A Single Kernel-Based Approach to Extract Drug-Drug Interactions from Biomedical Literature

Yijia Zhang*, Hongfei Lin, Zhihao Yang, Jian Wang, Yanpeng Li

College of Computer Science and Technology, Dalian University of Technology, Dalian, Liaoning, China

Abstract

When one drug influences the level or activity of another drug this is known as a drug-drug interaction (DDI). Knowledge of such interactions is crucial for patient safety. However, the volume and content of published biomedical literature on drug interactions is expanding rapidly, making it increasingly difficult for DDIs database curators to detect and collate DDIs information manually. In this paper, we propose a single kernel-based approach to extract DDIs from biomedical literature. This novel kernel-based approach can effectively make full use of syntactic structural information of the dependency graph. In particular, our approach can efficiently represent both single subgraph topological information and the relation of two subgraphs in the dependency graph. Experimental evaluations showed that our single kernel-based approach can achieve state-of-the-art performance on the publicly available DDI corpus without exploiting multiple kernels or additional domain resources.

Citation: Zhang Y, Lin H, Yang Z, Wang J, Li Y (2012) A Single Kernel-Based Approach to Extract Drug-Drug Interactions from Biomedical Literature. PLoS ONE 7(11): e48901. doi:10.1371/journal.pone.0048901

Editor: Luis Mateus Rocha, Indiana University, United States of America

Received February 28, 2012; **Accepted** September 6, 2012; **Published** November 1, 2012

Funding: This work is supported by grant from the Natural Science Foundation of China (No. 60673039 and 61070098), the National High Tech Research and Development Plan of China (No. 2006AA01Z151), Doctoral Program Foundation of Institutions of Higher Education of China (20090041110002), the Fundamental Research Funds for the Central Universities (No. DUT10JS09) and Liaoning Province Doctor Startup Fund (No.20091015). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: zhyj@dlut.edu.cn

Introduction

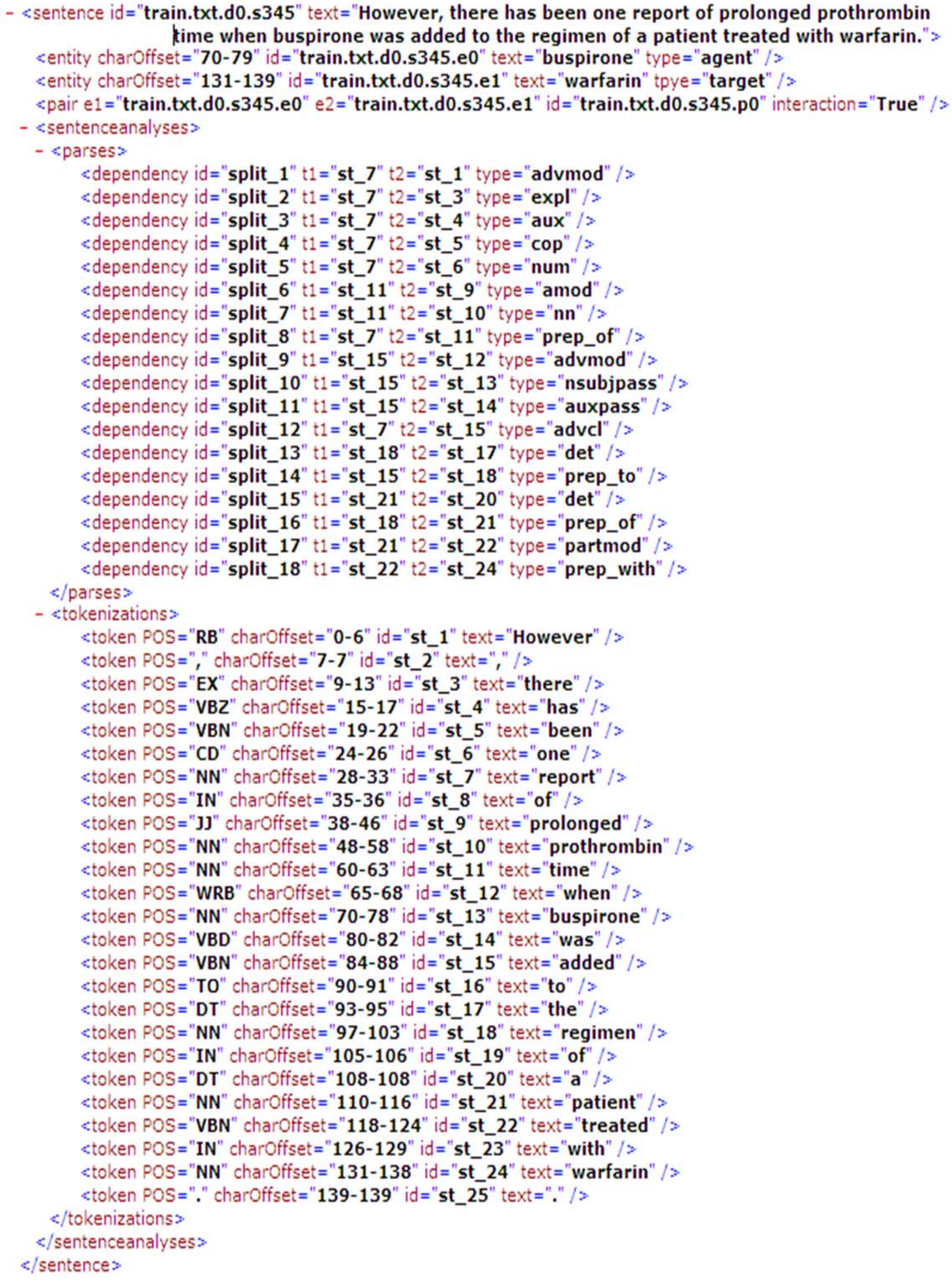
In general, a DDI occurs when one drug influences the level or activity of another drug. These DDIs in many ways affect the overall effectiveness of the drug and can sometimes pose a risk of serious side effects to patients [1,2]. Therefore, the detection of DDIs is crucial for both patient safety and health care cost control. Although health care professionals are supported by different DDI databases, the update periods of these databases are generally three years. Therefore, these databases are rarely complete [3]. Since drug interactions are frequently reported in clinical pharmacology journals and technical reports, a major source of detecting DDIs is the exponential increase in biomedical literature [4]. Thus, the automatic approach of extracting DDIs from biomedical literature can greatly contribute to the management of DDIs and allow scientists and curators early access to new discoveries.

Most biomedical relation extraction corpora have focused on genetic and protein interactions [5], such as BioInfer [6], GENIA [7] and AImed [9], rather than DDIs. Segura-Bedmar et al. created the first annotated DDI corpus [8], which provided an opportunity to use machine learning to automatically extract DDIs. In addition, the DDI Extraction Challenge 2011[8] has attracted more research interests.

One major methodology of relation extraction is pattern engineering, which adopts specific types of patterns or matching rules as the core relation discovery operation [10,11]. The patterns are mainly represented in the form of sequences of words or syntactic constituents. Blaschke et al. [12] built a set of lexical rules based on clue words. Ono et al. [13], taking into account the surface clues and part-of-speech (POS) rules, defined a group of lexical and syntactic interaction patterns for biomedical relation extraction. Fundel et al. [14] developed a RelEx system for relation extraction, which is based on more syntactic rules. However, the pattern forms are too rigid to capture semantic/syntactic paraphrases or long-range relations. Therefore, these pattern-based methods generally suffer from low recall rates.

Alternatively, with the public availability of large annotated corpora, machine learning methodology has recently become a dominant approach for relation extraction tasks. Although relationships generally involve three or more entities, most of the existing approaches in relation extraction have focused on the extraction of binary relationships, such as DDIs and protein-protein interactions (PPIs). Thus, the methodology of machine learning generally tackles the relation extraction as a classification problem. The major challenge is how to supply the learner with the semantic/syntactic information to distinguish between interactions and non-interactions [15].

Recent studies [1,2,14–19] have shown that the dependency graph and syntactic parse tree of a candidate sentence carry vital information for relation extraction tasks if their accuracy is guaranteed. Therefore, approaches such as subsequence kernels [16,17], tree kernels [18] and shortest path kernels [19] have been proposed and successfully used for relation extraction. The basic idea behind kernel methods is to map the dependency graph or syntactic parse tree into a suitable feature space. Unfortunately, due to the powerful expressiveness of graphs, defining appropriate graph kernel functions has proved difficult [20]. In order to control the complexity of kernel methods, existing kernel methods generally exploit limited information of the dependency graph



```
- <sentence id="train.txt.d0.s345" text="However, there has been one report of prolonged prothrombin
                  time when buspirone was added to the regimen of a patient treated with warfarin.">
    <entity charOffset="70-79" id="train.txt.d0.s345.e0" text="buspirone" type="agent" />
    <entity charOffset="131-139" id="train.txt.d0.s345.e1" text="warfarin" tpye="target" />
    <pair e1="train.txt.d0.s345.e0" e2="train.txt.d0.s345.e1" id="train.txt.d0.s345.p0" interaction="True" />
  - <sentenceanalyses>
    - <parses>
        <dependency id="split_1" t1="st_7" t2="st_1" type="advmod" />
        <dependency id="split_2" t1="st_7" t2="st_3" type="expl" />
        <dependency id="split_3" t1="st_7" t2="st_4" type="aux" />
        <dependency id="split_4" t1="st_7" t2="st_5" type="cop" />
        <dependency id="split_5" t1="st_7" t2="st_6" type="num" />
        <dependency id="split_6" t1="st_11" t2="st_9" type="amod" />
        <dependency id="split_7" t1="st_11" t2="st_10" type="nn" />
        <dependency id="split_8" t1="st_7" t2="st_11" type="prep_of" />
        <dependency id="split_9" t1="st_15" t2="st_12" type="advmod" />
        <dependency id="split_10" t1="st_15" t2="st_13" type="nsubjpass" />
        <dependency id="split_11" t1="st_15" t2="st_14" type="auxpass" />
        <dependency id="split_12" t1="st_7" t2="st_15" type="advcl" />
        <dependency id="split_13" t1="st_18" t2="st_17" type="det" />
        <dependency id="split_14" t1="st_15" t2="st_18" type="prep_to" />
        <dependency id="split_15" t1="st_21" t2="st_20" type="det" />
        <dependency id="split_16" t1="st_18" t2="st_21" type="prep_of" />
        <dependency id="split_17" t1="st_21" t2="st_22" type="partmod" />
        <dependency id="split_18" t1="st_22" t2="st_24" type="prep_with" />
      </parses>
    - <tokenizations>
        <token POS="RB" charOffset="0-6" id="st_1" text="However" />
        <token POS="," charOffset="7-7" id="st_2" text="," />
        <token POS="EX" charOffset="9-13" id="st_3" text="there" />
        <token POS="VBZ" charOffset="15-17" id="st_4" text="has" />
        <token POS="VBN" charOffset="19-22" id="st_5" text="been" />
        <token POS="CD" charOffset="24-26" id="st_6" text="one" />
        <token POS="NN" charOffset="28-33" id="st_7" text="report" />
        <token POS="IN" charOffset="35-36" id="st_8" text="of" />
        <token POS="JJ" charOffset="38-46" id="st_9" text="prolonged" />
        <token POS="NN" charOffset="48-58" id="st_10" text="prothrombin" />
        <token POS="NN" charOffset="60-63" id="st_11" text="time" />
        <token POS="WRB" charOffset="65-68" id="st_12" text="when" />
        <token POS="NN" charOffset="70-78" id="st_13" text="buspirone" />
        <token POS="VBD" charOffset="80-82" id="st_14" text="was" />
        <token POS="VBN" charOffset="84-88" id="st_15" text="added" />
        <token POS="TO" charOffset="90-91" id="st_16" text="to" />
        <token POS="DT" charOffset="93-95" id="st_17" text="the" />
        <token POS="NN" charOffset="97-103" id="st_18" text="regimen" />
        <token POS="IN" charOffset="105-106" id="st_19" text="of" />
        <token POS="DT" charOffset="108-108" id="st_20" text="a" />
        <token POS="NN" charOffset="110-116" id="st_21" text="patient" />
        <token POS="VBN" charOffset="118-124" id="st_22" text="treated" />
        <token POS="IN" charOffset="126-129" id="st_23" text="with" />
        <token POS="NN" charOffset="131-138" id="st_24" text="warfarin" />
        <token POS="." charOffset="139-139" id="st_25" text="." />
      </tokenizations>
    </sentenceanalyses>
  </sentence>
```

Figure 1. A sentence on the DDI corpus which underwent parsing with the Charniak and Lease parser.
doi:10.1371/journal.pone.0048901.g001

representing the sentence structure. For instance, the walk-weighted subsequence kernel [17] matches the e-walk and v-walk on the shortest path of the dependency graph, which can only represent the semantic/syntactic information of the shortest path. The tree kernel [18] can represent tree structure, but it is still not enough to completely represent all the semantic/syntactic information of the dependency graph. The all-path graph kernel [15] only computes the basic label of each node and neglects the contiguous structure of the node. The NH kernel [21] can represent the single subgraph topological information, but cannot represent the relation information of different subgraphs in the whole dependency graph. In an effort to improve the performance of these kernel methods, researchers have concentrated on combining them. Miwa et al. [22] proposed the composite kernel, which combines multiple kernels: the all-path kernel, the bag-of-words kernel and the subset tree kernel. A similar approach was used by Yang et al. [23] who proposed a weighted multiple kernel learning-based approach including the feature-based kernel, tree kernel, all-path kernel and POS path kernel. In particular, the DDI Extraction Challenge 2011 [8] showed that approaches based on multiple kernels achieved better results than other approaches.

In this paper, we propose the hash subgraph pairwise (HSP) kernel-based approach for DDIs extraction tasks. We show that a single kernel-based approach can achieve state-of-the-art performance without exploiting multiple kernels. Compared to the existing kernel approaches, the HSP kernel can efficiently represent more structural information of the dependency graph. Firstly, we represent the dependency structure and linear order of candidate sentence by a graph representation including the dependency subgraph and linear subgraph. Secondly, we construct hierarchical labels for each node of graph and use the hash operation to compute the value of the labels, which can effectively represent the contiguous structure of the subgraph. Generally, the relation between nodes or subgraphs has an impact on the classification of graphs. Thirdly, based on this hypothesis, the HSP kernel maps the graph into the subgraph pairs feature space. In particular, the HSP kernel can set each subgraph pair feature by different weights according to the distance between the subgraph pair. Therefore, the HSP kernel can represent the single subgraph topological information as well as the relation of two subgraphs. Since the whole original sentential structure contains noise for DDIs extraction tasks, we propose a graph pruning method to prune apparently noisy information from the original sentential structure and emphasize the relevant syntactic information. We evaluate the HSP kernel approach on the DDIs Extraction Challenge 2011 task corpus and compare our approach with state-of-the-art approaches.

Methods

In this section, we first present the graph representation of sentence structure. Then we introduce how to prune the graph representation to remove the noisy information from the original sentential structure. Finally, we describe in details how to use the HSP graph kernel to extract DDIs.

Graph Representation of Sentence Structure

As in recent studies of DDIs extraction [1,2,8], we tackled the task by learning a decision function that determines whether an unordered candidate drug pair has a relevant relationship in a sentence. Recent studies [1,2,14–19] have shown that biomedical relation extraction can benefit from the dependency graph or syntactic parse tree of candidate sentences. Therefore, the candidate sentence underwent parsing with the Charniak and Lease parser [24] and we then supplemented the syntactic information as the unified format proposed by Pyysalo et al. [25]. Figure 1 shows an example of a candidate sentence (DrugDDI.s345) that contains syntactic information including

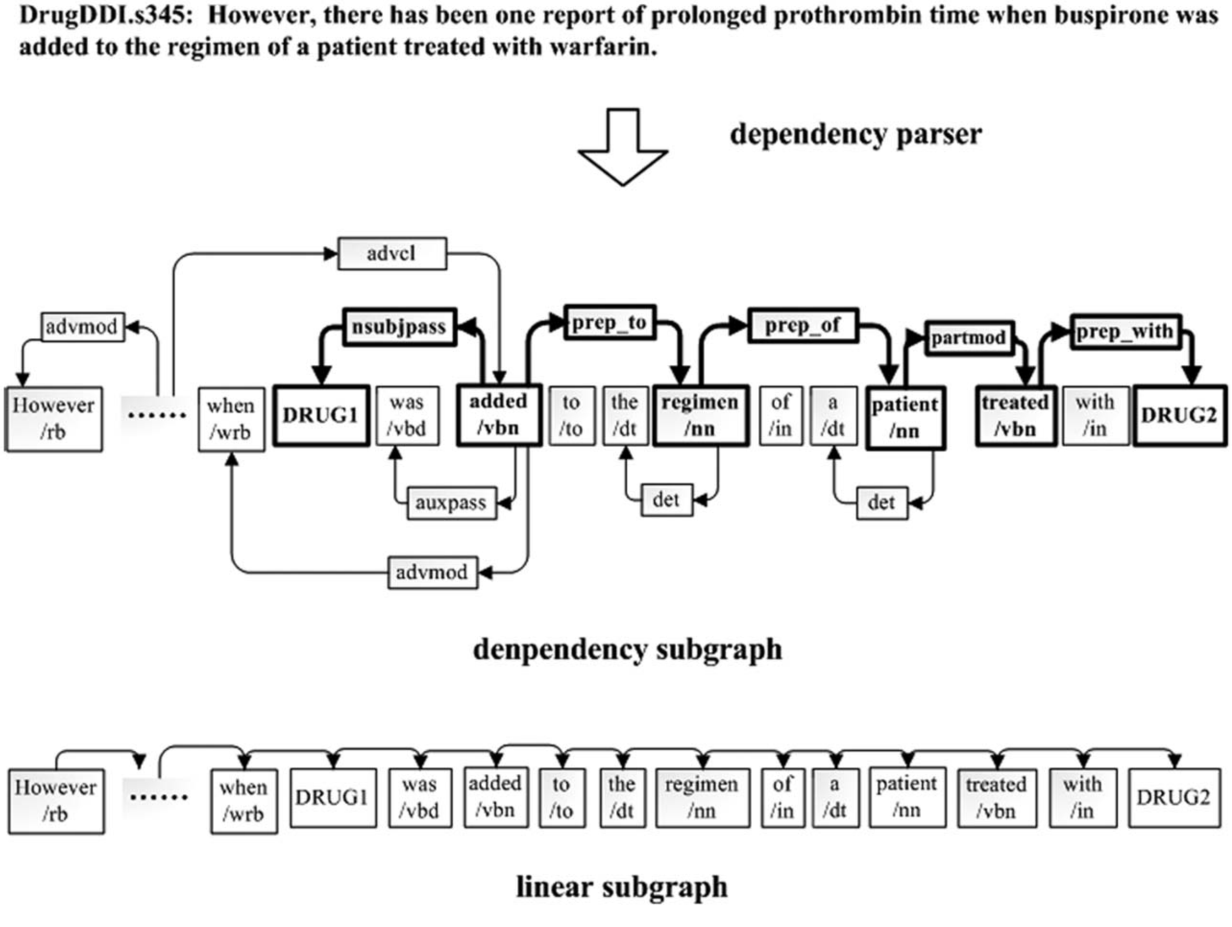


Figure 2. Graph representation of a sentence.
doi:10.1371/journal.pone.0048901.g002

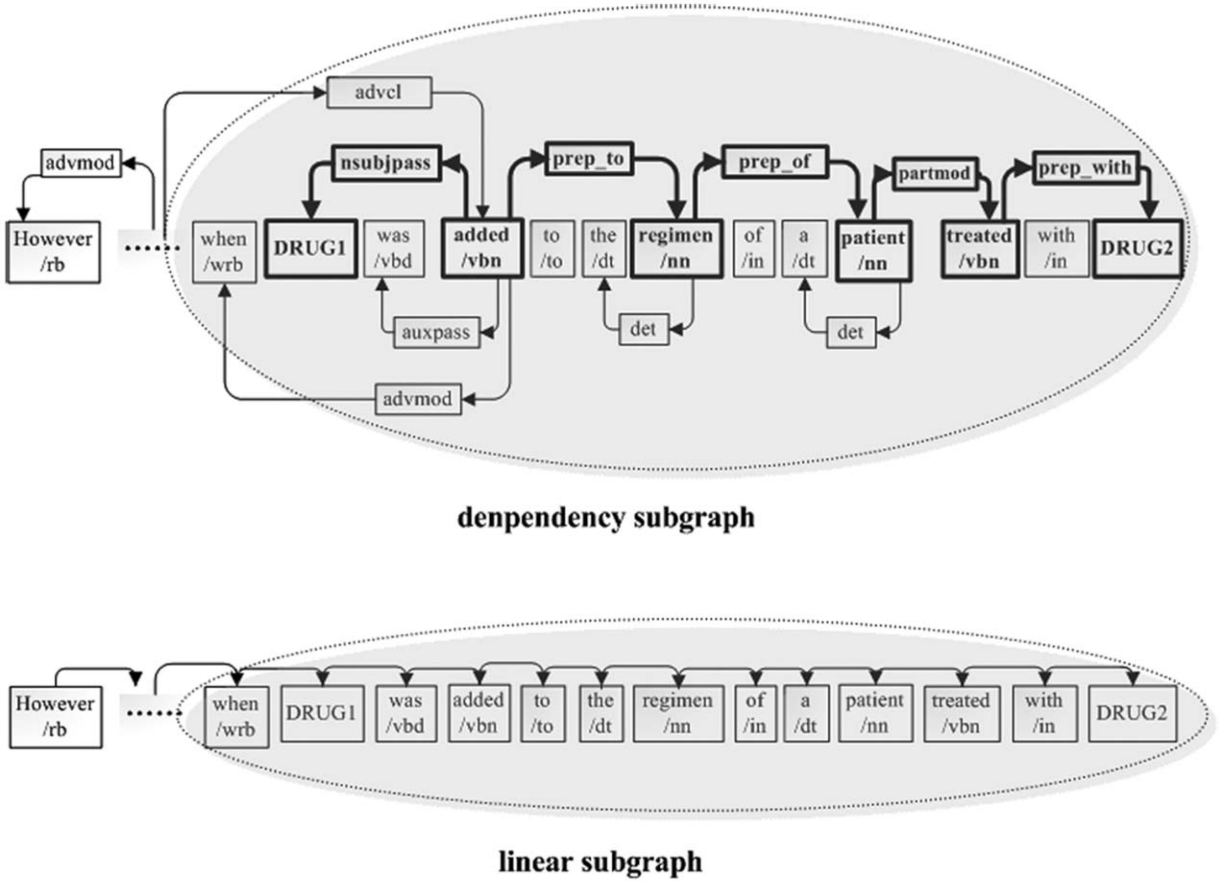


Figure 3. Illustration of pruning the dependency subgraph and linear subgraph.
doi:10.1371/journal.pone.0048901.g003

the dependency relation and POS of each token. Secondly, we represented each sentence as a directed vertex-labeled graph that consisted of a dependency subgraph and linear subgraph, which was similar to previous studies [15,21]. The dependency subgraph represented the dependency structure of the sentence and the linear subgraph represented the linear order of the sentence. Figure 2 is the graph representation generated from the sentence in Figure 1. In Figure 2, every node has a label for the token or dependency relation. For instance, "treated/vbn" denotes that the text of the token node is "treated" and the POS is "vbn", whereas "nsubjpass" denotes that the dependency relation of token nodes "DRUG1" and "added/vbn" is "nsubjpass" type. DRUG1 and DRUG2 denote candidate drug names, respectively, and the shortest path between them is shown in bold in the dependency subgraph.

HSP Graph Kernel

In recent years, various kernel methods have been employed for this task. In general, the syntactic structures around the candidate drug pairs contain more valuable information for DDIs tasks in the dependency graph. Therefore, we proposed the HSP kernel for DDIs task based on the all-path graph kernel [15], which can represent the single subgraph topological information as well as the relation of two subgraphs. We first briefly introduced the following related notion:

Let v be a set of vertices (or nodes) and ε be a set of edges (or links). Then, a graph $G = (v, \varepsilon)$ is called a directed graph if ε is a set of directed links $\varepsilon \subset v \times v$.

Definition 1 (Vertex-Labeled Graph) Let κ be a set of labels (or attributes) and $m \subset v \times \kappa$ be label allocations. Then, $G = (v, \varepsilon, m)$ is called a vertex-labeled graph.

Definition 2 (Inner Product) For two $m \times n$ matrices A and B, the inner product is defined as (1).

$$\langle A,B \rangle = \sum_{i=0}^{|m|} \sum_{j=0}^{|n|} A_{i,j} B_{i,j} \tag{1}$$

Firstly, we used a unique binary array consisting of D-bits (0 or 1) to denote each label of the graph G, such as $l0(v)=\{b_1,b_2,...,b_D\}$, where the constant D satisfies $2^D-1>>|\sum|$. Therefore, it can represent an unsigned integer value up to 2^D-1, and the node label set $\sum$ is a finite set of discrete values. Thus, we can obtain a basic bit label $l0(G)$ of G. Secondly, we followed previous studies [21,26] to define the hash operation. Let $XOR(si,sj)=si\oplus sj$ denote the XOR operation between two bit labels si and sj, which produces another binary array with each bit representing the XOR value for each digit. Let $ROTo(s)=\{bo+1,bo+2,...,bD,b1,...,bo\}$ denote the $ROTo$ operation for $s=\{b_1,b_2,...,b_D\}$ shifts the last $D-o$ bits to the left by o bits, and moves the first o bits to the right end. We can iteratively calculate the hash label for each node using (2), where $V_1^{adj}...V_d^{adj}$ denote the adjacent nodes of v. Moreover, we can distinguish between in-coming edge and out-going edge by setting

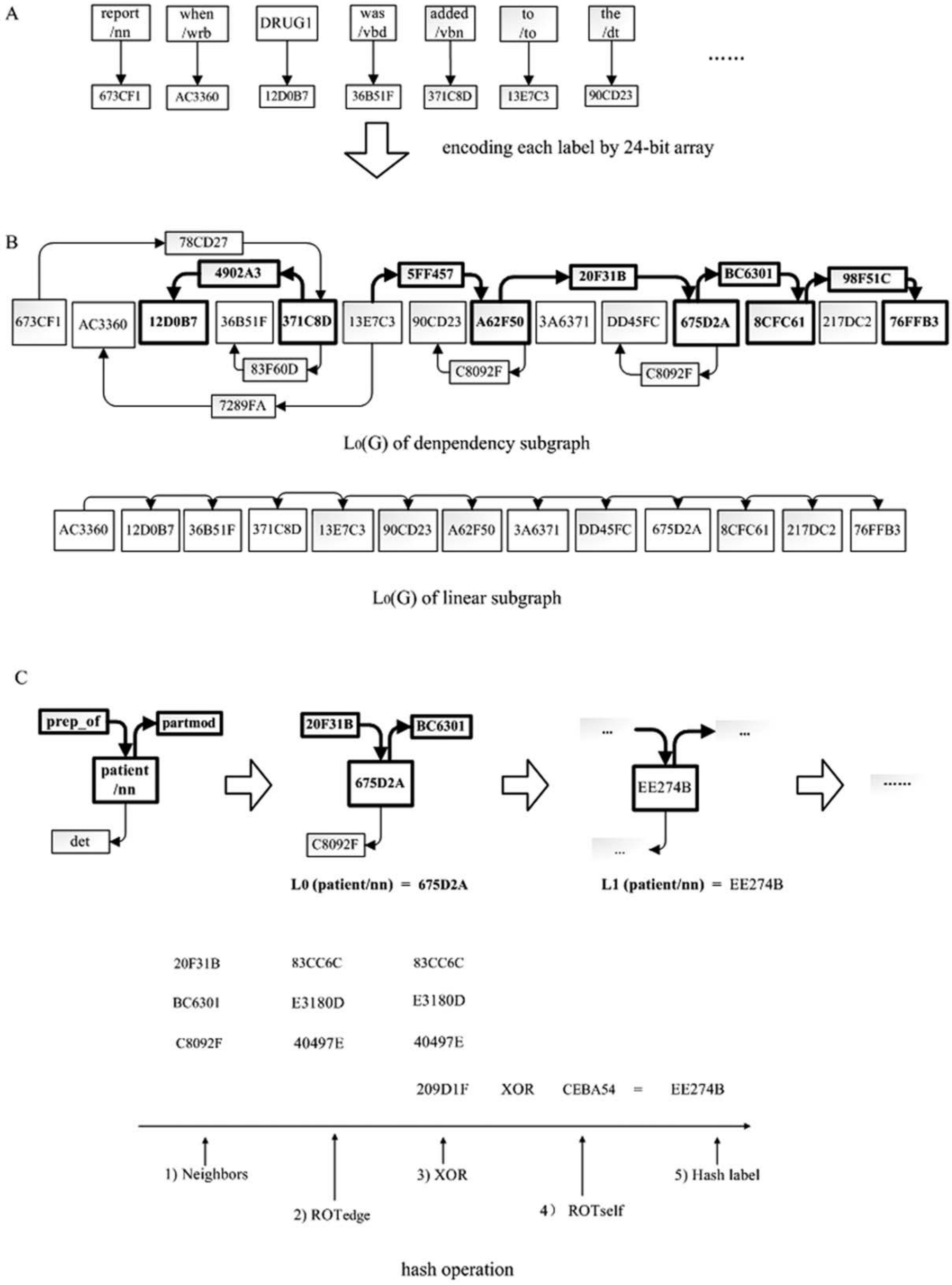


Figure 4. An example of how to compute hierarchical hash labels.
doi:10.1371/journal.pone.0048901.g004

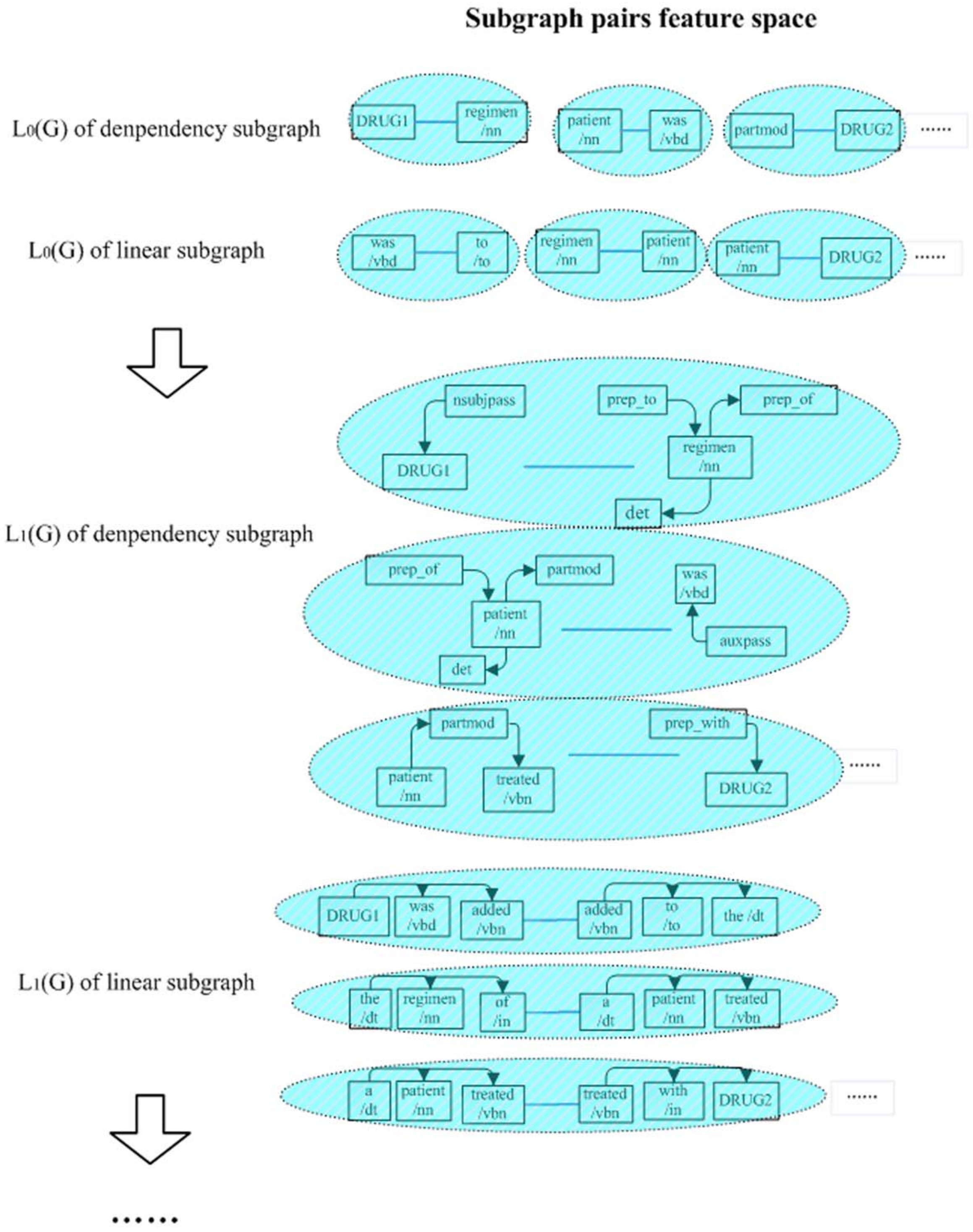


Figure 5. Illustration of subgraph pairs features mapped from $L_0(G)$, $L_1(G)$,**...** $L_r*(G)$.
doi:10.1371/journal.pone.0048901.g005

different $ROTedge$ operations. For instance, if the edge $v1\ v$ is an in-coming edge of node v, let $ROTedge = ROT2$, and if the edge $v1\ v$ is an out-going edge of node v, let $ROTedge = ROT3$.

$$li+1(v) = NH(v) =$$
$$ROT1(li(v)) \oplus (ROTedge1(li(v_1^{adj})) \oplus \cdots, \oplus ROTedged(li(v_d^{adj}))) \quad (2)$$

Let $L_1(G) = NH(L_0(G))$ denote the neighborhood hash function to a graph G. Furthermore, the neighborhood hash function can be applied iteratively as $L_i+1(G) = NH(L_i(G))$, and $L_0(G)$, $L_1(G)$, $L_i+1(G)$, is the hierarchical hash labels of G. Since the hash operation can aggregate the neighborhood nodes, $l0(v)$ only represents the basic information of node v, whereas $l1(v)$ can represent the structural information of the subgraph of radius 1. Similarly, $li(v)$ has the capability to represent the structural information of the subgraph of radius i. All bit operations such as XOR and ROT can be done in one clock, if the fixed length D is no more than the bit size of the processor architecture (32 or 64). Therefore, we can efficiently compute the hierarchical hash labels of the whole graph. Finally, based on hierarchical hash labels, the HSP graph kernel is defined as (3), where E is the adjacency matrix of G and L_0, $L_1 \ldots ,L_r$ are the hierarchical hash labels of G.

$$K(G,G') = \sum_{r=0}^{r^*} \beta^r \langle Lr\left(\sum_{i=0}^{\infty} \lambda i E^i\right) Lr^T, Lr'\left(\sum_{j=0}^{\infty} \lambda j E'^j\right) Lr'^T \rangle$$
$$= \sum_{r=0}^{r^*} \sum_{m=0}^{|k|} \sum_{n=0}^{|k|} \beta^r \left[Lr\left(\sum_{i=0}^{\infty} \lambda i E^i\right) Lr^T\right]_{mn} \left[Lr'\left(\sum_{j=0}^{\infty} \lambda j E'^j\right) Lr'^T\right]_{mn} \quad (3)$$

It is well known that each element $[E^2]ij$ gives the number of walks of length 2 from vi to vj. Similarly, each component $[E^n]ij$ gives the number of walks of length n from vi to vj. Due to the hierarchical hash labels, L_0, $L_1 \ldots ,L_r$ can express the distribution of the contiguous neighbors of each node. The matrix power series $\sum_{i=0}^{\infty} \lambda i E^i$ combines the effect of subgraph pairs with different distances. κ is the set of possible hash labels and r^* is the upper bound for the number of hierarchy. Hence, the HSP kernel can represent the full graph by mapping the graph into high dimensional subgraph pairs feature space, rather than concentrate only on special types of graphs, such as the tree kernel and string kernel. In particular, the HSP kernel can more effectively represent the relation of subgraphs. $\lambda 0, \lambda 1, \ldots \lambda i, \ldots$ are the weights sequence for subgraph pairs with different distances ($\lambda i \in \Re$; $\lambda i \geq 0$). To control the complexity of the HSP kernel, we let $\lambda 0 = \lambda^0, \lambda 1 = \lambda^1, \ldots \lambda i = \lambda^i, \ldots$ and $\lambda < 1$, and efficiently calculated the matrix power series using (4). In equation (4), matrix inversion is only the cubic time complexity.

$$\sum_{i=0}^{\infty} \lambda^i E^i = (I - \lambda E)^{-1} - I \quad (4)$$

In addition, we set the decay factor sequence $\beta^0, \beta^1, \ldots \beta^{r^*}$ for hierarchical hash labels in (3) to scale the impact of subgraph pairs with different sizes, where $\beta > 0$.

Table 1. Statistics of the DDI corpora.

	Training sets	Test sets	Total
Documents	435	144	579
Candidate DDI pairs	23827	7026	30853
Positive DDI pairs	2402	756	3158
Negative DDI pairs	21425	6270	27695

doi:10.1371/journal.pone.0048901.t001

Graph Pruning Method

The HSP kernel can accurately compute the contiguous topological information and the relative information of subgraph pairs in a graph. Therefore, the noisy information of graph representation can obviously reduce the performance of the HSP kernel. Unfortunately, previous studies [15,17,21] have shown that the whole original sentence structure contains too much noise for biomedical relation extraction tasks. To address this problem, we proposed a pruning method to prune apparently noisy information from the sentence representation and emphasize the relevant syntactic information.

Pruning methods for relation extraction were first introduced by Zhang et al. [27] who suggested seven types of pruning methods [28]. For biomedical relation extraction, the study [15] showed that the shortest path between candidate proteins in the dependency subgraph contained more vital distinguishing information. Bunescu et al. [19] followed this hypothesis and only exploited the shortest path information of the dependency graph. Furthermore, Zhou et al. [29] reported that subtrees enclosed by the shortest path between two entities describe their relation better than other subtrees, even though, in some cases, these subtrees can miss important syntactic structures. However, few approaches have been used to prune the dependency graph for biomedical relation extraction tasks. Next, we introduced the method to prune the graph representation in Figure 2. We divided the graph representation into the dependency subgraph and linear subgraph. For the dependency subgraph, we preserved only the tokens on the shortest path between candidate drug pairs, their direct neighbor tokens and dependency relations between these tokens. For the linear subgraph, we preserved the tokens between candidate drug pairs and the direct neighbor tokens of candidate drugs. Figure 3 shows how to prune the dependency subgraph and linear subgraph. To preserve the vital context, we pruned the syntactic structure out of the shadow region and preserved only the syntactic structure encompassing the candidate drug entities and between them. From Figure 3, it can be seen that we mainly preserved the syntactic structures between "when/wrb" and "DRUG2", and pruned the syntactic structure of "However, there has been one report of prolonged prothrombin time" in both the dependency subgraph and linear subgraph. Without pruning, all the syntactic structures of the sentence will intricately participate in deciding the candidate DDI. Therefore, the pruning method prunes apparently noisy information as well as emphasizes the relevant syntactic information. For example, without pruning, although the syntactic structure of "However, there…prothrombin time" contains little valuable information, it still participates in predicting the candidated DDI. Instead, after pruning, the classifier can concentrate on the syntactic structure of "when …DRUG2". In addition, the pruning method can effectively separate features when two or more interactions exist in a sentence.

Table 2. Effectiveness of parameterβ.

β	*P*	*R*	*F*	σF	AUC	σAUC
β = 0.2	55.5	64.6	59.7	2.1	88.3	1.8
β = 0.4	56.4	66.5	61.0	2.8	89.5	1.7
β = 0.6	56.8	**68.0**	**61.9**	2.4	90.1	2.2
β = 0.8	57.1	67.4	61.7	2.7	90.6	1.5
β = 1.0	56.7	65.6	60.8	1.8	**91.0**	1.4
β = 1.5	57.3	63.3	60.2	2.0	90.7	1.9
β = 2.0	**57.6**	62.5	59.9	1.7	90.2	1.2

F: *F*-score; *P*: precision; *R*: recall. σF andσAUC are the standard deviation of the *F*-score and AUC in cross validation, respectively.

doi:10.1371/journal.pone.0048901.t002

DDIs Extraction

We next present how to use the HSP graph kernel to extract DDIs from biomedical literature. Firstly, we used a *D*-bits binary array to encode each node of the graph representation. To reduce the problematic hash collisions, we chose D = 24 in our experiments. Figure 4A shows the encoding process of the graph representation in Figure 3 and the bit labels are represented by hex forms. Thus, we can obtain the $L_0(G)$ of the dependency subgraph and linear subgraph, as shown in Figure 4B. Secondly, we computed the hierarchical hash labels for each node in the dependency subgraph and linear subgraph. Figure 4C illustrates how to calculate the hash label of the node “patient/nn”. The value of $L_0(patient/nn)$ was “675D2A” which only represented the basic token information of the node. However, the value of $L_1(patient/nn)$ was “EE274B” which represented the contiguous structure information of 1-neighbors. Furthermore, we can obtain the $L_1(G)$ by calculating the whole graph representations from $L_0(G)$. Similarly, we iteratively computed the hierarchical hash labels of G, that is $L_2(G)$, …, $L_r*(G)$. Thirdly, we assigned the same weight $\lambda = 0.9$ to all edges of the graph representation and computed the similarity of two graph representations using (3).

In essence, the HSP graph kernel can map the dependency subgraph and linear subgraph into subgraph pairs feature space. In Figure 5, the shadow regions denote the subgraph pairs features which were extracted from the $L_0(G)$, $L_1(G)$,…$L_r*(G)$. For $L_0(G)$, each subgraph only contained one node, and any two subgraphs in the dependency subgraph or in the linear subgraph formed a subgraph pair feature. For instance, the nodes “DRUG1” and “regimen/nn” in $L_0(G)$ of the dependency subgraph formed a feature “DRUG1”-“regimen/nn”. For $L_1(G)$, each subgraph contained one central node and its direct neighbor nodes (1-neighbors). In other words, each subgraph can represent the topology information and syntactic information of the region with a radius of 1. We compared the feature space of the HSP kernel with the tree kernel and string kernel. On the one hand, the features extracted from the linear subgraph were similar to the features extracted by the string kernel. On the other hand, the features extracted from the dependency subgraph were more complex than the features extracted by the tree kernel. Therefore, compared with the tree kernel and string kernel, the HSP kernel can map the graph representation into both simple features and complex syntactic features. Moreover, the subgraph pair feature can represent the relation between any two complex syntactic structures in the dependency subgraph or linear subgraph. Obviously, the subgraph pair feature of $L_1(G)$ contained much more valuable information than the feature of $L_0(G)$. However, with the enlargement of the subgraph, the large-scale subgraph pair feature will cause the system to classify instance in a strict manner, which will generally lead to over-fitting problems. Thus, we should control the upper bound parameter r^* to balance the performance of the HSP kernel for different tasks.

Experimental Setting

We evaluated our method using DDI Extraction Challenge 2011 corpora [8] which is the first publicly available corpora for DDI extraction tasks. The statistics of the DDI corpora are listed in Table 1, which contains 579 documents and 30853 candidate DDIs pairs. These documents were randomly selected from the DrugBank database, which were split into training sets and test sets. In addition, all sentences in the documents underwent parsing with the Charniak and Lease parser [24], and the syntactic information was added similar to the example shown in Figure 1. The implement of our method with the name DDI_Extraction_-Tool.zip is available in Supplementary Information.

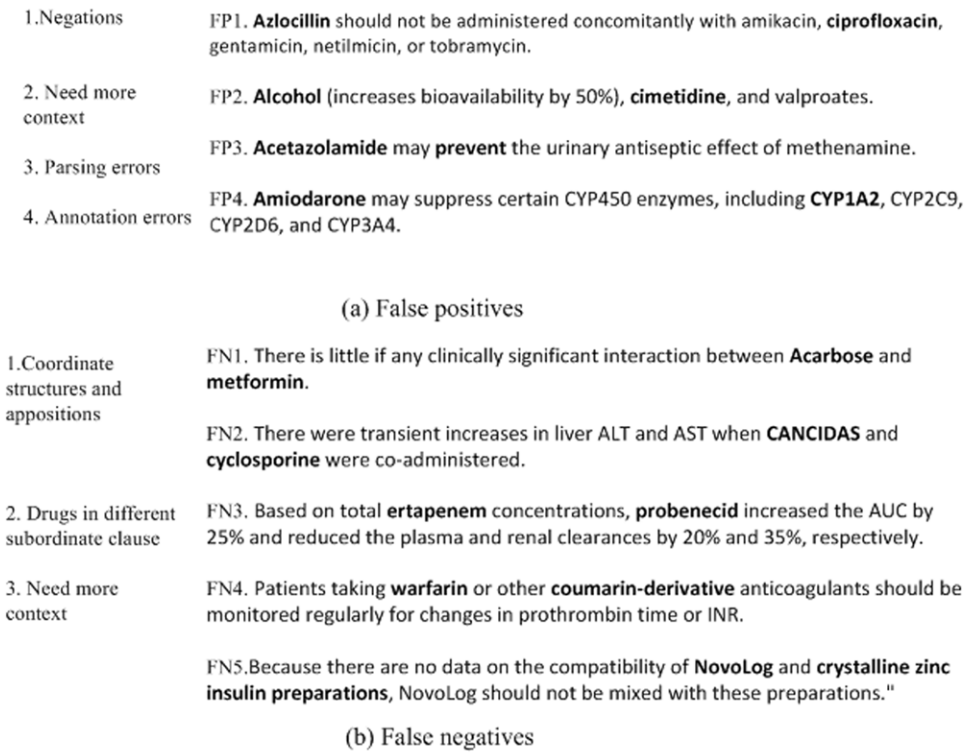
To keep our evaluation metrics the same as the DDI Extraction Challenge 2011 task [8], we optimized the parameters of our approach for DDIs extraction tasks by conducting 10-fold cross validation on the training datasets, and then tested the test datasets. This guaranteed the maximal use of the available data and allowed a comparison with the other approaches. We implemented the HSP kernel with the user defined kernel interface of SVM-light (http://svmlight.joachims.org/). Similar to previous studies [15,21], we empirically estimated the regularization parameters of SVM (C-values) on training datasets.

Table 3. Performance of our approach in comparison with other approaches.

Approach	*TP*	*FP*	*FN*	*TN*	*P*	*R*	*F*	*Acc*	*MCC*	*AUC*
WBI [31]	543	354	212	5917	60.5	71.9	**65.7**	91.9	**61.5**	-
Our approach	508	297	248	5973	**63.1**	67.2	65.1	**92.2**	60.8	**92.4**
LIMSI-FBK [32]	532	376	223	5895	58.6	70.5	64.0	91.5	59.5	-
FBK-HLT [33]	529	377	226	5894	58.4	70.1	63.7	91.4	59.2	-
UTurku [34]	520	376	235	5895	58.0	68.9	63.0	91.3	58.4	-
BNBNLEL [36]	420	266	335	6005	61.2	55.6	58.3	91.5	53.6	-
Naive Bayes approach	603	786	153	5484	43.4	**79.8**	56.2	86.6	52.3	84.1
SVM approach	382	346	374	5924	52.5	50.6	51.5	89.8	45.8	87.2

F: *F*-score; *P*: precision; *R*: recall; *Acc*: accuracy; *MCC*: Matthews correlation coefficient.

doi:10.1371/journal.pone.0048901.t003



1.Negations

FP1. **Azlocillin** should not be administered concomitantly with amikacin, **ciprofloxacin**, gentamicin, netilmicin, or tobramycin.

2. Need more context

FP2. **Alcohol** (increases bioavailability by 50%), **cimetidine**, and valproates.

3. Parsing errors

FP3. **Acetazolamide** may **prevent** the urinary antiseptic effect of methenamine.

4. Annotation errors

FP4. **Amiodarone** may suppress certain CYP450 enzymes, including **CYP1A2**, CYP2C9, CYP2D6, and CYP3A4.

(a) False positives

1.Coordinate structures and appositions

FN1. There is little if any clinically significant interaction between **Acarbose** and **metformin**.

FN2. There were transient increases in liver ALT and AST when **CANCIDAS** and **cyclosporine** were co-administered.

2. Drugs in different subordinate clause

FN3. Based on total **ertapenem** concentrations, **probenecid** increased the AUC by 25% and reduced the plasma and renal clearances by 20% and 35%, respectively.

3. Need more context

FN4. Patients taking **warfarin** or other **coumarin-derivative** anticoagulants should be monitored regularly for changes in prothrombin time or INR.

FN5.Because there are no data on the compatibility of **NovoLog** and **crystalline zinc insulin preparations**, NovoLog should not be mixed with these preparations."

(b) False negatives

Figure 6. Examples of false negatives and false positives generated by our approach.
doi:10.1371/journal.pone.0048901.g006

The majority of DDI extraction system evaluations use the balanced *F*-score measure for quantifying the performance of the systems, which is defined as *F*-score = (2*PR*)/(*P*+*R*), where *P* denotes precision and *R* denotes recall. In addition, we reported the AUC measure [30] and MCC measure [37], which have been recommended for performance evaluation [15,21,23,38].

Results and Discussion

Performance on Training Datasets

Firstly, we set the parameter r^* in (3), which is the upper bound for the number of hierarchy. In general, an increase in the hierarchy of hash labels will cause the HSP kernel to compute more large-scale subgraph pairs and the system classifies each instance in a strict and detailed manner. Therefore, when the value of r^* is too large, over-fitting problems will generally occur particularly for biomedical relation extraction tasks. After preliminary experiments, we set $r^*=2$ in our experiment.

Secondly, we investigated the effect of decay factor β in (3) for DDIs extraction tasks, which can balance the degree of contribution of subgraph pairs with different sizes in the HSP kernel computation. Table 2 shows the evaluation results on the training datasets and the value in bold is the highest value of each column. It is obvious that parameter β influenced the overall performance of DDIs extraction. The gap between the best and worst *F*-score was 2.2%. In particular, our approach achieved the best performance (an *F*-score of 61.9%), when $\beta=0.6$. From Table 2, it can be seen that the precision can be improved when β is increased (from 55.5% to 57.6%). Similarly, the recall can be improved significantly when β ranged from 0.2 to 0.6. However, when β ranged from 0.6 to 2.0, the recall dropped sharply. The main reason for this is that the increase in β promoted the weights of larger subgraph pairs for the HSP kernel. Since large subgraph pairs contain more syntactic/semantic structural information than small subgraph pairs, this caused our approach to classify each instance in a strict and detailed manner. Therefore, the increase in β generally contributes to the precision. However, if β is too large, it will reduce the recall drastically. In addition, our approach achieved best AUC of 91.0% at $\beta=1.0$, which was similar to the change in *F*-score as a whole. Consequently, we choose $\beta=0.6$ as the optimal parameter for DDIs extraction tasks.

Performance of our Approach Compared to Other Approaches

We tested our approach on test datasets using optimal parameters. In recent years, some kernel methods have been proposed and successfully applied to biomedical relation extraction. However, most of these studies focused on PPI extraction, and not DDI extraction. Although PPI extraction is similar to DDI extraction, there are some differences between them. For instance, the terminological specificity and the way researchers report their findings in different biomedical domains vary considerably. Therefore, the results from PPI extraction do not necessarily extrapolate to DDI extraction. To evaluate the performance of our approach for DDI extraction, we compared our approach with other approaches (Table 3) which included the top rank approaches reported in the DDI Extraction-2011 Challenge and other simple machine learning approaches, such as SVM and naive Bayes (NB). We built rich features for the SVM and NB approaches and empirically estimated the regularization parameters of SVM (C-values). The rich features consisted of bag-of-words (BOW) features, bigrams features and trigrams features. The NB approach achieved a high recall of 79.8%, but overall both the NB approach and SVM approach were inferior to our approach. For instance, our approach achieved an F-score of 65.1% which was an 8.9% point margin compared to the NB approach and a 13.6% point margin compared to the SVM approach. These results indicated that the common features such as BOW features and word-gram features are not enough for DDI extraction tasks, because the vital syntactic information of the dependency graph cannot be directly mapped into the feature space.

It is known that the combination of multiple kernels is the best option for improving the effectiveness of kernel-based approaches [22,23]. From Table 3, it can be seen that the WBI approach [31] achieved best performance (an *F*-score of 65.7%, a recall of 71.9% and a MCC of 61.5%), this approach exploits the combination of several kernels and a case-based reasoning system using a voting approach. In addition, the WBI approach also achieved best performance in the DDI Extraction 2011 Challenge. Similarly, the LIMSI-FBK approach [32] and the FBK-HLT approach [33] achieved competitive performances (*F*-score of 64.0% and 63.7%, respectively), and both approaches benefited from compositing several kernels including the MEDT kernel, PST kernel, and SL kernel. In addition, the UTurku approach [34] exploits the domain knowledge such as DrugBank [35], and achieved an *F*-score of 63.0% and a MCC of 58.4%. The BNBNLEL approach [36] constructs rich features and uses random forests to extract DDIs. However, due to a low recall of 55.6%, the BNBNLEL approach only achieved an *F*-score of 58.3% and a MCC of 53.6%. Compared with the above approaches, our single kernel-based approach achieved the best precision (63.1%) and the best accuracy (92.2%). Moreover, our approach achieved an *F*-score of 65.1% and a MCC of 60.8%, which was only slightly inferior to the WBI approach. In particular, our approach was obviously superior to the LIMSI-FBK approach [32] and FBK-HLT approach [33], which are composed of the tree kernel, context kernel and other kernels. This indicated that our approach can more accurately compute and represent the syntactic structural information than other such kernels for DDIs extraction tasks. However, we also note that our approach only achieved a recall of 67.2%, which was far below the multiple kernel-based approaches. This was mainly because the multiple kernel approaches can take into account richer features than the HSP kernel. For instance, as the WBI [31] approach consists of three kernels, it can classify each candidate DDI based on all-paths graph features, shortest path features and shallow linguistic features. Overall, we can only use the HSP kernel to achieve state-of-the-art performance rather than combine several kernels or exploit additional domain resources.

Error Analysis

Finally, we manually analyzed drug mention pairs which were not correctly classified by our approach. Evaluated using the final test set, our approach made a total of 545 errors, 297 of which were false positives and 248 of which were false negatives. Figure 6 shows the principal causes for the false positives and false negatives generated. The two drugs of each candidate pair are in bold.

The most frequent cause of false positives is our approach was the failure to identify negation expressions. In Figure 6a, FP1 is an example of these false positives. Therefore, a possible approach to improve performance is to introduce a pre-processing step for negation expressions. Another frequent cause of false positives is the DDI cannot be verified without the context. For instance, in Figure 6a FP2, there is very little information in the sentence and we could not verify the interaction between "Alcohol" and "cimetidine" without context. Furthermore, some false positives are caused by corpus errors. In FP3, "prevent" is a verb, but the corpus treats "prevent" as a drug due to parsing error. Moreover, according to the sentence FP4, the DDI between "Amiodarone" and "CYP1A2" should be annotated by "True", however, the corpus annotates the DDI with "False".

With regard to false negatives, most errors are caused by coordinate structures and appositions. In Figure 6b, FN1 and FN2 are two examples of these false negatives. Further studies should be performed on coordinate structures and appositions. In addition, it is difficult for our approach to deal with complex sentences in which two drugs are in different subordinate clauses. For example, our approach failed to verify the DDI in FN3. As in false positives, some false negatives are due to the need for more context information. In FN4 and FN5, we could not verify the DDIs without more context information. Therefore, such candidate DDIs should be taken out of the DDI corpus.

Conclusion

In this paper, we propose a single kernel-based approach to automatically extract DDIs from biomedical literature. To preprocess the dependency graph, we applied a novel pruning method to prune apparently noisy information and emphasize the relevant syntactic information. The experimental results demonstrated that our approach can effectively represent the syntactic structural information of the dependency graph. Furthermore, it is encouraging to see that our single kernel-based approach was comparable to the top rank multiple kernel-based approaches, and achieved state-of-the-art performance. Our major contributions to this research field are:

1. We proposed the HSP kernel approach for DDIs extraction tasks and evaluated our approach on a publicly available DDI corpus.
2. We proposed a novel pruning method to prune apparently noisy information of the dependency graph and emphasize the relevant syntactic information.
3. We evaluated our approach on a publicly available DDI corpus and compared the performance of our approach with state-of-the-art approaches.
4. The experimental results indicated that state-of-the-art performance can also be achieved by the single kernel approach.

Supporting Information

Tool S1 DDI_Extraction_Tool.
(ZIP)

Author Contributions

Conceived and designed the experiments: YZ HL ZY. Performed the experiments: YZ JW YL. Analyzed the data: YZ YL. Wrote the paper: YZ.

References

1. Segura-Bedma I, Martínez P, Pablo-Sánchez C(2011)Using a shallow linguistic kernel for drug–drug interaction extraction. Journal of Biomedical Informatics 44:789–804.
2. Segura-Bedmar I, Martínez P, Pablo-Sánchez C (2011) A linguistic rule-based approach to extract drug-drug interactions from pharmacological documents in biomedical texts. BMC Bioinformatics 12: S1.
3. Rodríguez-Terol A, Camacho C (2009) Others, calidad estructural de las bases de datos de interacciones. Farmacia Hospitalaria 33:134.
4. Segura-Bedmar I, Martínez P, Segura-Bedmar M (2008) Drug name recognition and classification in biomedical texts: a case study outlining approaches underpinning automated systems. Drug Discov Today 13: 816–823.

5. Zweigenbaum P, Demner-Fushman D, Yu H, Cohen K (2007) Frontiers of biomedical text mining: current progress. Briefings in Bioinformatics 8:358–375.
6. Pyysalo S, Ginter F, Heimonen J, Björne J, Boberg J, et al. (2007) BioInfer: A Corpus for Information Extraction in the Biomedical Domain. BMC Bioinformatics 8: 50.
7. Kim D, Ohta T, Tsujii J (2008) Corpus Annotation for Mining Biomedical Events from Literature. BMC Bioinformatics 9:10.
8. Segura-Bedmar I, Martínez P, Sanchez-Cisneros D (2011) The 1st DDIExtraction-2011 challenge task: Extraction of Drug-Drug Interactions from biomedical texts. Proc of DDIExtraction-2011 challenge task Spain. 1–9.
9. Bunescu R, Ge R, Kate R, Marcotte E, Mooney R, et al. (2005) Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. Artificial Intelligence in Medicine 33: 139–155.
10. Blaschke C, Valencia A (2002) The frame-based module of the suiseki information extraction system. IEEE Intelligent Systems 17: 14–20.
11. Corney D, Buxton B, Langdon, Jones D (2004) BioRAT: extracting biological information from full-length papers. Bioinformatics 20: 3206–3213.
12. Blaschke C, Andrade M, Ouzounis C, Valencia A (1999) Automatic extraction of biological information from scientific text: protein-protein interactions. Proc Int. Conf. Intell. Syst. Mol. Biol. 60–67.
13. Ono T, Hishigaki H, Tanigam A, Takagi T (2001) Automated extraction of information on protein–protein interactions from the biological literature. Bioinformatics 17: 156–161.
14. Fundel K, Kuffner R, Zimmer R (2007) RelEx-Relation extraction using dependency parse trees. Bioinformatics 23: 365–371.
15. Airola A, Pyysalo S, Björne J, Pahikkala T, Ginter T, et al. (2008) All-Paths graph kernel for protein-Protein interaction extraction with evaluation of cross-corpus learning. BMC Bioinformatic 9: S2.
16. Bunescu R, Mooney R (2006) Subsequence kernels for relation extraction. Proc 18th Advances in Neural Information Processing Systems.: 171–178.
17. Kim S, Yoon J, Yang J, Park S (2010) Walk-Weighted subsequence kernels for protein-protein interaction extraction. BMC Bioinformatics 11: 107.
18. Zelenko D, Aone C, Richardella A (2003) Kernel methods for relation extraction. Journal of Machine Learning Research 4: 1083–1106.
19. Bunescu R, Mooney R (2005) A shortest path dependency kernel for relation extraction. Proc Human Language Technology Conf. and Conf. Empirical Methods in Natural Language Processing Association for Computational Linguistics. 724–731.
20. Gärtner T, Lach P, Wrobel S (2003) On graph kernels: hardness results and efficient alternatives. Proc 16th Learning Theory Conf. 129–143.
21. Zhang Y, Lin H, Yang Z, Li Y (2011) Neighborhood hash graph kernel for protein-protein interaction extraction. Journal of biomedical informatics 44: 1086–1092.
22. Miwa M, Soetre R, Miyao Y, Tsujii J (2009) Protein–protein interaction extraction by leveraging multiple kernels and parsers. Journal of Medical Informatics 78: 39–46.
23. Yang Z, Tang N, Zhang X, Lin H, Li Y, et al. (2011) Multiple kernel learning in protein–protein interaction extraction from biomedical literature. Artificial Intelligence in Medicine 51: 163–17.
24. Lease M, Charniak E (2005) Parsing biomedical literature. Proc 2nd Int. Joint Conf. Natural Language Processing Korea: 58–69.
25. Pyysalo S, Airola A, Heimonen J, Björne J, Ginter F, et al. (2008) Comparative analysis of five protein-protein interaction corpora. BMC Bioinformatics 9: S6.
26. Shohei H, Hisashi K (2009) A linear-time graph kernel. Proc IEEE international conference on data mining Florida: 179–88.
27. Zhang M, Zhang J, Su J, Zhou G (2006) A composite kernel to extract relations between entities with both flat and structured features. Proc International Conference on Computational Linguistics and Annual Meeting of the ACL: 825–832.
28. Zhang M, Dong G, Aiti A (2008) Exploring syntactic structured features over parse trees for relation extraction using kernel methods. Information Processing and Management 44: 687–701.
29. Zhou G, Zhang M, Ji D, Zhu Q (2007) Tree kernel-based relation extraction with context-sensitive structured pares tree information. Proc EMNLP and CNLL Prague Czech Republic 728–736.
30. Hanley J, McNeil B (1982) The meaning and use of the area under a receiver operating characteristic (ROC) rurve. Radiology 143: 29–36.
31. Thomas P, Neves M, Solt I, Tikk D, Leser U (2011) Relation extraction for drug-drug interactions using ensemble learning. Proc DDIExtraction-2011 challenge task Spain: 11–18.
32. Chowdhury M, Abacha A, Lavelli A, Zweigenbaum P (2011) Two diferent machine learning techniques for drug-drug interaction extraction. Proc DDIExtraction-2011 challenge task Spain: 19–26.
33. Chowdhury M, Lavelli A (2011) Drug-drug interaction extraction using composite kernels. Proc DDIExtraction-2011 challenge task Spain: 27–33.
34. Björne J, Airola A, Pahikkala T, Salakoski T (2011) Drug-drug interaction extraction with rls and svm classiffers. Proc DDIExtraction-2011 challenge task Spain: 35–42.
35. Wishart D, Knox C, Guo A, Cheng D, Shrivastava S, et al. (2008) DrugBank: a knowledgebase for drugs, drug actions and drug targets. Nucleic Acids Research 36: D901–D906.
36. Garcia-Blasco S, Mola-Velasco S, Danger R, Rosso P (2011) Automatic drug-drug ineraction detection: a machine learning approach with maximal frequent sequence extraction. Proc DDIExtraction-2011 challenge task Spain: 51–58.
37. Baldi P, Brunak S, Chauvin Y, Andersen C A, Nielsen H (2000) Assessing the accuracy of prediction algorithms for classification: an overview. Bioinformatics 16: 412–424.
38. Carugo O (2007) Detailed estimation of bioinformatics prediction reliability through the Fragmented Prediction Performance Plots. BMC Bioinformatics 8:380.