as standards or references during the creation of corpora. Additionally, we include all entity mentions from each training corpus for a fair comparison with other trainable NER systems.

Table 4
Six corpora for performance evaluation of entity extraction module.

Entity type	Corpus; specific class
Cell	CRAFT [44]; cl
Cellular component	CRAFT [44]; go_cc
Species	CRAFT [44]; ncbitaxon
Molecular function & biological process	CRAFT [44]; go_bpmf
Gene/protein	GENETAG [45]
Anatomical concept	AnEM Corpus [46]
Disease	NCBI Disease Corpus [47]
Drug	DDI Corpus [48]; drug
Metabolite	Metabolite and Enzyme Corpus [50]; metabolite

Table 5
List and statistics of dictionaries for the experiments.

Entity type	Dictionaries	Number of unique names
Cell	Cell Ontology [51], MeSH	17,513
Cellular component	Gene Ontology [52]	4,152
Species	MeSH, DrugBank [49]	4,125
Molecular function & biological process	Gene Ontology [52]	56,271
Gene/protein	HMDB-P [2], UniProt [3]	591,435
Anatomical concept	Cell Ontology [51], MeSH, Gene Ontology [52]	24,935
Disease	MeSH, KEGG Disease [4]	9,768
Drug	DrugBank [49]	14,364
Metabolite	Lipid Maps [53], MassBank [54]	49,254

4.2.3. Experiments

While we develop the PKDE4J's entity extraction module, we measure the impact of the approximate string matching technique on the performance of entity extraction. Fig. 7 presents the results of a preliminary experiment in which we measure the performance with either exact matching or approximated matching on four entity types of the CRAFT corpus, all other conditions being equal. The threshold of edit distance is equally set as .998 for approximated matching. Fig. 7 shows that performance generally falls when the approximate matching algorithm is applied. When we analyze why precision and recall drop, we find that if the dictionary contains "Mice, Obese," for instance, the approximate matching allows it to be matched with "mice, obese" and "obese mice," while the exact matching only recognizes "mice, obese." Due to the logic of the system, which does not permit overlapped entity tagging, the use of the approximate match results in tagging "obese mice" but not "mice" for the input "obese mice," whose answer is tagging solely "mice". Such cases bring a slight reduction in true positives for the experiment with the approximate string matching in comparison with that of the exact matching.

4.2.4. Performance evaluation of entity-extraction module

Based on the result of the preliminary experiment, we configure the entity extraction module to perform from unigram to seven-gram, exact matching. The function of lemmatization is not used, while conversion to lowercase and special character removal is activated. With such settings, we evaluate the module for each of nine entity types on the corresponding test corpus and dictionary described in the Sections 4.2.1 and 4.2.2.

Fig. 8 demonstrates the evaluation results of PKDE4J's entity extraction module and other NER systems on the nine corpora. The performances of the comparison systems on CRAFT, AnEM, and NCBI Disease correspond to the report of [41]. Those on the remaining corpora refer to the original paper of each system, i.e., Gimli to [18], BANNER to [42], MetaboliNER (Dictionary and CRF) to [50], LASIGE to [55], and WBI-NER to [56].

Overall, PKDE4J's NER module shows the best performance on six types (cellular component, molecular function & biological process, gene or protein, disease, drug, and metabolite), and gained the second place on the other three types. It achieves an *F*-measure of

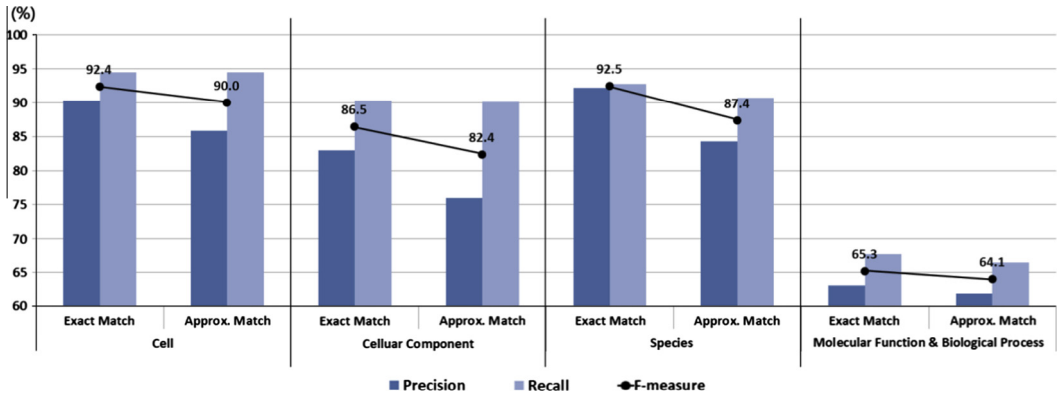

Fig. 7. Performance comparison of exact and approximate string matching on CRAFT.

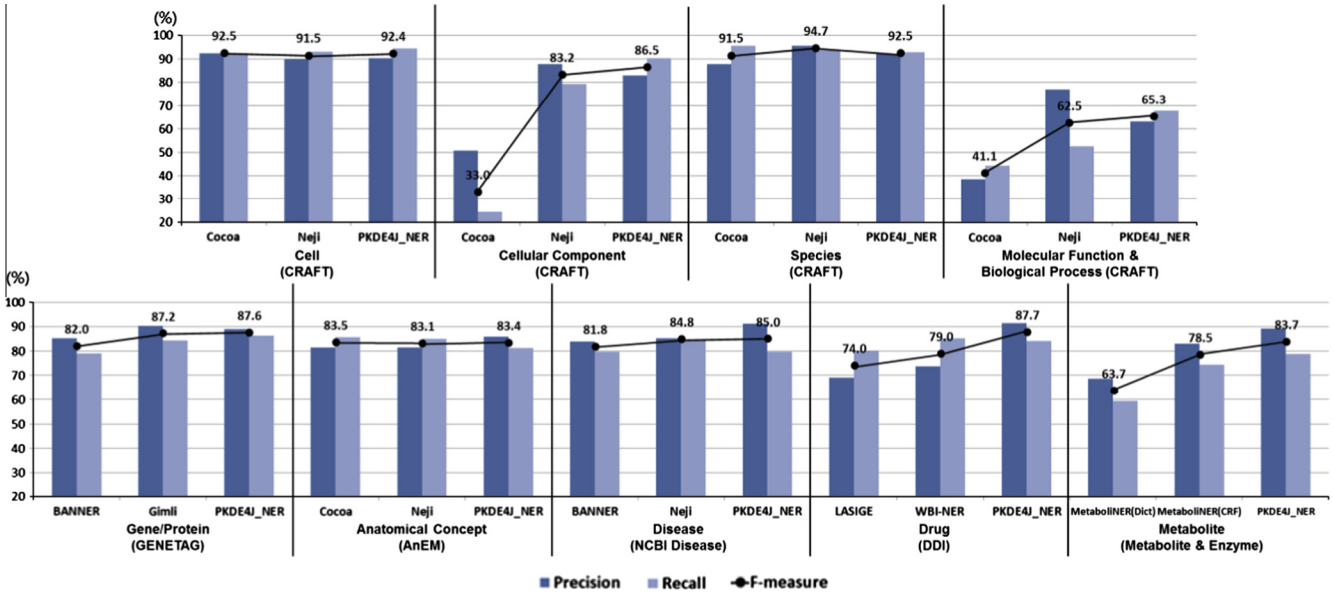

Fig. 8. Performance comparison for entity extraction on various entity types and corpora.

65.3–92.5% with an average of 85% across all the nine entity types and six corpora. The results indicate that on the entity types where the names are structurally intricate, such as molecular function & biological process (MFBP) and metabolite, the performance of our dictionary-based module remarkably surpasses that of the other systems.

The most outstanding improvement over the other systems appears on drug names with a difference in *F*-measure of at least 9% (87.7% in total). Although WEB-NER performs better than PKDE4J in terms of recall, the difference is extremely small, about 1%. The second notable improvement is achieved on metabolite names with a difference in *F*-measure of more than 5% (83.7% in total) in comparison with MetaboliNER, deriving from the enhanced precision and recall. On the other types like gene or disease, PKDE4J shows competitive results against the other NER systems, with improvements in precision or recall of up to 7%. It also has an outstanding performance over Cocoa and Neji on cellular component and MFBP names of the CRAFT corpus due to an 11% increase in recall to 66%.

However, PKDE4J outperforms only one comparison system on cell, species, and anatomical concept names. Such outcome is caused by a slight reduction of recall ranging from 1% to 4% in the cases of species and anatomical concept. On cell names, the performance difference between our system and other systems is far smaller in terms of precision, recall, and *F*-measure. But because the *F*-measure differences on all those three entity types do not exceed 1% on average, it can be said that the performance of our system is comparable with or better than the existing systems.

4.3. Evaluation for relation extraction

4.3.1. Corpora

To demonstrate the flexibility of the RE module of PKDE4J, we evaluate it using five corpora of different characteristics and relation types (Table 6). First, the BioInfer corpus [57] is focused on the relationship among proteins, genes, and RNAs. It contains 1100 sentences collected from PubMed's abstracts and the sentences have annotations about entity, entity relationship, and dependency. There are 6349 entities appearing in 1100 sentences. AIMed [58] contains 225 MEDLINE abstracts and has 1955 sentences about human proteins and genes. In the corpus, 1000 true PPIs and 4834 false PPIs are annotated. The other three corpora are general relation extraction corpus having more than two entity types. GAD [59] is a corpus which was semi-automatically annotated with gene–disease relationships. The corpus has 5329 sentences that consist of 2800 TRUE interactions to be distinguished

Table 6
Five corpora for performance evaluation of relation extraction module.

Relation type	Corpora; specific class
Protein–protein interaction	BioInfer [57], AIMed [58]
Gene–disease association	GAD [59], CoMAGC [61]; Gene–cancer
Drug–disease association	PolySearch [60]

from 2529 FALSE associations. The PolySearch [60] corpus is developed to extract relations between human disease, gene/protein, mutations (SNPs), drugs, metabolites, pathways, tissues, organs, and subcellular localizations. The CoMAGC [61] corpus is made with multifaceted annotations of gene–cancer relations for causality relations of gene/cancer. It contains 821 sentences obtained from MEDLINE abstracts and focuses on gene–cancer relations and gene expression.

4.3.2. Experiments

To investigate the impact of different combinations of rule sets on the performance of relation extraction, we measure the performance difference of the RE module by a set of different rules with the GAD corpus. Table 7 shows the five different rule sets and the series of rules belonging to each. The detailed explanations of the rule set are provided in the Methods section.

Through the use of the properties file, which allows the user to select each of the rules to be activated, we acquire *F*-measure values for different rule-set combinations in the experiment with the GAD corpus. (See Table 8 for the result on the GAD corpus and Appendix 1 on other corpora.) Baseline (B) achieves an *F*-measure of 52.2%, having the highest performance in the single rule set category while Voice and Negation (Tn) performed most poorly with an *F*-measure of 39.3%. In the category of the combination of four rule sets, the highest performance is achieved by the combination of Baseline, Clause, Nominalization, and Verb in Dependency Path (BCNV), with an *F*-measure of 74.7%, whereas the lowest performance is achieved by the combination of Baseline, Voice and Negation, Clause, and Nominalization (BTnCN), with an *F*-measure of 68.6%. The combination of all five rule sets achieves the highest performance among all combinations, with an *F*-measure of 83.8%.

4.3.3. Performance evaluation of relation extraction module

To evaluate the performance of PKDE4J's relation extraction module, we use the aforementioned test corpora, the same resources (e.g. the list of biomedical verbs, etc.), and the combination of five rule sets as a default configuration setting. For each task, we adjust the number of resources or rule sets depending on the property of each corpus. For example, the list of semantically negative verbs, originally implemented on the Verb in Dependency Path rule, is excluded in the experiments with the PPI corpora (BioInfer and AIMed), as they only annotate whether the pairs of proteins have an association or not. Another example is that the CoMAGC corpus does not consider the active or passive voice, so we evaluate the performance of the corpus without the Voice rule.

Table 7
Rule combinations used in relation extraction.

Rule set category	Acronym	Rule set
Baseline	B	Number entities between entities Entities in between Surface distance Entity counts Same head Entity order Full tree path Path length
Voice/Negation	Tn	Negation Voice
Clause	C	Contain clause Negation clause
Nominalization	N	Detect nominalization Weak nominalization
Verb in dependency path	V	Verb in dependency path No verb in dependency path

Table 8
F-measure comparison of different rule combinations.

Rule combinations	*F*-measure (%)
B	52.2
Tn	39.3
C	46.5
N	47.1
V	47.0
B + Tn + C + N (BTnCN)	68.6
B + Tn + C + V (BTnCV)	69.9
B + Tn + N + V (BTnNV)	71.1
B + C + N + V (BCNV)	74.7
B + Tn + C + N + V (BTnCNV)	83.8

We compare its performance with previous reports of comparison systems (Fig. 9). The performance of RelEx on BioInfer and AIMed can be found in [62], while that of the Hybrid system is referred to [43], which is the best performance achieved by the SVM classifier. Choi and Myaeng also presented their system's performance on the AIMed corpus in [63]. BeFree's performance on GAD is taken from [23] and compared with PKDE4J's on the same corpus. Meanwhile, we report PKDE4J's performances on PolySearch and CoMAGC alone in Fig. 9, as we were not able to find evaluations of existing RE systems evaluated using these corpora.

PKDE4J demonstrates a competitive performance in comparison with other RE systems, with *F*-measure values ranging from the high 70s to low 80s. On the BioInfer corpus of PPI type, PKDE4J achieves an *F*-measure of 82.7%, and the values of precision and recall are much higher than those of other systems, excluding the high recall (98%) of the Hybrid approach. In the experiment with AIMed, our approach achieves an *F*-measure of 77.4% whereas the system of Choi and Myaeng achieves an *F*-measure of 67%. Such results with two corpora imply that PKDE4J's RE module shows a relatively stable performance, despite the dissimilar characteristics of corpora.

In the GAD corpus annotated with gene–disease relations, PKDE4J achieves a higher *F*-measure (83.8%) compared with BeFree's performance (82.2%). Our approach achieves an *F*-measure of 84.5% for drug–gene relation types of the PolySearch corpus, and at the same time performs best among all systems in terms of *F*-measure values. Although there is no equivalent comparison system for the experiment on CoMAGC, we alternatively compare PKDE4J with a search engine developed by Lee et al. [61]. They conduct the evaluation using the CoMAGC-trained MaxEnt classifier. The accuracy of the search engine reaches 79.8% for cancer change type and 89.7% for proposition type (causality/observation), whereas the *F*-measure of our approach is 78.8%. However, it is not an apples-to-apples comparison for two reasons. First, Lee et al. use accuracy measure whereas we use *F*-measure. Second, they limit the training set to a specific type while we use the entire CoMAGC corpus.

On the whole, our experimental results with NER and RE modules tested on varied data validate the extensible and flexible attributes of PKDE4J. The system even maintains fairly good performance (greater than 83% for NER and 77% for RE in most cases) irrespective of the target type and corpus, unlike other analogous systems.

To understand the errors generated by our system, we manually examine false-positive and false-negative relations resulting from

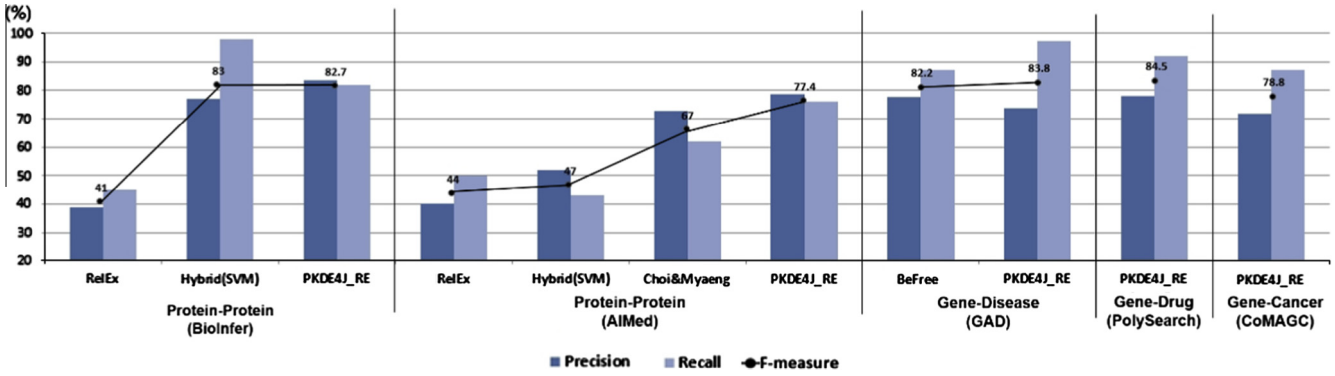

Fig. 9. Performance comparison for relation extraction on various relation types and corpora.

<False Positives>

FP1 We conclude that homozygosity for the G1514-->A mutation is exclusively responsible for the adult form of **Sandhoff disease** in this family, and that the A619-->G substitution is not a deleterious mutation but rather a common **HEXB** polymorphism.

FP2 Although there remains a possibility that the **DRD2** TaqI A polymorphism plays some role in modifying the phenotype of the disease, these results suggest that neither the A1 allele nor the homozygous A1 genotype is associated with **alcoholism**.

FP3 In conclusion, there was no association between the beta-fibrinogen -455 G/A, GP IIIa PIA1/A2, PAI-1 4G/5G, **factor V Leiden** 1691 G/A, TNFalpha -238 G/A, TNFalpha -308 G/A, IL-1alpha -889 C/T, the IL-1beta -511 C/T, MTHFR 677 C/T and eNOS 4 b/a gene polymorphisms and the risk of **restenosis** after PTCA as well as recurrent restenosis after repeated PTCA.

FP4 Although our results are negative, this was the first study to investigate **GAD** genes in **schizophrenia**, and further studies of these genes, particularly with schizophrenia subtypes, may prove valuable.

<False Negatives>

FN1 The results of our study indicate that **GABRA 3** gene might also be involved in the genetic pathophysiology of unipolar **major depressive disorder** (at least in female patients), even if the findings do not support a predominant role of GABRA 3.

FN2 Our findings suggest that the A-G polymorphism of **EGF** is involved not only in the occurrence but also in the malignant progression of **gastric cancer**.

FN3 In conclusion, the M416V polymorphism of **GYS1** gene is not associated with insulin resistance in **type 2 diabetes**.

FN4 The result of this study suggests that the **GPX1** genotype is unlikely to be associated with the risk of developing **prostate cancer**.

Fig. 10. Example sentences of the errors generated by PKDE4J's relation extraction. The two entities of the pair are highlighted in bold.

the evaluation of the GAD corpus. We identify several cases that may cause the PKDE4J's incorrect prediction. Fig. 10 contains a handful of example sentences for the errors. The first case is the occasional failure to detect the precise dependency trees for complex sentences with multiple clauses (FP1, FP2) or sentences with many "and"s (conjunctions) and commas (FP3). Another case (FP4) is that our system lacks the ability to judge the presence of relations in sentences consisting of more than two semantically connected clauses. Thirdly, some false negatives (FN1, FN2) reveal that our system requires further improvement to process special types of conjunctions such as "even if" (subordinating type) and "not only. . .but also" (correlative type). Lastly, FN3 and FN4 indicate the case where the corpus has inaccurate annotations. These cases can be considered for the further performance improvement.

5. Discussion

The largest merit of the PKDE4J system is its ability to extract relations between entities of target types. The range of the target types is boundless, as the user can construct the dictionary in any manner he chooses. The results can be analyzed in numerous ways for new discovery. The right composition of the NER dictionary can derive a high performance, as evidenced by the evaluation of the DDI corpus [48].

The dictionary-based entity-extraction module of PKDE4J works better than machine-learning approaches like Cocoa [40] and Neji [41] for the entity types that include many names that are intricately composed of multiple words. Such types include molecular function & biological process and or cellular components. The

mixture of n-gram matching as a baseline and string matching for recognizing mentions with larger gram sizes can leverage that notable point. The addition of machine-learning methods to the entity extraction module is a promising direction for future work, and may resolve the limitation of the dictionary-based method to distinguish entity mentions from conjunction words that have the same spelling as those mentions.

With respect to the relation extraction module, we found PKDE4J can cover a broader range of relations owing to a greater diversity of rules than the simple rule-based RelEx system [22]. The comparison of PKDE4J and the Hybrid system suggests that rules associated with dependency-parsing information help achieve a robust and stable performance on many corpora. Although BeFree [23] also uses the dependency-related features, the evaluation of the GAD corpus [59] demonstrates that the combinations of our system's rule sets are sufficient to achieve a better performance than machine learning approaches. Additional rule sets can be generated and integrated to address the cases mentioned above as a future work.

6. Conclusions

We propose PKDE4J, an automated system to extract entities and relations from unstructured text based on a flexible and extensible framework. PKDE4J is able to address multiple types of entities and relations. In the system's configurable environment, it is also possible to plug varied combinations of natural language processing components, as well as to add dictionaries and plentiful rule sets for recognizing accurate entities and relations. PKDE4J is thus a comprehensive, flexible, and effective text-mining system for knowledge discovery, applicable not only to the biomedical field but to other domains as well.

The competitive performance of PKDE4J is validated through the evaluation on a diversity of public corpora. Its entity extraction module achieves an *F*-measure of about 85% on average when extracting one type of named entities, while its relation extraction module achieves an *F*-measure of about 81% on average. The modules present better results than existing NER or RE solutions. Moreover, few systems exist to deal with extracting both entities and relations within a single framework, as our proposed system does.

PKDE4J can serve as the middleware for many applications. One possible application is creating a knowledge graph. With named entities and relations extracted by PKDE4J, one can build a knowledge graph in which the nodes represent entities and the edges represent relations. Once the knowledge graph is constructed, indirect linkages between source nodes and target nodes can be analyzed, and previously unknown relationships can be discovered.

At the moment, PKDE4J can be slow when handling too many types of entities and relations at once. We plan to improve the speed of PKDE4J for large-scale extraction tasks by incorporating a parallel distributed architecture such as Hadoop. We anticipate that PKDE4J will greatly contribute to the further development of similar text mining systems.

Conflict of Interest

Author states that there is no 'Conflict of Interest'.

Acknowledgments

This work was supported by the Bio-Synergy Research Project (NRF-2013M3A9C4078138) of the Ministry of Science, ICT, and Future Planning through the National Research Foundation. We would like to express our sincere thanks to the reviewers for their insightful comments on our paper.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.jbi.2015.08.008.

References

[1] C.D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S.J. Bethard, D. McClosky, The Stanford CoreNLP natural language processing toolkit, in: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, June 2014, pp. 55–60).
[2] D.S. Wishart, T. Jewison, A.C. Guo, M. Wilson, C. Knox, Y. Liu, Y. Djoumbou, R. Mandal, F. Aziat, E. Dong, S. Bouatra, I. Sinelnikov, D. Arndt, J. Xia, P. Liu, F. Yallou, T. Bjorndahl, R. Perez-Pineiro, R. Eisner, F. Allen, V. Neveu, R. Greiner, A. Scalbert, HMDB 3.0—the human metabolome database in 2013, Nucl. Acids Res. (2012) gks1065.
[3] UniProt Consortium, Activities at the universal protein resource (UniProt), Nucl. Acids Res. 42 (D1) (2014) D191–D198.
[4] M. Kanehisa, S. Goto, KEGG: kyoto encyclopedia of genes and genomes, Nucl. Acids Res. 28 (1) (2000) 27–30.
[5] K.I. Fukuda, T. Tsunoda, A. Tamura, T. Takagi, Toward information extraction: identifying protein names from biological papers, Pac. Symp. Biocomput. 707 (18) (January 1998) 707–718.
[6] D. Proux, F. Rechenmann, L. Julliard, V. Pillet, B. Jacq, Detecting gene symbols and names in biological texts, Genome Inform. 9 (1998) 72–80.
[7] D. Hanisch, K. Fundel, H.T. Mevissen, R. Zimmer, J. Fluck, ProMiner: rule-based protein and gene entity recognition, BMC Bioinform. 6 (Suppl 1) (2005) S14.
[8] P. Corbett, P. Murray-Rust, High-throughput identification of chemistry in life science texts, Computational Life Sciences II, Springer, Berlin Heidelberg, 2006, pp. 107–118.
[9] J.H. Chiang, H.C. Yu, Literature extraction of protein functions using sentence pattern mining, Knowl. Data Eng., IEEE Trans. 17 (8) (2005) 1088–1098.
[10] Z. Yang, H. Lin, Y. Li, Exploiting the performance of dictionary-based bio-entity name recognition in biomedical literature, Comput. Biol. Chem. 32 (4) (2008) 287–291.
[11] Y. Tsuruoka, J.I. Tsujii, Improving the performance of dictionary-based approaches in protein name recognition, J. Biomed. Inform. 37 (6) (2004) 461–470.
[12] T. Munkhdalai, M. Li, T. Kim, O. Namsrai, S.P. Jeong, J. Shin, K.H. Ryu, Bio named entity recognition based on co-training algorithm, in: Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on, IEEE, March 2012, pp. 857–862.
[13] C.N. Hsu, Y.M. Chang, C.J. Kuo, Y.S. Lin, H.S. Huang, I.F. Chung, Integrating high dimensional bi-directional parsing models for gene mention tagging, Bioinformatics 24 (13) (2008) i286–i294.
[14] Y. Li, H. Lin, Z. Yang, Incorporating rich background knowledge for gene named entity classification and recognition, BMC Bioinform. 10 (1) (2009) 223.
[15] D. Campos, S. Matos, J.L. Oliveira, Gimli: open source and high-performance biomedical name recognition, BMC Bioinform. 14 (1) (2013) 54.
[16] N. Kang, B. Singh, C. Bui, Z. Afzal, E.M. van Mulligen, J.A. Kors, Knowledge-based extraction of adverse drug events from biomedical text, BMC Bioinform. 15 (1) (2014) 64.
[17] R. Jelier, G. Jenster, L.C. Dorssers, C.C. van der Eijk, E.M. van Mulligen, B. Mons, J.A. Kors, Co-occurrence based meta-analysis of scientific texts: retrieving biological relationships between genes, Bioinformatics 21 (9) (2005) 2049–2058.
[18] G. Leroy, H. Chen, Genescene: an ontology-enhanced integration of linguistic and co-occurrence based relations in biomedical texts, J. Am. Soc. Inform. Sci. Technol. 56 (5) (2005) 457–468.
[19] A. Auger, C. Barrière, Pattern-based approaches to semantic relation extraction: a state-of-the-art, Terminology 14 (1) (2008) 1–19.
[20] M. Song, H. Yu, W.S. Han, Combining active learning and semi-supervised learning techniques to extract protein interaction sentences, BMC Bioinform. 12 (Suppl 12) (2011) S4.
[21] M.F.M. Chowdhury, A. Lavelli, Combining tree structures, flat features and patterns for biomedical relation extraction, in: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, Association for Computational Linguistics, April 2012, pp. 420–429.
[22] K. Fundel, R. Küffner, R. Zimmer, RelEx—relation extraction using dependency parse trees, Bioinformatics 23 (3) (2007) 365–371.
[23] À. Bravo, J. Piñero, N. Queralt, M. Rautschka, L.I. Furlong, Extraction of relations between genes and diseases from text and large-scale data analysis: implications for translational research, bioRxiv (2014) 007443.
[24] J. Björne, J. Heimonen, F. Ginter, A. Airola, T. Pahikkala, T. Salakoski, Extracting complex biological events with rich graph-based feature sets, in: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task, Association for Computational Linguistics, June 2009, pp. 10–18.
[25] J. Björne, T. Salakoski, TEES 2.1: automated annotation scheme learning in the BioNLP 2013 shared task, in: Proceedings of the BioNLP Shared Task 2013 Workshop, August 2013, pp. 16–25.
[26] H. Kilicoglu, S. Bergler, Syntactic dependency based heuristics for biological event extraction, in: Proceedings of the Workshop on Current Trends in

Biomedical Natural Language Processing: Shared Task, Association for Computational Linguistics, June 2009, pp. 119–127.
[27] M. Miwa, R. Sætre, J.D. Kim, J.I. Tsujii, Event extraction with complex event classification using rich features, J. Bioinform. Comput. Biol. 8 (01) (2010) 131–146.
[28] M. Miwa, P. Thompson, S. Ananiadou, Boosting automatic event extraction from the literature using domain adaptation and coreference resolution, Bioinformatics 28 (13) (2012) 1759–1765.
[29] M. Gerner, F. Sarafraz, C.M. Bergman, G. Nenadic, BioContext: an integrated text mining system for large-scale extraction and contextualization of biomolecular events, Bioinformatics 28 (16) (2012) 2154–2161.
[30] Y. Kano, J. Björne, F. Ginter, T. Salakoski, E. Buyko, U. Hahn, K. Bretonnel Cohen, K. Verspoor, C. Roeder, L.E. Hunter, H. Kilicoglu, S. Bergler, S. Van Landeghem, T. Van Parys, Y. Van de Peer, M. Miwa, S. Ananiadou, M. Neves, A. Pascual-Montano, A. Özgür, D.R. Radev, S. Riedel, R. Sætre, H.-W. Chun, J.-D. Kim, S. Pyysalo, T. Ohta, J.I. Tsujii, U-Compare bio-event meta-service: compatible BioNLP event extraction services, BMC Bioinform. 12 (1) (2011) 481.
[31] C. Manning, T. Grow, T. Grenager, J. Finkel, J. Bauer, PTBTokenizer, Retrieved from <http://nlp.stanford.edu/software/tokenizer.shtml> (n.d.).
[32] J.D. Kim, T. Ohta, Y. Tateisi, J.I. Tsujii, GENIA corpus—a semantically annotated corpus for bio-textmining, Bioinformatics 19 (Suppl 1) (2003) i180–i182.
[33] J.R. Finkel, T. Grenager, C. Manning, Incorporating non-local information into information extraction systems by gibbs sampling, in: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, June 2005, pp. 363–370.
[34] K.B. Cohen, M. Palmer, L. Hunter, Nominalization and alternations in biomedical language, PLoS One 3 (9) (2008) e3158.
[35] W.E. Winkler, String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage, in: Proceedings of the Section on Survey Research Methods, American Statistical Association, 1990, pp. 354–359.
[36] A.X. Chang, C.D. Manning, TokensRegex: defining cascaded regular expressions over tokens, Technical Report CSTR 2014-02, Department of Computer Science, Stanford University, 2014.
[37] B. Garbinato, R. Guerraoui, Using the strategy design pattern to compose reliable distributed protocols, in: COOTS, June 1997, pp. 221–232).
[38] L. Sun, A. Korhonen, Improving verb clustering with automatically acquired selectional preferences, in: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2–Volume 2, Association for Computational Linguistics, August 2009, pp. 638–647.
[39] M. Zhang, J. Zhang, J. Su, Exploring syntactic features for relation extraction using a convolution tree kernel, in: Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Association for Computational Linguistics, June 2006, pp. 288–295.
[40] RelAgent Technologies, Cocoa, Compact Cover Annotator for Biological Noun Phrases, Retrieved from <http://npjoint.com>, 2012.
[41] D. Campos, S. Matos, J.L. Oliveira, A modular framework for biomedical concept recognition, BMC Bioinform. 14 (1) (2013) 281.
[42] R. Leaman, G. Gonzalez, BANNER: an executable survey of advances in biomedical named entity recognition, Pac. Symp. Biocomput. 13 (January 2008) 652–663.
[43] S.J. Song, G.E. Heo, H.J. Kim, H.J. Jung, Y.H. Kim, M. Song, Grounded feature selection for biomedical relation extraction by the combinative approach, in: Proceedings of the ACM 8th International Workshop on Data and Text Mining in Bioinformatics, ACM, November 2014, pp. 29–32.
[44] M. Bada, M. Eckert, D. Evans, K. Garcia, K. Shipley, D. Sitnikov, W.A. Baumgartner, K. Bretonnel Cohen, K. Verspoor, J.A. Blake, L.E. Hunter, Concept annotation in the CRAFT corpus, BMC Bioinform. 13 (1) (2012) 161.
[45] L. Tanabe, N. Xie, L.H. Thom, W. Matten, W.J. Wilbur, GENETAG: a tagged corpus for gene/protein named entity recognition, BMC Bioinform. 6 (Suppl 1) (2005) S3.
[46] T. Ohta, S. Pyysalo, J.I. Tsujii, S. Ananiadou, Open-domain anatomical entity mention detection, in: Proceedings of the Workshop on Detecting Structure in Scholarly Discourse, Association for Computational Linguistics, July 2012, pp. 27–36.
[47] R.I. Doğan, R. Leaman, Z. Lu, NCBI disease corpus: a resource for disease name recognition and concept normalization, J. Biomed. Inform. 47 (2014) 1–10.
[48] M. Herrero-Zazo, I. Segura-Bedmar, P. Martínez, T. Declerck, The DDI corpus: an annotated corpus with pharmacological substances and drug–drug interactions, J. Biomed. Inform. 46 (5) (2013) 914–920.
[49] V. Law, C. Knox, Y. Djoumbou, T. Jewison, A.C. Guo, Y. Liu, A. Maciejewski, D. Arndt, M. Wilson, V. Neveu, A. Tang, G. Gabriel, C. Ly, S. Adamjee, Z.T. Dame, B. Han, Y. Zhou, D.S. Wishart, DrugBank 4.0: shedding new light on drug metabolism, Nucl. Acids Res. 42 (D1) (2014) D1091–D1097.
[50] C. Nobata, P.D. Dobson, S.A. Iqbal, P. Mendes, J.I. Tsujii, D.B. Kell, S. Ananiadou, Mining metabolites: extracting the yeast metabolome from the literature, Metabolomics 7 (1) (2011) 94–101.
[51] T.F. Meehan, A.M. Masci, A. Abdulla, L.G. Cowell, J.A. Blake, C.J. Mungall, A.D. Diehl, Logical development of the cell ontology, BMC Bioinform. 12 (1) (2011) 6.
[52] M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, G. Sherlock, Gene Ontology: tool for the unification of biology, Nat. Genet. 25 (1) (2000) 25–29.
[53] M. Sud, E. Fahy, D. Cotter, A. Brown, E.A. Dennis, C.K. Glass, A.H. Merrill Jr, R.C. Murphy, C.R.H. Raetz, D.W. Russell, S. Subramaniam, Lmsd: lipid maps structure database, Nucl. Acids Res. 35 (Suppl 1) (2007) D527–D532.
[54] H. Horai, M. Arita, S. Kanaya, Y. Nihei, T. Ikeda, K. Suwa, Y. Ojima, K. Tanaka, S. Tanaka, K. Aoshima, Y. Oda, Y. Kakazu, M. Kusano, T. Tohge, F. Matsuda, Y. Sawada, M. Yokota Hirai, H. Nakanishi, K. Ikeda, N. Akimoto, T. Maoka, H. Takahashi, T. Ara, N. Sakurai, H. Suzuki, D. Shibata, S. Neumann, T. Iida, K. Tanaka, K. Funatsu, F. Matsuura, T. Soga, R. Taguchi, K. Saito, T. Nishioka, MassBank: a public repository for sharing mass spectral data for life sciences, J. Mass Spectrom. 45 (7) (2010) 703–714.
[55] T. Grego, F. Pinto, F.M. Couto, LASIGE: using conditional random fields and chebi ontology, Proc. SemEval (2013) 660–666.
[56] T. Rocktäschel, T. Huber, M. Weidlich, U. Leser, WBI-NER: the impact of domain-specific features on the performance of identifying and classifying mentions of drugs, Proc. SemEval (2013) 356–363.
[57] S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, T. Salakoski, BioInfer: a corpus for information extraction in the biomedical domain, BMC Bioinform. 8 (1) (2007) 50.
[58] R. Bunescu, R. Ge, R.J. Kate, E.M. Marcotte, R.J. Mooney, A.K. Ramani, Y.W. Wong, Comparative experiments on learning information extractors for proteins and their interactions, Artif. Intell. Med. 33 (2) (2005) 139–155.
[59] K.G. Becker, K.C. Barnes, T.J. Bright, S.A. Wang, The genetic association database, Nat. Genet. 36 (5) (2004) 431–432.
[60] D. Cheng, C. Knox, N. Young, P. Stothard, S. Damaraju, D.S. Wishart, PolySearch: a web-based text mining system for extracting relationships between human diseases, genes, mutations, drugs and metabolites, Nucl. Acids Res. 36 (Suppl 2) (2008) W399–W405.
[61] H.J. Lee, S.H. Shim, M.R. Song, H. Lee, J.C. Park, CoMAGC: a corpus with multi-faceted annotations of gene–cancer relations, BMC Bioinform. 14 (1) (2013) 323.
[62] S. Pyysalo, A. Airola, J. Heimonen, J. Björne, F. Ginter, T. Salakoski, Comparative analysis of five protein–protein interaction corpora, BMC Bioinform. 9 (Suppl 3) (2008) S6.
[63] S.P. Choi, S.H. Myaeng, Simplicity is better: revisiting single kernel PPI extraction, in: Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics, August 2010, pp. 206–214.