A novel feature-based approach to extract drug-drug interactions from biomedical text

Bui, Q.C.; Sloot, P.M.A.; van Mulligen, E.M.; Kors, J.A.

Published in:
Bioinformatics

DOI:
10.1093/bioinformatics/btu557

Citation for published version (APA):
A novel feature-based approach to extract drug-drug interactions from biomedical text

Quoc-Chinh Bui\textsuperscript{1,*}, Peter M.A. Sloot\textsuperscript{2,3,4}, Erik M. van Mulligen\textsuperscript{1} and Jan A. Kors\textsuperscript{1}

\textsuperscript{1}Department of Medical Informatics, Erasmus University Medical Center Rotterdam, The Netherlands.
\textsuperscript{2}Informatics Institute, Faculty of Science, University of Amsterdam, The Netherlands.
\textsuperscript{3}Complexity Institute, Nanyang Technological University, Singapore.
\textsuperscript{4}ITMO University, St. Petersburg, Russian Federation.

ABSTRACT

Motivation: Knowledge of drug-drug interactions (DDIs) is crucial for healthcare professionals in order to avoid adverse effects when co-administering drugs to patients. Since most newly discovered DDIs are made available through scientific publications, automatic DDI extraction is highly relevant.

Results: We propose a novel feature-based approach to extract DDIs from text. Our approach consists of three steps. First, we apply text preprocessing to convert input sentences from a given dataset into structured representations. Second, we map each candidate DDI pair from that dataset into a suitable syntactic structure. Based on that, a novel set of features is used to generate feature vectors for these candidate DDI pairs. Third, the obtained feature vectors are used to train a support vector machine (SVM) classifier. When evaluated on two DDI extraction challenge test datasets from 2011 and 2013, our system achieves F-scores of 71.1\% and 83.5\%, respectively, outperforming any state-of-the-art DDI extraction system.

Availability: The source code is available for academic use at \url{http://www.biosemantics.org/uploads/DDI.zip}

Contact: q.bui@erasmusmc.nl

1 INTRODUCTION

Drug-drug interaction (DDI) is a situation when one drug increases or decreases the effect of another drug (Tari \textit{et al.}, 2010). Information about DDIs is crucial for drug administration in order to avoid adverse drug reactions or therapeutic failure (van Roon \textit{et al.}, 2009). For example, a recent study reports that DDIs are a significant cause of hospital admissions (Dechanont \textit{et al.}, 2014). While specialized databases are available for finding known DDIs, such as DrugBank (\url{http://www.drugbank.ca}) or Micromedex (\url{http://micromedex.com}), their coverage is limited and there are discrepancies in DDI listing between existing databases (Wong \textit{et al.}, 2008). As a consequence, most of newly discovered DDIs need to be extracted from scientific publications (Herrero-Zazo \textit{et al.}, 2013). Text mining techniques such as automatic relation extraction have been applied successfully in large-scale experiments to extract various types of relations (e.g., protein-protein interactions (PPIs), gene-disease) efficiently (Rebholz-Schuhmann \textit{et al.}, 2012; Hahn \textit{et al.}, 2012). Therefore, automatic DDI extraction methods can be particularly relevant to effectively extract DDIs and corresponding evidence from the scientific literature.

To develop and evaluate automatic DDI extraction methods, a DDI corpus has been created by Herrero-Zazo \textit{et al.} (2013). This corpus was manually annotated with 18,502 pharmacological substances, mainly consisting of generic and brand names, and 5,028 DDIs. With the availability of this corpus and the introduction of two DDI extraction challenges in 2011 and 2013 (Segura-Bedmar \textit{et al.}, 2011a, 2013), several approaches have been proposed to extract DDIs from biomedical text. In both challenges, systems built on machine learning (ML) approaches were dominant and achieved the best results (Segura-Bedmar \textit{et al.}, 2011a, 2013). In these systems, the DDI extraction tasks are modeled as classification problems where each candidate DDI pair is classified as an interacting pair or not. To build the classification models, data from annotated DDI corpora are often transformed into more structural representations using various natural language processing (NLP) tools. Among these ML-based systems, SVM methods are the most popular (Segura-Bedmar \textit{et al.}, 2013). In general, ML-based DDI extraction systems can be categorized into two groups, namely feature- and kernel-based methods.

In feature-based systems, each data instance is represented as a feature vector in an n-dimensional space. The main focus in these systems is to define features that potentially best represent the data characteristics. For DDI extraction tasks, various feature types have been employed ranging from lexical to syntactic, and semantic information. For example, Segura-Bedmar \textit{et al.} (2011b) developed a system using bag-of-words and local context features. To improve the performance of feature-based systems, some authors combine multiple types of features with the hope that these features can complement each other. He \textit{et al.} (2013) introduced a system that uses lexical, semantic and domain knowledge features. Chowdhury and Lavelli (2013) proposed a system that combines heterogeneous features. Their system comprises lexical, syntactic,
semantic, and negation features derived from sentences and their corresponding parse trees.

In kernel-based systems, the structural representations of data instances, e.g., syntactic parse trees or dependency graphs, are exploited. Various kernels have been proposed to quantify the similarities between two instances by computing the similarities of their representations. These kernels differ from each other based on how syntactic representations are used and how similarity functions are calculated (Tikk et al., 2013). For the DDI extraction challenges, the use of kernels varies between the participating systems. Among them, the most commonly used kernels are All-paths Graph kernel (Airola et al., 2008), Shallow Linguistic kernel (Giuliano et al., 2006), and Path-enclose Tree kernel (Moschitti, 2004). Since the proposed kernels exploit different types of structural representations and similarity functions, they all have pros and cons. To compensate for the weakness of each individual kernel, kernel combination is often used. For example, Chowdhury and Lavelli (2013b) proposed a hybrid kernel which combines three different kernels. Their system achieved the best results in the DDI extraction 2013 challenge (Task 2). Furthermore, the combination can take place at the output level (ensemble approach) where the output of multiple systems is combined using a voting scheme. Thomas et al. (2011) developed a system which combines the output of two kernel-based systems and a case-based reasoning system using a majority voting scheme. This system yielded the best result in the DDI extraction 2011 challenge.

Although systems employing feature-based kernels alone did not yield the best performance in the DDI extraction challenges, feature-based kernels still play an important role in relation extraction tasks. In fact, the winning teams of the DDI extraction 2011 and 2013 challenges both incorporate feature-based kernels proposed by Giuliano et al. (2006) as part of their systems. Furthermore, Miwa et al. (2009) have shown that their feature-based PPI extraction system achieved state-of-the-art results on five PPI corpora. A recent study by Tikk et al. (2013) on the performance of various types of kernels for PPI extraction tasks also suggests that in order to improve the performance of the current PPI extraction systems, novel feature sets should be explored over novel kernel functions. This suggestion may also apply to the DDI extraction tasks since most current approaches to extract DDI pairs have also previously been used to extract PPI pairs.

In this paper we propose a novel feature-based approach to extract DDIs from biomedical text. Our approach differs from existing approaches in two ways. First, we partition candidate DDI pairs into five groups based on their syntactic structures. Second, we apply a set of novel features which is optimized for each group based on the syntactic properties. Our results show that the proposed system achieves the best results in terms of F-scores and performance efficiency when compared with the state-of-the-art DDI extraction systems.

2 METHODS

Our method consists of three steps. First, we apply text preprocessing to convert input sentences into structured representations. Second, a feature vector for each candidate DDI pair is extracted from the corresponding structured representation using predefined feature sets. In the last step, the obtained feature vectors are used to train an SVM classifier to generate a predictive model, which is used to classify candidate DDI pairs of the test dataset.

2.1 Text preprocessing

The text preprocessing step consists of filtering out irrelevant sentences, entity blinding, word tokenizing, part-of-speech (POS) tagging, and parsing sentences with a shallow parser. We manually created a list of 292 trigger words by combining a list of trigger words previously used to extract PPIs (Bui et al., 2011) and some trigger words specific to DDI taken from the training dataset. Sentences that contain one drug or have no trigger word are filtered out. Next, to improve generalization of the input sentences, all drug names are blinded by assigning names as DRUGi where i is the drug index. Each sentence is then tokenized and POS tagged with the LingPipe NLP toolkit (http://alias-i.com/lingpipe). Finally, the tokens and their tags are used as input for the OpenNLP shallow parser (https://opennlp.apache.org/) to produce chunks.

2.2 Structured representation

We adapt the structured representation proposed by Bui and Sloot (2012) to express candidate DDI pairs. This structured representation, which consists of three syntactic layers (chunk, phrase, and clause), is generated based on the chunks outputted from the shallow parser. Since there are many cases where DDI pairs span into more than one single clause, we represent these cases using multiple single clauses. We modify the structured representation as follows:

Phrase: consists of a list of chunks (i.e. the output of the shallow parser).

Figure 1b shows examples of phrases (dashed boxes), which consist of noun chunks (NCs; plain boxes) connected by preposition chunks (PCs; shadowed boxes).

Clause: consists of a verb chunk and two phrases that are located in the left and in the right of the verb chunk. Complex sentences are represented by...
multiple clauses. For example, Figure 1a shows a clause that has a verb chunk connected with the left phrase (subject) and the right phrase (object). Figure 1c shows a complex sentence that consists of three clauses. Furthermore, to reduce the number of clauses generated for each input sentence, only verb chunks that belong to the main clauses are used to construct the structured representation.

With the proposed structured representation, we can express relationships of almost all drug pairs. Figure 1a and 1b show examples of drug pairs that interact (positive DDI) and that do not interact (negative DDI) expressed by the structured representations.

### 2.3 Features

In this section we describe a set of novel features that are specifically designed to exploit the strength of the structured representations. To generate features for each candidate DDI pair, we find the smallest syntactic container (e.g., a phrase, a clause, or clauses) from the structured representation containing that pair. For example, the smallest syntactic container of the DRUG1-DRUG2 pair in Figure 1b is a phrase whereas the smallest syntactic container of the DRUG2-DRUG3 pair in Figure 1c encloses two clauses. Given a candidate DDI pair and its syntactic container, we check whether the syntactic container contains any trigger words. If the syntactic container functions as a subject, we also check its right verb chunk for trigger words since there are cases in which trigger words do not belong to the subjects but to their right verb chunks. If no trigger word is detected then the candidate DDI pair is skipped, otherwise the following features are generated depending on its container type (e.g., subject, clause):

**Lexical features:** are used to capture relations between each drug of the candidate DDI pair and its surrounding tokens. These relations might reveal the syntactic role of the drug within the phrase containing it, such as whether the drug is a part of the coordination or is an abbreviation of another drug. Lexical features of each drug are three tokens on the left and three tokens on the right of that drug. Left and right tokens are distinguished by adding _L and _R suffixes, respectively. In addition, if a token is a drug (e.g. DRUG1 or DRUG2) then that token is replaced by ‘arg’. For example, lexical features of the DRUG2 in Figure 1b are: of_L, arg_L, with_L. Since DRUG2 is the last token of that phrase, there is no feature extracted from the right side.

**Phrase features:** are applicable for a candidate DDI pair of which the syntactic container is a phrase. These features are designed to capture relations of the candidate DDI pair and trigger words that belong to the phrase containing that pair. For each trigger word, we determine its relative position within the phrase by checking the following cases:

- Trigger [prep]* arg1 [prep]* arg2 (case 1)
- Arg1 [prep]* trigger [prep]* arg2 (case 2)
- Arg1 [prep]* arg2 [prep]* trigger (case 3)

Here prep are prepositions connecting chunks that contain the trigger and the DDI pair. Arg1 and arg2 are drugs of the (ordered) candidate DDI pair. The ‘*’ indicates that zero or more prepositions are required. Based on the obtained case, corresponding features are generated to represent the position between the trigger and the candidate DDI pair (i.e. left, middle, right), and to indicate which prepositions are used to connect the trigger and the target pair as well as the chunks between the drugs of the target pair. For example, features generated for the DRUG1-DRUG2 pair in Figure 1b are: use_of_arg1, arg1_with_arg2_case1. Furthermore, if there is a negative modifier (e.g., no, not) which belongs to the same chunk that contains a trigger, we insert the modifier as the prefix for that trigger.

Since it is non-trivial to automatically determine which trigger actually has a relation with (i.e. governs) the candidate DDI pair, all detected triggers are used to generate phrase features.

**Verb features:** are bag-of-words (unigrams and bigrams) generated from the verb chunk of the clause to which the candidate DDI pair belongs. The verb features indicate how the drug in the left phrase (subject) and the drug in the right phrase (object) are related.

**Syntactic features:** are designed to capture the surrounding syntactic structure of each drug of the candidate DDI pair within the phrase to which it belongs. To do this, we assign indices for all preceding noun and preposition chunks which connect to the noun chunk containing that drug. Furthermore, we also check whether there is any drug succeeding that drug and which prepositions are used to connect them. For example, the syntactic features generated for DRUG1 in Figure 1b are: NC1, PC2, has_more_args, and with_arg. Together with verb features, syntactic features particularly help to distinguish between DDI pairs that have a drug governed by its preceding noun and DDI pairs that have drugs spanning into two phrases (i.e. subject and object) of a clause. For example, consider the positive DRUG1-DRUG2 pair in Figure 1a and the negative DRUG2-DRUG3 pair in Figure 1b. Although both pairs have the same sequence of tokens, if the syntactic structure is used then DRUG1 in Figure 1a and DRUG2 in Figure 1b have completely different syntactic features.

**Auxiliary features:** consist of three features that capture information related to the drugs of the target pair. In particular, the first feature keeps track if drug names of the pair are real names vs. pronouns (e.g., these drugs, this drug). The second feature denotes whether the drugs of the have the same name, and the third feature indicates if the target drugs are in the same chunk.

### 2.4 Partitioning DDI pairs

In a previous study, Bui et al. (2011) showed that partitioning candidate PPI pairs based on syntactic properties and selecting partition-specific feature improved the performance of their PPI extraction system. Following this strategy, we categorize candidate DDI pairs into different groups based on their syntactic containers. To reduce the number of syntactic groups being generated, we only consider candidate DDI pairs that span over at most two clauses. For example, the DRUG1-DRUG3 pair in Figure 1c is ignored since it spans over three clauses. This partitioning process results in five syntactic groups, namely subject, object, clause, clause_2, and NP. Here clause_2 denotes a syntactic structure that spans over two clauses and NP denotes an input sentence that only contains a phrase.

Due to space limitations, we refer to the Supplementary source code for more details on text preprocessing and feature generation.

### 2.5 Machine learning

Recent relation extraction competitions have shown that the use of SVMs in relation extraction systems is dominant and systems which employ SVMs achieved the best performance (Segura-Bedmar et al., 2011a, 2013; Nédellec et al., 2013). In this study, we use the LIBSVM classifier with a default RBF kernel (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) for classification of DDI pairs. All individual features extracted for each DDI pair are normalized and combined into a single feature vector as proposed by Miwa et al. (2009). To find the best parameter C and gamma for each model, we use the CVParameterSelection function from the WEKA toolbox (http://www.cs.waikato.ac.nz/ml/weka/).

### 3 RESULTS AND DISCUSSION

#### 3.1 Datasets

We use the DDI extraction 2011 and 2013 datasets (hereafter referred to as DDI-2011 and DDI-2013) provided by the DDI extraction 2011 and 2013 challenges to evaluate our extraction...
method. Each dataset consists of two parts, a training dataset and a test dataset. There are differences between the two challenge datasets. The DDI-2011 datasets contain documents selected from the DrugBank database whereas the DDI-2013 datasets consist of documents selected from the DrugBank database and Medline abstracts. Furthermore, in the DDI-2011 dataset, each drug pair was annotated either as a true interaction (positive instance) or no interaction (negative instance) while the DDI-2013 datasets have more fine-grained annotations with different interaction types. Statistics of the datasets are shown in Table 1.

Table 1. Statistics of the DDI-2011 and DDI-2013 training and test datasets. The DDI-2013 datasets are split into two subsets (DB-2013 and ML-2013) based on document types. Sen., Pos., and Neg. denote numbers of input sentences, positive instances and negative instances, respectively.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDI-2011</td>
<td>4267</td>
<td>2402</td>
</tr>
<tr>
<td>DB-2013</td>
<td>5675</td>
<td>3788</td>
</tr>
<tr>
<td>ML-2013</td>
<td>1031</td>
<td>232</td>
</tr>
</tbody>
</table>

3.2 Transformation of datasets

When applying the text preprocessing and partitioning steps for each dataset, we obtain a transformed dataset where irrelevant DDI pairs are filtered out and the original dataset is split into five groups. Table 2 and Table 3 show statistics of the transformed datasets for training and test datasets, respectively. The data in these tables indicate that the text preprocessing has effectively filtered out significant numbers of negative instances (true negatives) with a small cost of missing positive instances (false negatives). Overall, the numbers of filtered instances vary from 2.5% to 4.1% for false negatives and from 27.9% to 33.8% for true negatives on the DrugBank datasets. However, the numbers of false negatives on the Medline dataset are unexpectedly high, ranging from 8.6% to 17.9.0%. Furthermore, a small number of positive instances are ignored during the partition step due to their complex syntactic structures. These numbers are shown in Table 2 and Table 3 as ignored cases.

In addition, the data from Table 2 and Table 3 show that the numbers of instances vary significantly between groups of each dataset and across datasets. This indicates that the performance on each group might also differ accordingly.

3.3 Evaluation settings

We use the standard evaluation measures (Precision, Recall, and F-score) proposed by the DDI extraction challenge to evaluate the performance of our system (Segura-Bedmar et al., 2013). As our method mainly focuses on the detection of interaction pairs, we ignore the interaction types annotated in the DDI-2013 dataset. (Note that the detection of DDI pairs is an important step in the extraction pipeline of most of the systems that participated in the DDI extraction 2013 challenge, including the top two systems). In addition, since we partition each dataset into five groups, we need to train the classifier separately for each group. To find the optimal feature sets for these groups, we tried various combinations of the proposed features. The best feature sets for each group are shown in Table 4. These features were determined based on the DB-2013 training set but used for all evaluations.

We evaluate the performance of our system on each test dataset after training on the corresponding training dataset, except for the ML-2013 test dataset. For this test dataset, the system is trained on the combined DB-2013 and the ML-2013 training datasets as suggested by Thomas et al. (2013) and Chowdhury and Lavelli (2013b).

Table 2. Statistics of the transformed training datasets after applying preprocessing steps. Pos. and Neg. denote positive and negative instances.

<table>
<thead>
<tr>
<th>Group</th>
<th>DB-2013</th>
<th>ML-2013</th>
<th>DDI-2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>876</td>
<td>4301</td>
<td>29</td>
</tr>
<tr>
<td>Object</td>
<td>356</td>
<td>4797</td>
<td>13</td>
</tr>
<tr>
<td>Clause</td>
<td>1852</td>
<td>2238</td>
<td>121</td>
</tr>
<tr>
<td>Clause_2</td>
<td>341</td>
<td>871</td>
<td>12</td>
</tr>
<tr>
<td>NP</td>
<td>197</td>
<td>1039</td>
<td>27</td>
</tr>
<tr>
<td>Total</td>
<td>3622</td>
<td>13246</td>
<td>202</td>
</tr>
<tr>
<td>(known cases)</td>
<td>(96%)</td>
<td>(60%)</td>
<td>(87%)</td>
</tr>
<tr>
<td>Ignored cases</td>
<td>60</td>
<td>479</td>
<td>10</td>
</tr>
<tr>
<td>Filtered out/skipped</td>
<td>106</td>
<td>8492</td>
<td>20</td>
</tr>
<tr>
<td>(2.8%)</td>
<td>(38.2%)</td>
<td>(8.6%)</td>
<td>(37.7%)</td>
</tr>
</tbody>
</table>

Table 3. Statistics of the transformed test datasets after applying preprocessing steps. Pos. and Neg. denote positive and negative instances.

<table>
<thead>
<tr>
<th>Group</th>
<th>DB-2013</th>
<th>ML-2013</th>
<th>DDI-2011</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>156</td>
<td>782</td>
<td>21</td>
</tr>
<tr>
<td>Object</td>
<td>90</td>
<td>1174</td>
<td>16</td>
</tr>
<tr>
<td>Clause</td>
<td>504</td>
<td>429</td>
<td>36</td>
</tr>
<tr>
<td>Clause_2</td>
<td>37</td>
<td>229</td>
<td>2</td>
</tr>
<tr>
<td>NP</td>
<td>61</td>
<td>367</td>
<td>3</td>
</tr>
<tr>
<td>Total</td>
<td>848</td>
<td>2981</td>
<td>78</td>
</tr>
<tr>
<td>(known cases)</td>
<td>(96%)</td>
<td>(68%)</td>
<td>(82%)</td>
</tr>
<tr>
<td>Ignored cases</td>
<td>14</td>
<td>178</td>
<td>0</td>
</tr>
<tr>
<td>Filtered out/skipped</td>
<td>22</td>
<td>1222</td>
<td>17</td>
</tr>
<tr>
<td>(2.5%)</td>
<td>(27.9%)</td>
<td>(17.9%)</td>
<td>(41.6%)</td>
</tr>
</tbody>
</table>

Table 4. Optimized features for each syntactic group.

<table>
<thead>
<tr>
<th>Group</th>
<th>Lexical Phrase</th>
<th>Verb</th>
<th>Syntactic</th>
<th>Auxiliary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Object</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Clause</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Clause_2</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.4 Performance of DDI extraction

Table 5 shows the results of our system evaluated on the DDI-2011 and DDI-2013 test datasets. In order to understand its performance on different document types (i.e. DrugBank and Medline abstracts), we present the results of the DDI-2013 sub datasets separately. Furthermore, to calculate recall, all positive instances missed by the previous preprocessing steps are considered as false negatives. Besides reporting the overall performance of the whole dataset, we also present the performances of individual groups. Note that recall for each group is calculated using data from Table 3 (which do not take into account filtered and ignored instances) whereas the recall for the overall performance for each test dataset is calculated using data from Table 1.

Table 5. Evaluation results on the DDI-2011 and DDI-2013 test datasets. The DDI-2013 test datasets are split into two subsets (DB-2013 and ML-2013) based on document types. P and R denote precision and recall, respectively.

<table>
<thead>
<tr>
<th></th>
<th>DB-2013</th>
<th>ML-2013</th>
<th>DDI-2011</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P (%)</td>
<td>R (%)</td>
<td>P (%)</td>
</tr>
<tr>
<td>Subject</td>
<td>83.92</td>
<td>76.92</td>
<td>86.67</td>
</tr>
<tr>
<td>Object</td>
<td>84.72</td>
<td>67.78</td>
<td>54.55</td>
</tr>
<tr>
<td>Clause</td>
<td>86.08</td>
<td>94.44</td>
<td>71.79</td>
</tr>
<tr>
<td>Clause_2</td>
<td>91.67</td>
<td>59.46</td>
<td>100.00</td>
</tr>
<tr>
<td>NP</td>
<td>88.64</td>
<td>63.93</td>
<td>25.00</td>
</tr>
</tbody>
</table>

The results in Table 5 show that our system performs well on the DB-2013 and DDI-2011 test datasets with F-scores of 83.5% and 71.1%, respectively. However, its performance decreases on the Medline test dataset with an F-score of 59.2%, which is 24.3 points lower than the DB-2013 test dataset. This performance decrease stems from the low recall, which can partly be explained by the loss of positive instances during the preprocessing steps. In addition, for each dataset, the performance on each group also differs significantly. These performance differences might be due to three factors. First, the ratio of the positive and negative instances varies among all groups (see Table 2 and Table 3). This causes the performance degradation for groups that have smaller instances varies among all groups (see Table 2 and Table 3). Furthermore, our system also yields better results when compared with these systems on the ML-2013 test dataset. The results in Table 7 show that the BioSem achieves an F-score of 59.2%, which is higher than the other systems 6.2 to 17.1 points. It is worth noting that the systems that participated in the challenges had to be developed under strict time constraints, which may have affected their performance. Nevertheless, the authors of the top performing systems have participated in the DDI-2011 Extraction challenge and thus were familiar with the task and could fine-tune their systems using the DDI-2011 test dataset.

To provide a fair performance comparison, we present the evaluation results of the best known systems that run on the DDI-2011 post-challenge test dataset in Table 8. We also provide the results of the best system of the DDI-2011 Extraction challenge for reference. The data show that post-challenge systems achieve higher performance in terms of F-scores as compared to the best system of the DDI-2011 Extraction challenge. These performance im-
provements might stem from the fact that these systems have a better design and/or could be fine-tuned on the available test dataset. Compared to these post-challenge systems, our system yields better results with F-score improvements ranging from 1.9 to 2.2 points. It is worth noting that the system proposed by Chowdhury and Lavelli (2013b) is the same system that achieved the best results in DDI-2013 challenge.

3.5 Performance analysis

In this section we address some issues related to the performance of the proposed system as well as discuss its complexity with respect to the state-of-the-art systems.

3.5.1 Performance variation on different datasets

In the previous section we have mentioned that the ratio of positive and negative instances might directly contribute to the differences in performance between syntactic groups (e.g., subject, object, etc.) of each dataset. This phenomenon can also be observed in the same groups across different datasets. For example, on the DB-2013 dataset, the ratios of positive/negative instances of the clause group are 0.83 and 1.20 for training and test datasets, whereas on the DDI-2011 dataset these values are 0.39 and 0.38, respectively (see Table 2 and 3). These differences might explain why precision and recall of the clause group differs between these two datasets: 86.1% vs. 65.8% for precision and 94.4% vs. 84.8% for recall. Furthermore, this might also explain the high precision of the subject group on the ML-2013 test dataset as the positive/negative ratios between training and test datasets are 0.11 and 1.05, respectively.

Another issue that might affect the system performance is the size of the datasets. This is clearly visible for the ML-2013 dataset, which is significantly smaller (14 times) than the DB-2013 dataset. Moreover, learning a model from a small training set is one of the challenges of an ML-based approach. This problem is even harder in our case since we further split the training set into five sub datasets. For example, when we used the ML-2013 dataset alone for training, our system achieved an F-score of 35.4% on the ML-2013 test dataset (data not shown). However, when trained on the combined DB-2013 and ML-2013 training datasets and evaluated on the ML-2013 test set, the F-score increases to 59.2%. This indicates that even though there are differences in structure between the document types (Cohen et al., 2010) of two datasets, increasing the size of the ML-2013 training set by adding training instances from the DB-2013 set, to some extent, helps improving the performance of our system on this test dataset.

3.5.2 Contribution of the proposed feature sets

When applying an ML-based approach for relation extraction tasks, each candidate pair is classified independently as being a true interaction pair or not. The benefit of this approach is that it can easily be used with any (binary) classifier. However, when each candidate DDI pair is considered independently, it is taken out of context. In the other words, the dependencies between the drugs of the candidate DDI pair and their neighboring drugs might be missed, which might lead to a wrong classification. For example, consider a positive DRUG1-DRUG2 pair and two negative DRUG1-DRUG3 and DRUG2-DRUG3 pairs in the sentence “Concurrent use of DRUG1 with DRUG2 may increase the effect of DRUG3” as shown in Figure 1b. For the DRUG2-DRUG3 pair, if only lexical features are used then one may miss the information that DRUG2 has already participated in a relation with DRUG1. For the DRUG1-DRUG3 pair, even if a dependency tree is used, one might still miss the information that DRUG1 has a relation with DRUG2. To address this problem, previous systems usually combine various types of features so that they can complement each other. In our system, we explicitly tackle this problem by introducing three novel feature sets, namely verb, phrase, and syntactic features.

Table 9 shows the contributions of the phrase, syntactic, and verb features on the performance of our system when evaluated on the DB-2013 test dataset. The data show that when the verb features are removed, the performance in terms of F-score degrades 3.56% compared to that of the whole feature set. While removing the phrase or syntactic feature alone decreases the performance slightly, removing both phrase and syntactic features results in the performance decreases 1.53%. This means that one of these features may only be suitable for certain groups. This phenomenon is clearly visible when we apply the optimized feature sets from Table 4 to the test dataset, resulting in an increase of 0.95% on the F-score compared to that of the whole feature sets.

<table>
<thead>
<tr>
<th>Features</th>
<th>P (%)</th>
<th>R (%)</th>
<th>F (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimized feature sets</td>
<td>85.88</td>
<td>81.22</td>
<td>83.48</td>
</tr>
<tr>
<td>(1) - Phrase - Syntactic</td>
<td>81.70</td>
<td>80.32</td>
<td>81.00</td>
</tr>
<tr>
<td>(1) - Phrase</td>
<td>84.63</td>
<td>79.75</td>
<td>82.12</td>
</tr>
<tr>
<td>(1) - Verb</td>
<td>81.90</td>
<td>76.24</td>
<td>78.97</td>
</tr>
<tr>
<td>(1) - Verb + Phrase + Syntactic</td>
<td>85.64</td>
<td>79.63</td>
<td>82.53</td>
</tr>
</tbody>
</table>

In addition, by mapping each candidate DDI pair into a syntactic container before generating features, we can enhance the lexical features by not generating unnecessary tokens surrounding each drug of the candidate DDI pair. For example, the number of lexical features generated for DRUG2 in Figure 1b is three features instead of six features for systems that use a flat structure.

3.5.3 Computational performance and complexity

To increase the performance of DDI extraction systems, most of top-performing systems use either ensemble approaches (Thomas et al., 2013, 2011) or kernel combination approaches (Chowdhury and Lavelli, 2013b; He et al., 2013). While they manage to increase the performances, the computational resources and the complexity of their systems also increase. Furthermore, some systems also incorporate domain knowledge (Thomas et al., 2013; He et al., 2013) to enhance the performance, but this hinders the adaptation of these systems to new relation extraction tasks.

By contrast, our proposed feature-based system uses a small set of features to generate feature vectors from a simple syntactic representation. It employs a shallow parser for analyzing input
sentences and only requires a single kernel to build predictive models. Therefore, it is simpler and requires less computational time compared to the other ML-based systems. For example, our system requires 51 seconds to process the DB-2013 dataset (22 seconds for the text preprocessing step and 29 seconds for training and classifying instances). This experiment was performed on a laptop with an Intel Core i7-2640M, 2.8 GHz processor.

3.5.4 Error analysis
To identify the main sources of error of our system, we analyze all errors (118 false positives (FPs), 130 false negatives (FNs)) produced by our system when evaluated on the DB-2013 test dataset. Overall, these errors (both FPs and FNs) can be categorized into four groups. The first group of errors (22 FPs, 39 FNs) is caused by parser errors or incorrect construction of structured representations. These errors lead to the wrong categorization of candidate DDI pairs. The second error group (34 FPs) is caused by a nondeterministic context, where the syntactic containers of the candidate DDI pairs alone are not enough to determine the outcome. The third error group (42 FPs, 91 FNs) is caused by unusual syntactic structures of the input sentences, anaphora problems, and the long distance between two drugs (measured by the number of chunks) of the candidate DDI pairs. The fourth error group (20 FPs) consists of cases where candidate DDI pairs syntactically seem to be true DDI pairs.

While most of the errors are non-trivial, the errors caused by input sentences with special syntactic structures can be tackled if rules are defined to convert these input sentences into a form that can be handled by the structured representation. For the other errors, substantial changes in the system are needed to further improve the current performance.

4 CONCLUSIONS
In this study we have proposed a novel feature-based approach to extract DDIs from text. The key factors of our approach are the combination of the novel feature sets and the partition of the datasets. By partitioning the original dataset into subsets based on their syntactic properties, we obtain more consistent sub datasets. By partitioning the original dataset into subsets based on their syntactic properties, we obtain more consistent sub datasets. Overall, these errors (both FPs and FNs) can be categorized into four groups. The first group of errors (22 FPs, 39 FNs) is caused by unusual syntactic structures of the input sentences, anaphora problems, and the long distance between two drugs (measured by the number of chunks) of the candidate DDI pairs. The fourth error group (20 FPs) consists of cases where candidate DDI pairs syntactically seem to be true DDI pairs.

While most of the errors are non-trivial, the errors caused by input sentences with special syntactic structures can be tackled if rules are defined to convert these input sentences into a form that can be handled by the structured representation. For the other errors, substantial changes in the system are needed to further improve the current performance.

REFERENCES

ACKNOWLEDGEMENTS
PMAS is partially supported by Russian Scientific Foundation, proposal #14-21-0037.

Conflict of Interest: none declared.