Abstract The exchange of goods and services between individuals is often formalised by a contract in which the parties establish norms to define what is expected of each one. Norms use deontic statements of obligation, prohibition, and permission, which may be in conflict. The task of manually detecting norm conflicts can be time-consuming and error-prone since contracts can be vast and complex. To automate such tasks, we develop an approach to identify potential conflicts between norms. We show the effectiveness of our approach and its individual components empirically using two publicly available corpora, and contribute with a new annotated test corpus for norm conflict identification.

Keywords norms, natural language processing, normative conflicts, deontic logic.

1 Introduction

In social groups, interactions between members often follow some kind of regulation to minimize conflicting behaviour. This regulation is based on expected behaviours for each group member, which ensures that members follow a socially accepted behaviour. Societies use norms to formalise expected behaviour and act as group regulators that enforce a defined conduct for certain situations for each agent in the society. Their definitions usually respect a social consensus, which is formed by group representatives. In general, norms follow deontic concepts that define three types of modalities: permissions, obligations and prohibitions (von Wright, 1951). A permission states a behaviour that is allowed to be executed. An obligation states a behaviour that must be executed. Finally, a prohibition states a behaviour that must not be executed. These concepts are present in most social relationships involving agreements between members of a society. Agreements are usually formed by two or more parties that agree in the exchange of goods or services. Such agreements are often formalised by contracts, which describe the parties and a set of clauses. Within the clauses, contracts describe norms to be agreed upon.

A contract defines the parties, their relations, and a set of clauses containing norms that explain what each party must comply with Gao et al (2012). Since the use of contracts to regulate the exchange of goods and services through the Internet has increased, contracts have been formalised for use in the online context. Online contracts are increasingly used for different kinds of agreements involving the trading of products and services. This increase demands much more effort in the process of contract creation and analysis, since contracts tend to be long and complex documents.

Contractual norms include some kind of logical description of behaviours, defining what is obliged, permitted or forbidden. For example, in a restaurant, customers are prohibited from smoking and obliged to pay for the food they eat. However, conflicts between norms may occur depending on the objects they refer to and the modalities used in them. A conflict arises when norms that represent different deontic concepts are applied to the same target. One example is when a norm obliges a certain company to pay for a product and another one prohibits it from doing so. Conflicts of this type may invalidate norms involved, confusing the party according to what is expected of it in the situation described. Thus, a specific requirement in a norm conflict is to ensure that norms described in contracts do not introduce inconsistencies into them. Such effort can prevent the case when long and complex contracts have conflicts introduced unwittingly. Automating the identification of such conflicts avoids contract inconsistency and assists the human analysis, facilitating the task of writing and correcting contracts.

Identifying potential conflicts between norms within contracts is usually done by humans who manually check each norm and compare their meaning. This is a laborious and time-consuming process, which in turn makes contract verification slow, difficult, and error-prone. There is a great potential to improve the speed and quality of verification by automating such process. Recent efforts (Vasconcelos et al, 2009; Figueiredo and da Silva, 2013; Pace and Schapachnik, 2012; Gorín et al, 2011) have tried to identify normative conflicts, however, there are few approaches dealing with norms in the natural language context.

To make possible a conflict analysis over norms written in natural language, we extract the elements that compose a norm. For this task, we leverage existing work from Gao and Singh (2014, 2013), which propose approaches to extract normative relationships and norm properties in business contracts. We extend such approaches to accomplish a model that identify potential conflicts between norms using norm properties. We use concepts from deontic logic as basis to identify normative conflicts. The relations between deontic meanings help us to check if two norms are conflicting. While we acknowledge that the type of deontic logic we use is somewhat naive and vulnerable to known paradoxes, as we show, our approach lends itself very well to practical implementation, and constitutes a powerful first step towards automated contract conflict identification.

We divide our approach into two phases. First, we extract information from the contract structure using natural language processing techniques, which allow us to identify logic components of a formal representation of norms. In the second phase, we use the extracted norm representations to compare the norms according to the parties to which they are applied. In this phase, we focus on the comparison of norm elements, using concepts of deontic logic and language similarity to identify corresponding information in norm pairs that may produce a conflict. To evaluate our approach, we created a corpus with normative conflicts manually inserted. Thus, using the set of contracts with conflicts we test our identifier that obtains an accuracy of 78% overall inserted conflicts.

In this work, we have two main contributions. The first one is related to the development of a first step to the automation of contract analysis. Since contract analysis is poorly developed (Curtotti and Mccreath, 2010; Gao and Singh, 2014, 2013), this work improves the area by presenting a semi-automatic approach that provides a basis to more complex approaches. Our second contribution is a tool to assist in preventing conflicts from being inserted into contracts.

This work is divided into seven sections. In Section 2, we cover the main concepts used in this work, such as norms, contracts, deontic logic, and norm conflicts. In Section 3, we describe the first phase of our approach, which is the creation of a norm representation. To do so, we apply information extraction techniques to identify norms and their elements, such as party name, modal verb, and norm action. In Section 4, we describe the second phase of the work, in which we use the extracted information about norms to compare them and detect potential conflicts. Each potential conflict is then classified in one of three types, namely *permission* $\times$ *prohibition*, *permission* $\times$ *obligation*, and *obligation*$\times$*prohibition*. In Section 5, we describe previous work that deal with norms and norm conflict and we compare our work to theirs. In Section 6, we discuss about the weaknesses of our approach and introduce some solution to them. Finally, in Section 7, we draw some conclusions about the work and we state some future work.

2 Background

2.1 Norms and Deontic Logic

According to Axelrod (1986), "a norm exists in a given social setting to the extent that individuals usually act in a certain way and are often punished when perceived not to be acting in this way". Norms arise when a coordinated behaviour serves to regulate conflict under a large number of individuals, and provide a powerful mechanism for regulating conflict in groups, governing much of our political and social lives. In contracts, clauses define a set of norms. In such case, these norms indicate what is expected from each party according to the contract definitions. Most norm representations are based on deontic logic, using concepts such as permissions, obligations and prohibitions. These concepts describe the type of restriction intended of a norm.

Deontic Logic has its origins in philosophical logic, applied modal logic, and ethical and legal theory. The aim of deontic logic is to describe ideal worlds, allowing the representation of deviations from the ideal (i.e. violations). Thus, deontic logic and the theory of normative positions have strong relevance to legal knowledge representation, and consequently it is applied to the analysis and representation of normative systems (Jones and Sergot, 1992). Norms often present deontic concepts to describe permissions, obligations, or prohibitions. A prohibition norm indicates an action that must not be done, and, if such action is carried out, a violation occurs. Conversely, a permission norm indicates an action that can either be performed or not. In most deontic systems, a prohibition is considered to be equivalent to the negation of a permission, thus, an action that is not permitted comprises a prohibition. Although these two modalities are sufficient to represent most norms, obligations are also commonly employed in norm representation. An obligation represents an action that must be done, and it is equivalent either to the negation of a permission not to act or a prohibition not to act. Formally, we represent norms using the modal operators of P for permission, F for prohibition, and O for obligation. Additionally, we consider p and q as the parties in which a model operator is applied to. We represent actions with capital letters, such as A, B, and C. The negation of an action (e.g. $!A$) means not performing such action. For example, given an action A for "buy product W", the expression $!A$ means "not buy product W". We can represent, based on von Wright (1951) definitions, the equivalences between deontic concepts, as follows:

- $P(p, A) \equiv P(p, A) \vee P(p, !A)$
- $F(p, A) \equiv O(p, !A) \vee !P(p, A)$
- $O(p, A) \equiv F(p, !A) \vee !P(p, !A)$

Example 1 illustrates the equivalence between obligation and prohibition in two sentences. Equivalences allow us to compare and identify conflicts between norms. Conflicting norms often have different modal operators, thus, knowing these equivalences make it easy to identify differences.

Example 1 (Obligation and prohibition)

- Company X must follow rule Y.
- Company X must not disregard rule Y.

To extract norms and their properties, one must identify the contract structure and its components. Such components are responsible for dividing the contract into subjects, such as declaration of the parties and description of the clauses. Among such properties, the actions expressed in norms (that we will call norm actions in this paper) define a goal in which the party must satisfy for each case in a contract. These goals have two aspects as they can be either declarative or procedural. A declarative goal describes a sought state, i.e., a state-of-affairs according the described in the norm. Whereas a procedural goal describes a set of actions to achieve a certain goal (Winikoff et al, 2002). Example 2 illustrates both cases.

Example 2 (Declarative and Procedural Norm Actions)

- **Declarative**: Company A must keep the warehouse clean.
- **Procedural**: Company A shall clean the warehouse.

2.2 Contracts and their components

A contract is an agreement that two or more parties enter voluntarily, when it is useful to formalise that a certain duty comes into existence by a promise made by at least one of the parties (Rousseau and McLean Parks, 1993). The creation of a contract formalises what each party expects from the other, creating a warranty that the duties will be fulfilled. Therefore, it creates legally enforceable obligations between the parties. These enforceable obligations are defined by a series of norms, which are responsible for defining any expected behaviour from the parties. Contracts are used in many situations for different purposes. In any legal agreement, a contract handles the formalisation of agreement conditions.

With the use of the Internet, electronic contracts arise as a new way to represent formal agreements and are increasingly explored for commercial services. An electronic contract is very similar to a traditional paper-based commercial contract, following the same rules and structure. Most types of contract can be represented electronically, leading to the need of managing such contracts, dealing with representation and evaluation of agreements. In this work, we deal with contracts written in natural language, in which the task of analysing and evaluating norms is traditionally done by human readers. As more contracts are required to codify an increasing number of online services which span over multiple countries and different legal systems, the tasks of writing and verifying contracts such as end-user-licenses for these services by humans become more laborious, taking substantial time (Gao et al, 2012).

Analysing contracts from the Australian contract corpus (Curtotti and McCreath, 2011) and the corpus used by Gao et al (2012), we notice that most of them are a semi-structured document often divided into an initial description of the parties (header) and a set of clauses describing the expected behaviours from the parties. The header contains information about the parties, it presents and assigns them abbreviations or equivalent names. The set of clauses in a contract determines what each party must comply with along the duration of the contract. Norm clauses are often directed to one or more parties, likewise they may describe an expected behaviour from the agreement itself.

2.3 Modal Verbs and Their Deontic Meanings

Norms use concepts from deontic logic as a basis to describe permissions, prohibitions, and obligations. Such elements are often represented by modal verbs. Modality and modal verbs are connected by their meanings since modality describes deontic concepts and modal verbs describe modalities in sentences. For example, in the sentence: "Company X **must** pay the product taxes."; **must** is the modal verb that describes the obligation modality.

English has a set of modal verbs including: MAY, CAN, MUST, OUGHT (TO), WILL, and SHALL (Palmer, 2001). Moreover, *might*, *could*, *would*, and *should* are usually considered as modal verbs as well. From this set, we can map the modal verbs into deontic meanings, i.e., permission, prohibition, and obligation. Steedman (1977) maps MUST and MAY to obligation and permission, respectively. The remaining of modal verbs are mapped according to English grammar rules (Palmer, 2001). Table 1 illustrates the mapping between modal verbs and deontic meanings based on the associations made by Steedman (1977) and complemented according to grammar rules interpreted by Palmer (2001).

Table 1 Mapping modal verbs into deontic meaning.

Modal Verb	**Deontic Meaning**
CAN, MAY	Permission
MUST, OUGHT, SHALL, WILL	Obligation
CANNOT, MAY NOT, MUST NOT, OUGHT NOT, SHALL NOT, WILL NOT	Prohibition

Using the mapping in Table 1, we can identify the deontic meaning of a norm by means of its modal verb. Examples from 3 to 5 (extracted from Sample Business Contracts[1]) present sentences and their extracted deontic meaning.

[1] http://goo.gl/lE3q6H and http://goo.gl/E8brjp

Example 3 Purchaser **shall** also be financially responsible for all taxes and freight in connection with the New Equipment. (**Obligation**)

Example 4 Each party hereto **shall not** disclose any confidential information received by it pursuant to this Agreement without the prior written consent of the other. (**Prohibition**)

Example 5 Roxio, upon request, **may** review such agreements at any time before or after execution. (**Permission**)

2.4 Norm Conflicts

In order to understand the concept of norm conflict, we review a series of norm conflict concepts extracted from different areas, such as international law and multi-agent systems. One concept of norm conflict is the one applied to international law. It states that a conflict between two norms arises when "a party to two treaties cannot comply with its obligations in both treaties simultaneously" (Jenks, 1953). The sentences in Example 6 illustrate this type of conflict.

Example 6 (Obligation conflict)

– If the equipment presents an error, Customer A must send it to address X.
– If the equipment presents an error, Customer A shall send it to address Y.

In this case, both norms indicate different actions (different addresses) for the same event, making compliance with both norms impossible, invalidating them. In this example, even if a very unlikely interpreter could argue that a customer shall send the equipment to one address and then to another, it would still constitute a potential conflict due to ambiguity in language.

Although this concept is the prevailing view in international law, Vranes refines this definition, characterising conflicts between permissions and obligations, and permissions and prohibitions (Vranes, 2006). A conflict between a permission and an obligation arises when the party has permission for a defined action and, simultaneously, the obligation for the same action. For example, see the sentences in Example 7.

Example 7 (Conflict between a permission and an obligation)

– Company A must pay the taxes.
– Company A may choose to pay the taxes.

In the example, the first norm states that the party must pay the taxes, whereas the second norm permits the party to choose whether to pay the taxes or not. In the example, one norm invalidates the other producing a conflict. A conflict between a permission and a prohibition arises when the norms, simultaneously, permit and forbid the party to perform a certain action, as Example 8 illustrates.

Example 8 (Conflict between a permission and a prohibition)

1. Company A must not buy the product X.
2. Company A may buy any product.

Kollingbaum et al (2007) consider the concept of norm conflict the same as the second conflict type of Vranes. They define that a norm conflict is an interference between permissions and prohibitions. These concepts are similar to the ones presented by Sadat-Akhavi (2003), which states that norm conflicts arise when one cannot comply with all requirements of two norms. It means that it is impossible to comply with both norms, since they are mutually exclusive and cannot coexist in a legal order. Thus, compliance with one norm entails non-compliance with the other.

Sadat-Akhavi (2003) describes four causes for a norm conflict to arise. The first cause is when the same act is subject to different types of norms. Thus, a conflict of norms arises "if two different types of norms regulate the same act, i.e., if the same act is both obligatory and prohibited, permitted and prohibited, or permitted and obligatory". Example 9 shows a norm conflict between an obligation and a prohibition.

Example 9 (Conflict between an obligation and a prohibition) :

– Norm 1: The receiving State shall exempt diplomatic agents from indirect taxes.
– Norm 2: The receiving State shall not exempt diplomatic agents from indirect taxes.

The second cause is when one norm requires an act, while another norm requires or permits a 'contrary' act. By 'contrary', we are saying that such actions cannot be performed at the same time but both can be avoided. Therefore, a conflict of norms is produced if "two contrary acts, or if one norm permits an act while the other norm requires a contrary act" (Sadat-Akhavi, 2003). Example 10 illustrates the conflict. Both norms indicate different places in which a prisoner of war must be treated. Norm 1 states that it must (or may) be done in the prisoner camps, whereas Norm 2 states that it must happen in civilian hospitals. The conflict arises in the moment that one tries to comply with one norm and, at the same time, is non-complying with the other.

Example 10 (Norm conflict between norm 1 and 2)

– Norm 1: Prisoners of war suffering from disease shall/may be treated in their camps.
– Norm 2: Prisoners of war suffering from disease shall be treated in civilian hospitals.

The third cause for a norm conflict is when one norm prohibits a 'necessary precondition' of another norm. Suppose two actions A and B, where B cannot be performed without A having been performed before. In this case, a norm conflict arises when one norm prohibits A and another norm allows B, as Example 11 shows. In the example, we consider action A as "enter area X" and action B as "render assistance to any person in danger in area X". To comply with Norm 1 one must disobey Norm 2.

Example 11 (Norm conflict between norm 1 and 2) :

- Norm 1: Ships flying the flag of State A shall/may render assistance to any person in danger in area X.
- Norm 2: Ships flying the flag of State A shall not enter area X.

The fourth cause for a norm conflict to arise is when one norm prohibits a 'necessary consequence' of another norm. Suppose that one cannot perform action B without producing A as result. A conflict arises when one norm obliges B and another norm prohibits A, as Example 12 shows. If we consider action B as "replace existing rails in area X" and A as the period of time that the line in area X will be hampered, one cannot comply with both norms 1 and 2 in Example 12.

Example 12 (Norm conflict between norm 1 and 2)

- Norm 1: State A shall replace existing rails with new ones in area X.
- Norm 2: State A shall not hamper the transport of goods on the existing line in area X.

Fenech et al (2009a) define four reasons for a conflict to arise:

1. An agent is obliged and forbidden to perform the same action, e.g., given an action A and an agent p: $O(p, A) \wedge F(p, A)$;
2. An agent is permitted and forbidden to perform the same action, e.g., $P(p, A) \wedge F(p, A)$;
3. An agent obliged to perform two contradictory actions, e.g., given the contradictory actions A and B and $O(p, A) \wedge O(p, B)$; and
4. An agent is permitted to perform an action and obliged to perform a contradictory action, e.g., given the contradictory actions A and B and a party p, $P(p, A) \wedge O(p, B)$.

Pace and Schapachnik (2012) describe conflicts between conflicting norms and mutually exclusive actions. Considering an agent p, P for permission and O for obligation, they state that a conflict arises when an agent has opposite permissions, for example, given an action A, the following configures a conflict:

- $P(p, A) \wedge !P(p, A)$

The same occurs when an agent has obligations instead of permissions:

- $O(p, A) \wedge !O(p, A)$

Here, a conflict may arise when an agent has an obligation to perform an action conflicting with either a permission or an obligation not to perform it. The following cases describe such conflicts:

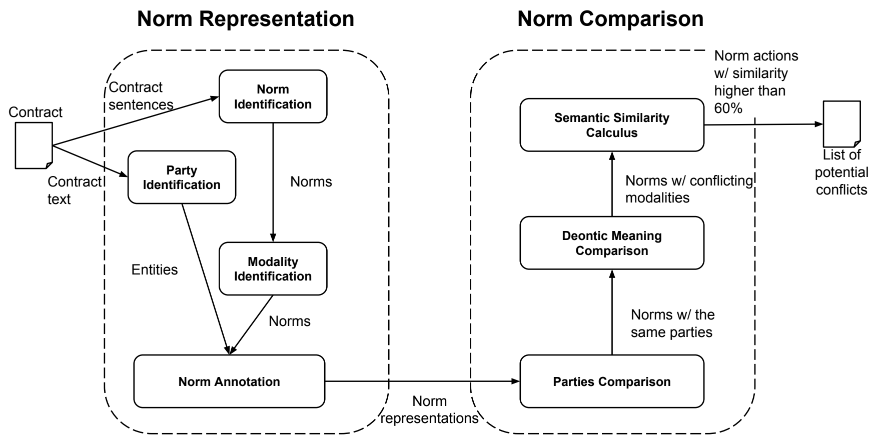

Fig. 1 Summary of the process of potential conflict identification.

- $O(p, A) \wedge P(p, !A)$;
- $O(p, A) \wedge O(p, !A)$; and
- $O(p, A) \wedge !P(p, A)$

The final type of conflict regards mutually exclusive actions, such that a conflict arises when an agent has an obligation to perform mutually exclusive actions, i.e., given two actions A and B, they conflict with the following case:

- $O(p, A) \wedge O(p, B)$

These conflicts involving mutually exclusive actions can help us to expand the notion of conflicts. However, the identification of mutually exclusive actions is a challenging task in natural language.

3 Detecting and Classifying Norms

Overall, our work is divided into two main phases, the creation of norm representations and the comparison between such representations. The former is responsible for extracting enough information to create a norm representation, this includes the norm itself, the parties, norm modalities, and norm actions. The latter uses this information to compare norms and identify potential conflicts. Figure 1 illustrates the architecture used in this work. In the figure, we define contract text as the raw text from a contract, i.e., without a preprocessing step. Whereas the list of contract sentences is the text divided into sentences, which means that they are the result of a preprocessing step.

In this section, we focus on the first phase, which is the creation of the norm representation. This phase consists of four processes, norm, party, modality, and norm action identification.

3.1 Norm Identification

The first step towards the norm conflict identification is the norm identification itself. Since we are dealing with natural language, norms are described as sentences within contracts. For this task, we consider contract sentences to be of two exclusive types: norm sentences and non-norm sentences. A norm sentence usually follows a well-defined 4-component structure: an indexing number or letter, one or more named parties, a modal verb, and a behaviour description. Example 13 illustrates a typical norm sentence.

Example 13 "**9.** The **Commission must** first attempt to resolve an industrial dispute by conciliation.".

Example 14 "The Code Participants are parties to a Dispute within the meaning of clause 8.2 of the Code.".

Conversely, common sentences have a different structure, as Example 14 shows. They seldom have an identifier and modal verbs and often have a different structure than norm sentences.

As modal verbs are the central element that differentiates a norm sentence from a common sentence, we defined a list of modal verbs to consider in a regular expression. For this work, we consider six modal verbs, namely: may, can, must, ought, will, and shall. We use a rule-based approach following the regular expression below to decide whether a sentence is a norm sentence or not:

– $Rule = .^{+}?\ modal\_verb\ .^{+}$
– $modal\_verbs = (may|can|must|ought|will|shall)$

3.2 Norm Representation

Since we can identify norm sentences from contracts, we propose a norm representation to compare such norms and identify potential conflicts. In this work, we use the norm representation to divide a norm into three components. These components are: party name, modal verb (deontic meaning), and norm action. Example 15 shows a norm representation applied to a real norm.

– **party_name**: For the conflicting cases we deal in this work, conflicting norms are applied to the same party. Therefore, our norm representation must identify what party must comply with the norm.
– **deontic_meaning**: Norm conflicts arise from differences between deontic meanings, such as obligation and prohibition, applied to the same context. Thus, deontic meaning is a relevant aspect to consider in norm representations. This component may contain one of three possible values, namely permission, prohibition, and obligation, depending on the modal verb in the norm.

– **norm_action**: Two norms with different deontic meanings present a conflict when they apply the same action to the same party.

Example 15 (Norm representation)
At the end of the term of this Agreement,
<party_name> 7UP/RC</party_name>
<modal_verb> may<modal_verb>
<norm_action> use the Equipment to produce Products as well as other energy drinks.</norm_action>

3.3 Party Identification

Given the norms from a contract, we need to identify each element of the norm based on the defined norm representation. The first step towards the construction of the norm representation is the entity extraction for the purpose of party identification. This task is divided into three subtasks: the identification of the contract parties, which are often defined as either person or company names; the identification of the aliases (abbreviations, nicknames), which are name simplifications often used to refer to a certain party within the contract; and the definition of a relation between the party names and their aliases, if they exist. Since the Australian contract corpus has many template contracts, which are contracts that have blank spaces representing parties (facilitating party identification), we decided to use another corpus for the rest of the tasks in our approach. Thus, we use the corpus provided by Gao et al (2012) in their work. This corpus has 2093 real contracts divided into several categories, such as manufacturing and purchasing.

To extract parties from contracts, we need to understand the contract structure. Since contracts are semi-structured documents, we often find the parties' definition in the beginning of a contract. Among the contracts analysed, we identify several ways in which parties are described. However, one particular pattern is commonly used. Consequently, we base our entity extraction on this pattern, as Example 16 illustrates (extracted from `http://goo.gl/vCHiUI`).

Example 16 This agreement is made by and **between** LUCENT TECHNOLOGIES INC., a Delaware corporation, acting through its Microelectronics Group, having an office at Two Oak Way, Berkeley Heights, New Jersey 07922 ("Lucent") **and** CD RADIO INC., a Delaware corporation, having its principal place of business at 2175 K Street NW, Washington, DC, 20037 ("CD Radio").

Entities are described using a simple pattern, which starts with the word "between" and ends with the end of the sentence. The sequence of words after "between" defines the first entity, it extends until the word "and", which separates the first entity from the second one. Party names containing the "and" in the name can be mistaken as two different parties. We consider only contracts with exactly two parties, i.e., contracts that describe the exchange of goods or services between either two persons or two companies. Using this pattern, we create a regular expression that extracts this block of text from contracts. From this block of text, we extract the entities. First, we use a list (Extracted from `http://goo.gl/YC5Utv`) with 2526 company names in order to find entities in the block of text. If the entity is not in the list, we use the NLTK (Bird, 2006) part-of-speech tagger to annotate the text, and then we extract the sequence of elements tagged as proper nouns. In Example 16, the names in capital letters represent the party names.

However, in many cases entities are represented by aliases within the contract. This information is often present in the same opening block of text that defines the parties in the contract. Within the description of each party, surrounded by parenthesis and, in some cases, double quotes, the name that will be used in the contract is defined. In this case, we apply a simple regular expression to extract the nicknames and abbreviations, when they exist. Using Example 16, which has the aliases identified by an underline, we extract:

– Entities: {Lucent Technologies Inc., CD Radio Inc.}
– Abbreviations: {Lucent, CD Radio}

To ensure that the alias indicates the party full name, we apply the regular expression in the stretch that defines the party. For example, from "between" to "and", which describes the first party, if the regular expression matches an element, we address it as the alias to the first party.

3.4 Modality Identification

The identification of modalities is an important step in the process of detecting potential conflicts. The conflicts we deal in this work consist mainly of different modalities applied to the same context, i.e., identifying opposite modalities in a pair of norms is half of the process on conflict identification. Based on the modal verb in a norm, we divide the sentence into subject and action since the party (subject) presented before the modal verb is intended to comply with the action specified after the modal verb. Additionally, identifying the deontic meaning of a modal verb in a norm allows us to compare norms and detect deontic conflicts between two norms.

To identify modal verbs, we use the list of modal verbs defined in Section 2.3. For each norm, we try to identify one of the elements of the list among the norm tokens. When identified, we check whether there is a negation after the verb, such as "may not". In this case, we compare what we found with dictionary entries indicating the deontic modality based on the modal verb with or without the negation. The dictionary follows the mappings from modal verbs to deontic meanings described in Table 1. Formally, let Θ be a norm sentence represented by the following sequence $(\theta_1, \theta_2, ..., \theta_n)$, where θ_i is the *ith* token in the norm. Additionally, let $(\lambda_1, \lambda_2, ..., \lambda_n)$ be a sequence Λ of modal verbs, where λ_i is the *ith* modal verb in the set. Let $\{\lambda_1 : \phi_1, \lambda_2 : \phi_2, ..., \lambda_n : \phi_n\} \in \Phi$ be a dictionary, where each entry has a λ

(modal verb) as key and returns a ϕ, which is a deontic meaning corresponding to the modal verb. To identify the modality of a norm, we need to find a θ_i in Θ that is equal to a λ_j in Λ. When we find a θ_i equal to a λ_j, we use λ_j as key to Φ, which returns a corresponding ϕ_l, thus, we address ϕ_l as the Θ deontic meaning. At the end of the process, we obtain an annotated norm according to its deontic meaning, and knowing the position of the modal verb allows us to identify where the action is, as Example 17 shows.

Example 17 (Annotated norm)
<Party1>Lucent<Party1> <Obligation>shall</Obligation> <norm_action>acknowledge all orders in writing</norm_action>.

4 Detecting and Classifying Potential Conflicts

Given the norm representation, in this section we develop the second phase of our approach. Here, we want to use the representation to compare different norms and detect potential conflicts between them. From these conflicts, we want to classify them according to the deontic meaning they present. In Section 6.4, we discuss our conflicting cases choice describing its limitations.

To detect a potential conflict, we compare two norms from the same contract and verify whether some conditions are satisfied. Such conditions, applied to a pair of norms, are:

– Both norms are applied to **the same party**;
– Both norms have **conflicting deontic meanings**; and
– Both norms refer to **the same act**, i.e., the same consequent in which the party must fulfil.

In order to classify a potential conflict, we perform a comparison between deontic meanings in norm pairs. Thus, based on the deontic conflict it arises, we indicate a conflict type. In this work, we deal with three conflict types, which we defined based on the first cause for a conflict to arise proposed by Sadat-Akhavi (2003). Such types are:

(1) $Permission \times Prohibition$
(2) $Permission \times Obligation$
(3) $Obligation \times Prohibition$

Following the relation between modal verb and deontic meaning, we are able to distinguish between the three types. For example, a norm pair can be classified as a potential conflict of type (2) if one norm has "can" as modal verb and the other has "must" as modal verb.

4.1 Computing Semantic Similarity

Two norms constitute a potential conflict when they have different deontic modalities but the same action, as Example 18 shows. However, as usual in natural language, norm actions can be written differently and yet have the same meaning, which leads to the need for a method that measures the semantic similarity between two norm actions. Example 19 illustrates two norms that have norm actions with the same meaning but written in different ways. Thus, given two norms applied to the same party, we extract the norm actions and calculate the semantic similarity between them. Based on this measure, we decide whether a norm pair must be considered a potential conflict.

Example 18 (Norms with different deontic modalities but the same action)

– Purchaser must <act>pay the product taxes</act>.
– Purchaser shall not <act>pay the product taxes</act>.

Example 19 (Norms with same meaning) :

– Purchaser must <act>pay the product taxes</act>.
– Purchaser shall not <act>pay the taxes applied to the product</act>.

To measure the semantic similarity between two norm actions, we use Algorithm 1. This algorithm takes two parameters σ_1 and σ_2, the word-vector corresponding to the norm actions of two sentences, and assumes that the number of words in σ_1 is smaller or equal to the number of words in σ_2. The comparison of these sentences consists of iterating over the indexes of each word in both word-vectors (line 2). If both vectors have the exact same word in the same position we add 1 to the final score (line 3). Otherwise, we compare the word in the first vector to every other word in σ_2 (line 6). If the same word is found in a different position, we add 0.7 to the final similarity score (line 8) representing a penalty for the different positions of the same word. If the same word is not found in a different position, the algorithm tries to compare the similarity between the synonyms of both words being compared (lines 11 and 12) keeping the highest semantic similarity between them. We calculate the semantic similarity for individual words using the Wu-Palmer (WUP) (Wu and Palmer, 1994) measure provided by WordNet, which generates a score that represents how semantic similar two word senses are based on the depth of the two senses in the taxonomy and their most specific ancestor node (the SIMILARITY function in Line 13). We chose the WUP measure by comparing it with four other measures provided by WordNet, namely Path Length, Resnik (1995), Lin (1998), Jiang and Conrath (1997), and Leacock and Chodorow (1998). After iterating over all words, we add the highest value to the final score (line 15). Finally, we normalise the similarity score by the mean length of both sentences (line 16).

In summary, the algorithm has three main cases: first, add 1 to the final score when the same word appears in the same position of both norm actions; second, add 0.7 to the final score when a word appears in both norm actions but in different positions. This value is a penalty due to the word different position; third, find the highest value of semantic similarity between a word in the first norm action and the words of the second one. Example 20 illustrates the application of the semantic similarity algorithm over norm actions resulting in a similarity of 60%.

Algorithm 1 Calculates the semantic similarity between two norm actions.

```
Require: $|\sigma_1| \leq |\sigma_2|$
 1: procedure COMPUTE_SIMILARITY($\sigma_1, \sigma_2$)
 2:     for $ind_1$ in $\sigma_1$ do                              ▷ Iterate over the indexes of $\sigma_1$
 3:         if $\sigma_1[ind_1]$=$\sigma_2[ind_1]$ then $sim \leftarrow sim + 1$
 4:         else
 5:             $sim_{max} \leftarrow -\infty$
 6:             for $ind_2$ in $\sigma_2$ do
 7:                 if $\sigma_1[ind_1]$=$\sigma_2[ind_2]$ then
 8:                     $sim_{max} \leftarrow 0.7$
 9:                     break
10:                 else
11:                     $s_1 \leftarrow$ synonyms of $\sigma_1$
12:                     $s_2 \leftarrow$ synonyms of $\sigma_2$
13:                     $sim_{1,2} \leftarrow \max_{s_1,s_2}(\text{SIMILARITY}(s_1, s_2))$
14:                     $sim_{max} \leftarrow \max(sim_{max}, sim_{1,2})$
15:             $sim \leftarrow sim + \max(0, sim_{max})$
16:     return $sim/\text{MEAN}(len(\sigma_1), len(\sigma_2))$
```

Example 20 (Semantic Similarity: 0.60)

– Cubist shall <act>at all times upon reasonable notice and during normal business hours have the right to inspect the Facility to ascertain compliance with GMPs.</act>
– Cubist may <act>be, at all times and upon reasonable notice and normal business hours, denied an inspection of the Facility to ascertain compliance with GMPs.</act>

Once calculated, the semantic similarity between norm actions allows us to detect potential conflicts between norms with the same parties. To this end, we define a threshold of 60%, in which norm pairs that have norm actions with semantic similarity equal or greater than 60% are considered conflicts. We chose this threshold by testing different values for it (see Section 4.2.3). We conclude that thresholds smaller than 60% identify norm actions that do not present semantic similarity, whereas thresholds greater than 60% exclude norm actions semantic similar. Figure 2 illustrates the detection of a potential conflict. It consists of the comparison between each component of the norm representation, namely the parties, the deontic meanings, and the norm actions.

4.2 Evaluation

4.2.1 Norm Identification

To evaluate the norm identification process, we used the Australian contract corpus (Curtotti and McCreath, 2011) to create a gold standard. The corpus consists of 256 unannotated contracts, which are composed of different contract

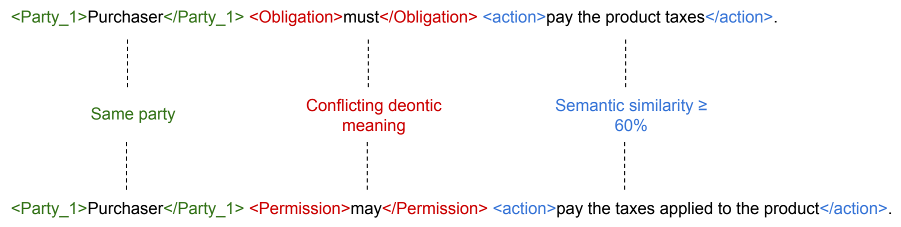
<Party_1>Purchaser</Party_1> <Obligation>must</Obligation> <action>pay the product taxes</action>.
Same party
Conflicting deontic meaning
Semantic similarity ≥ 60%
<Party_1>Purchaser</Party_1> <Permission>may</Permission> <action>pay the taxes applied to the product</action>.

Fig. 2 Example of potential conflict detection.

types involving different subject matters (e.g. business contracts, home leases, among others). We manually annotated 92 contracts and we created two sets: a norm sentence set with 9864 norms and a common sentence (non-norm) set with 10554 sentences. We test our approach over the 92 annotated contracts and we obtain 79% of precision, 98% of recall, and 87% of F-measure. The errors we found are related to contract sentences containing either modal verbs that do not represent a norm or sentences that use other types of expressions to describe a norm, such as "*Company X* is obliged to".

4.2.2 Party Identification

In order to evaluate the party identification, we manually evaluated an execution of the party identifier over 100 contracts from the manufacturing type (extracted from `http://contracts.onecle.com/type/47.shtml`). As result, we obtained an accuracy of 60% considering the whole task of identifying party names, aliases, and their relation. The errors arise from differences between the patterns, such as: contracts with three parties (4 errors); party descriptions that do not use "between"/"and" pattern (10 errors); aliases pattern does not match (17 errors); and other errors related to the link between party name and alias.

4.2.3 Semantic Similarity Calculus

We compare the results of our semantic similarity algorithm to those presented by Li et al (2006). They propose an algorithm to compute the similarity between sentences, which considers both semantic and word order information. They apply their algorithm to 16 sentence pairs showing the similarity values obtained. The semantic similarity value varies from 0 to 1, where 0 means no similarity between the sentences and 1 means equal sentences. We use these sentence pairs to compare the results with theirs. Since our approach uses the WUP measure to calculate the semantic similarity between words, we created an approach that does not use the WUP MEASURE to compare the difference between them. Such approach, instead, compares whether two words are equal without measuring their semantic similarity. We call this approach

Simple_Similarity and we add it in our comparison. Table 2 shows the obtained results.

We conclude that the algorithms present differences in their results, such as the number 4 of Table 2. Our approach presents more precise results for some sentence pairs, such as the tuple (“I like that bachelor.”, “I like that unmarried man.”) (number 8 in Table 2). However, when calculating the similarity between (“Red alcoholic drink.”, “An English dictionary.”) it results in a much higher similarity than the others, showing that, in some cases, our algorithm returns high values to words with no semantic similarity. It occurs due to the algorithm procedure, which compares each word to a vast list of synonyms often finding a word with a high similarity. In comparison to Simple Similarity, which does not use WUP similarity measure, we obtained better results when applied to sentence pairs that have similar meanings but different phrasal constructions. As an example, the sentence pair (“Red alcoholic drink.”, “A bottle of wine.”) in which WUP Similarity computes 0.31 of similarity while Simple Similarity obtains only 0.14. Since Simple Similarity does not take into consideration the synonyms of a word, sentence pairs like the previous example receive lower similarity. In the sentence pair (“A glass of cider.”, “A full cup of apple juice.”), WUP Similarity obtains 0.5 while Simple Similarity obtains 0.34, which shows that Simple Similarity limits its computation to the equality of the words. Although our approach has limitations, it is much simpler to implement than the one presented by Li *et al.* and slightly better than a simple comparison between words. For these reasons, we use Algorithm 1 in our approach to compute the semantic similarity between sentences.

Table 2 Comparison between the approaches using WUP Similarity Measure; Simple Similarity; and Li et al.

	Sentence Pair	**Similarity_WUP**	**Simple_Similarity**	**Li *et al.***
1	(“I have a pen.”, “Where do you live?”)	0.08	0	0
2	(“Red alcoholic drink.”, “An English dictionary.”)	0.43	0.25	0
3	(“I have a pen.”, “Where is ink.”)	0.32	0.14	0.12
4	(“I have a hammer.”, “Take some apples.”)	0.37	0.24	0.12
5	(“Canis familiaris are animals.”, “Dogs are common pets.”)	0.34	0.34	0.36
6	(“Red alcoholic drink.”, “Fresh apple juice.”)	0.49	0.25	0.42
7	(“I have a hammer.”, “Take some nails.”)	0.34	0.24	0.5
8	(“I like that bachelor.”, “I like that unmarried man.”)	0.72	0.62	0.56
9	(“Red alcoholic drink.”, “A bottle of wine.”)	0.31	0.14	0.58
10	(“Red alcoholic drink.”, “Fresh orange juice.”)	0.45	0.25	0.61
11	(“It is a dog.”, “It is a log.”)	0.85	0.8	0.62
12	(“A glass of cider.”, “A full cup of apple juice.”)	0.5	0.34	0.67
13	(“It is a dog.”, “That must be your dog.”)	0.38	0.32	0.73
14	(“Dogs are animals.”, “They are common pets.”)	0.43	0.34	0.73
15	(“It is a dog.”, “It is a pig.”)	0.85	0.8	0.79
16	(“John is very nice.”, “Is John very nice?”)	0.68	0.68	0.97

In order to identify the best threshold value for semantic similarity between norm actions, we performed a comparison using five different threshold values: 40%, 50%, 60%, 70%, and 80%. To compare them, we measured how each value impacts the final result in the potential conflict identification. Using the set of contracts with manually inserted conflicts as the gold standard (we detail the process of generating the gold standard in Section 4.2.4), we test the potential conflict identifier for each threshold value. Table 3 shows the results using different measures. For a better understanding, we based our calculations on the total number of norm pair candidates, i.e., from the set of identified norms, we calculate the total number of norm pair combinations. For example, a set of 5 norms can form a total of 10 norm pairs. For the 16 contracts we have in the gold standard set, there are 11,928 norm pairs. Since we identify the norm set using the Norm Identification algorithm during the contract processing, some norms may not be identified. The gold standard contains 121 norm conflicts out of the 11,928 norm pairs, i.e., around 1% of the total number of norm pairs. In the evaluation, we consider to be true positives (tp) the norm pairs identified by the potential conflict identifier that are part of the 121 norm conflicts. We consider true negatives (tn) to be the norm pairs that have no conflict and were not identified as conflicting. False positives (fp) are the norm pairs identified as potential conflicts but have no conflict, and false negatives (fn) are the conflicting norm pairs that were not identified as such. Using these four measures (detailed in Appendix B), we calculate accuracy (Equation 2), precision (Equation 3), recall (Equation 4), f-score (Equation 5), true positive rate (Equation 6), specificity (Equation 7), and false positive rate (Equation 8). Table 3 shows the measures and their result for each threshold value. We highlight the best results for each measure.

As results show, when we apply a small threshold value (of 40%) for semantic similarity we get the highest number of true positives and false negatives, which results in a higher recall and true positive rate. However, using a threshold value of 40% we also get the highest number of false positives, i.e., although we find most of the conflicting norm pairs (83), we identify 189 other norm pairs that are not conflicts. This high number of false positives results in a small precision (0.31) and consequently a small f-score (0.42). When we apply a high threshold value (of 80%), we get the highest number of true negatives and false positives, which represent a conservative selection of potential conflicts. However, this threshold also yields the smallest number of true positives (31) and false negatives (90), which result in small recall (0.26) and f-score (0.39) values. These results show that although a high threshold results in less errors than other values, it also identifies less conflicts, which is the main goal in this work. Using a threshold value of 50%, we identify only 2 (two) conflicts less than the 40% threshold and we also get a much smaller value of false positives (32). However, using the 50% threshold value we do not get the highest values of precision, recall, and f-score. Using a threshold of 70%, we get similar results when comparing to the 80% threshold, which results in small values of recall, and f-score. This threshold value gets the highest precision (of 0.82) due to the few false positives (10), however, it also results in only 47 true positives, a small number when we compare to the 60% threshold value (80). Finally, using a threshold of 60% we get only three true positives lesser than the 40% threshold and we get a much lesser value of false positives (21). These values result in good precision and recall measures (0.79 and 0.66, respectively) and the higher f-score (0.72). Considering these results, we conclude that a threshold of 60% is the best choice for the task of identifying semantic similar between norm actions for this particular dataset. We use the f-score measure as the main indicator of how well a threshold value performs. Our accuracy measure (the one in Equation 2, not the one described in Equation 1), specificity, and false positive rate do not vary significantly for any threshold value due to the high number of true negatives. Although the true positive rate measure does vary with threshold values because the number of true positives changes, it ignores the number of false positive, which is one of the main issues our technique aims to mitigate (i.e. a large number of potential conflicts for human contract reviewer).

Table 3 Threshold values comparison

Threshold	40%	50%	60%	70%	80%
True Positives	**83**	81	80	47	31
True Negatives	11618	11775	11786	11797	**11799**
False Positives	189	32	21	10	**8**
False Negatives	**38**	40	41	74	90
Accuracy	0.98	**0.99**	**0.99**	**0.99**	**0.99**
Precision	0.31	0.72	0.79	**0.82**	0.79
Recall	**0.69**	0.67	0.66	0.39	0.26
F-Score	0.42	0.69	**0.72**	0.53	0.39
True Positive Rate	**0.69**	0.67	0.66	0.39	0.26
Specificity	0.98	**1.00**	**1.00**	**1.00**	**1.00**
False Positive Rate	0.02	**0.00**	**0.00**	**0.00**	**0.00**
Total number of conflicts			121		

4.2.4 Potential Conflict Identifier

The potential conflict identifier comprises all previous procedures, i.e., to perform this identification, we need to apply the entire process described in Figure 1. To evaluate this kind of identification, we need a corpus containing real norm conflicts. However, after extensive search, we could not find a corpus with real normative conflicts. Thus, we chose to manually create norm conflicts using a set of real norms as a basis, and apply a method to guide a user to insert conflicts given a set of norms (see A). Using the resulting conflicts in contracts, we execute our potential conflict identifier over the altered contracts.

For this task we divided two volunteers with no knowledge of the detection algorithms into two different types of conflict insertion. We asked the first volunteer to insert conflicts that have only differences in the modal verb, e.g. changing an obligation modal verb ('must') for a permission one ('may'). The volunteer created 94 conflicts in 10 different contracts, totalling 13 conflicts of type (1) (Permission x Prohibition), 36 conflicts of type (2) (Permission x Obligation), and 46 conflicts of type (3) (Obligation x Prohibition). Conversely, we asked the second volunteer to insert conflicts that contain deontic conflicts and modifications in the norm actions. Such modifications include maintaining the same meaning of norm actions but changing the words in order to have a different structure from the previous norm. Since inserting such conflicts demands more time and English language knowledge, the second volunteer created only 17 conflicts in 6 different contracts, totalling 2 conflicts of type (1) (Permission x Prohibition), 6 conflicts of type (2) (Permission x Obligation), and 4 conflicts of type (3) (Obligation x Prohibition). We execute the potential conflict identifier over 16 contracts from these conflicts.

The process of conflict creation is conducted by the user that selects an option for each process. First, the user selects a random contract, which the system selects randomly from the contract corpus. Following, the user may choose either to get another contract or get a random norm from the contract. The user then copies the random norm and makes the necessary modifications. From this point, the user may choose either to start all over again, get another norm from the same contract, or finish the conflict insertion. At any point, the user can come back to a previous step in the conflict insertion. Figure 4.2.4 illustrates the process in which the user is submitted to create a new conflict. Since the two volunteers made a semi-scripted contract modification, we do not consider measuring an agreement level on the modifications.

We acknowledge that this is not the best way to evaluate our approach. However, we did not have access to domain specialists that could be able to create reliable conflicting cases. As we know that this is a serious concern about our evaluation method, we decide to make the conflict creation process as clear as possible and accept that our approach still needs to be evaluated either in real cases or in conflicts created by specialists.

We perform the evaluation according to each type of conflict insertion, then we make a final round considering all conflicts. Table 4 shows the results we obtained with the first type of conflict. As the results show, we obtain an accuracy of 79% for the task. This accuracy, as well as the others, is a weighted average that sums the product of each contract accuracy by the total number of conflicts in the contract and then divide it by the total number of conflicts. Given $n\_contracts$ as the total number of contracts, $conflicts_{(i)}$ as the total number of conflicts in contract i, $accuracy_{(i)}$ as the accuracy obtained in contract i (which is the total number of conflicts found divided by the total number of conflicts in the contract), and $n\_conflicts$ as the total number of conflicts overall contracts, Equation 1 shows how we calculate the weighted average for the conflict identifier evaluation.

$$accuracy = \frac{\sum_{i=1}^{n\_contracts} conflicts_{(i)} * accuracy_{(i)}}{n\_conflicts} \tag{1}$$

Among the results, we identified 11 conflicts of type (1), 32 of type (2), and 31 of type (3), in addition to two extra potential conflicts that were not in the conflict set. The extra potential conflicts are two norm pairs found by the potential conflict identifier that were not inserted by volunteers. It means that these norm pairs could be two norm conflicts, however, we did not count them on the results since they were not defined on the gold standard. We can explain

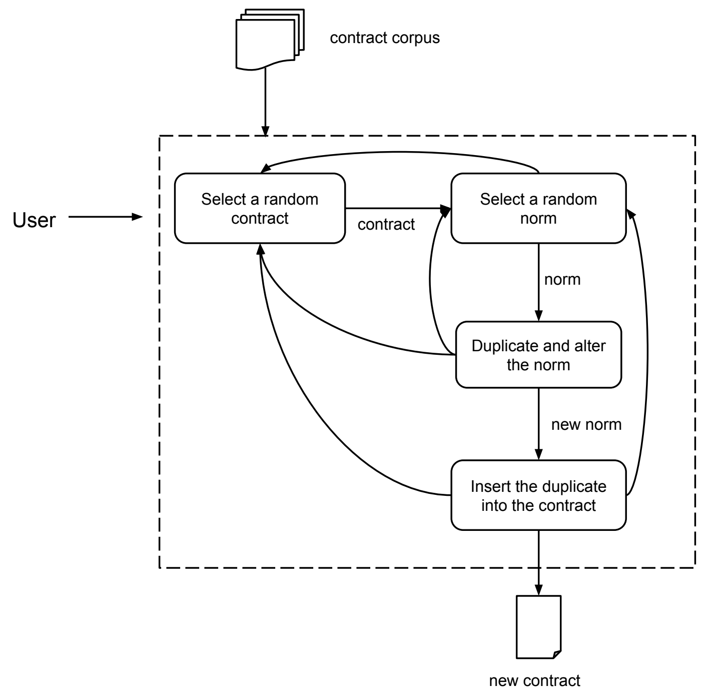

Fig. 3 Conflict insertion procedure.

the errors by considering the error rate in each subprocess that constitute the norm representation. Consequently, either an error in the norm identifier or in the party identifier may reflect on the final result. Since this set of conflicts is made of simple modifications in the modal verb, we expected good results. For this reason, we tested our potential conflict identifier over the second set of conflicts, which is made of modifications in the modal verb and in the norm action structure. Table 5 presents the results we obtained. We obtained a lower accuracy (70%) if compared with the previous set of conflicts, since in this one we have differences in the norm actions. Although the set has a small number of conflicts, the results show that in some contracts we identified all inserted conflicts. This confirms that the semantic similarity works to identify norm actions with the same meaning. Finally, Table 6 presents the accuracy overall conflicts without distinction. We obtain an accuracy of 78% in the potential conflict identification.

Table 4 Results of the potential conflict identifier execution over the first set of conflicts.

Contract	Conflicts	Identified	Accuracy
roth-mfg	14	11	0.78
chiron.mfg	8	8	1.0
ibm.mfg	7	5	0.71
johnson.mfg	7	5	0.71
southeast.mfg	6	3	0.5
nellson.supply	11	9	0.81
medica.mfg	17	14	0.82
usfoodservice	12	9	0.75
foamtec.mfg	2	1	0.5
cardinal.mfg	10	9	0.9
10	94	74	**0.79**

Table 5 Results of the potential conflict identifier execution over the second set of conflicts.

Contract	Conflicts	Identified	Accuracy
hershey.mfg	4	1	0.25
astrazeneca	4	1	0.25
aortech.mfg	2	2	1.0
cypress-mfg	3	3	1.0
cinram.mfg	1	1	1.0
dsm-capua.mfg	4	4	1.0
6	17	12	**0.70**

Table 6 Overall results of the potential conflict identifier.

Contract	Conflicts	Identified	Accuracy
16	111	86	**0.78**

5 Related Work

In this section, we briefly describe the key approaches from previous work that deals with issues related to contract processing, norm conflict identification and deontic reasoning in the real world. We start with a summary of work that deals directly with contracts, followed by approaches that deal with the normative and deontic aspect of contractual reasoning. First, our work was originally inspired by the availability of a contract corpus developed by Curtotti and McCreath (2011). Both authors propose an approach for annotating contracts using machine learning and rule-based techniques (Curtotti and Mccreath, 2010). The aim of the work is to classify sentence components in contracts according to their structure.

Similar to Curtotti and Mccreath, Athan et al (2013) develop a formal language, called LegalRuleML, to deal with contract annotation, specifically, the annotation of legal elements in contracts, such as norm agents, temporal restrictions, and events. LegalRuleML is an XML-based language that extends RuleML (Governatori, 2005), which is an XML based language for the representation of rules. The aim of the work is to improve the modelling of semantic norms, providing an expressive XML standard for modelling normative rules that satisfies legal domain requirements.

Both approaches from Curtotti and Mccreath and Athan et al introduce new mechanisms to annotate contracts in order to structure data for text processing. Although we use similar techniques to annotate contract clauses, we adopt a coarser annotation approach and focus on the semantic similarity of norm actions in order to find potential conflicts.

The following papers bring approaches to extract information from legal texts. Peters and Wyner (2016) develop a system to make a semi-automatic analysis of assigning Hohfeldian relations of Duty. Hohfeldian relations are regulative rules that specify actions, party roles to the actions, and the objects of actions. They develop a workflow with seven stages. In the first and second stages, they process the raw text extracting NLP features, such as POS-tag and lemmas. In the third and fourth stages, they identify terms related to Hohfeldian relations and the deontic structures in text. Using such information, in the last stages, they identify Hoefeldian concepts, have an expert evaluation, and the resulting annotations populate an ontology. Finally, they evaluate their approach using two documents and annotating Hohfeldian roles within them.

Gao et al (2012) propose an approach for extracting exceptions within norms in contracts, which arises when a norm contains a condition under which it is not to be applied. The resulting *Enlil* system uses natural language processing and machine learning to process contracts and identify service exceptions or contingency conditions within contracts. Follow-up work from Gao and Singh (2013) develops a hybrid approach for extracting business events and their temporal constraints from contracts. Their aim is to create a system that receives a contract as input and returns its events and temporal constraints. Gao and Singh (2014) create a mechanism for extracting and classifying norms in contracts, which is closer to the first part of the approach described in this paper. Using a modal verb filter and then extracting a feature vector, they create a classifier to extract norms from contracts. Although our norm identification mechanism is conceptually similar to the latter work, unlike these efforts, which focus on extracting specific information from contracts, we go further and identify potential conflicts between contract clauses (norms).

Wyner and Peters (2011) propose an approach to extract rules and their definitions from regulations. The extraction of rules is made by using a rule-based approach. Then, they use NLP techiniques to extract information from rule texts and use it to identify exception and condition elements in rules.

The following approaches aim to identify norm conflicts in multi-agent systems.Figueiredo and da Silva (2013) develop an algorithm to identify conflicts between norms within multi-agent systems formalised using the Z specification language. Likewise,Vasconcelos et al (2009) develop a mechanism to identify normative conflicts for multi-agent systems, based on a formal representation of norms with constraints, presenting formal definitions of normative conflicts and defining how they can be resolved. These efforts, however, assume formally specified norms with no semantic ambiguity, which is not the focus of our work.

Abstracting away from the natural language of contracts, several approaches assume legal documents in some formal language and focus on identifying norm conflicts. Rosso et al (2011) propose an approach to retrieve information from legal texts. They use a system named JIRS[2] (Java Information Retrieval System) that measures distances between sentences using n-grams. In their work, they deal with three problems: passage retrieval in treaties, patents, and contracts. They propose an approach that uses JIRS to identify conflicts in contracts. Based on a contract example between an airline and a ground operations company, they apply a series of rules to identify conflicts between norms. The process of conflict identification is divided into three main steps First they translate the contract from a formal contract language ($\mathcal{CL}$ (Fenech et al, 2009a)) to Contract Language clauses. Second, they analyse the clauses using a model checker performed by the contract analysis tool *CLAN* (Fenech et al, 2009b). From the identified conflicts, they use JIRS to translate the sentences from contract language to natural language.

Gorín et al (2011) develop a software tool for legal drafting stating that it is possible to use existing approaches to analyse software specifications in normative documents. The tool uses a linear temporal logic (LTL)-based (FormaLex) language and companion tools that use model checking to detect normative incoherence in legal documents. Pace and Schapachnik (2012) propose an automata-based semantics to formalise the onuses that deontic meanings (obligations, permission, and prohibitions) on one party imposes on the other. They use an existing automaton-based model to represent the interaction of a two-party system and extend it to deal with the absence of actions, mutually exclusive actions, and conflicts. Gabbard et al (2015) develop a system that helps lawyers with the contract drafting process by alerting about missing contract components and giving feedbacks to improve the contract quality. They use NLP techniques and machine learning to verify natural language contracts and alert about missing components. Their system has an online and an offline module in which the user can upload a contract and receive a feedback to improve the contract. Azzopardi et al (2016a) develop a tool to provide intelligent contract editing to lawyers by using off-the-shelf NLP techniques. They extract information from clauses in order to make an intelligent browsing among the clauses. The authors use a mix of regular expressions and named entity recognition to identify the clauses parties. In order to make the connection between natural language and a formal language, the authors propose a deontic logic representation that allows them to identify conflicts. Azzopardi et al (2016b) introduce an approach to automatically translate natural language into a deontic logic formalisation that allows automatic reasoning for conflict identification. Their approach deals with partiality and uses a formal analysis to enable reasoning under incomplete knowledge. The authors propose a case study containing two clauses with unknown information. Using the definitions in their logic representation they were able to identify a conflict between the clauses. Thus, although these papers deal with a variety of techniques for conflict detection for norms using formalisms of varying degrees

[2] http://sourceforge.net/projects/jirs/

of complexity, they assume legal documents in some formal language instead of contracts, as written by humans, which our work addresses.

The papers described above propose approaches to deal with norm conflicts in natural language The main difference between their work and ours is that most of them rely on a deontic logic representation to find conflicts. Such representation facilitates the comparison between norm elements and the conflict checking. In our work, we perform the norm conflict identification by directly comparing the elements identified in the norm texts.

6 Discussion

Although the approach we develop in this paper is capable of processing a variety of contracts using traditional NLP techniques to identify norms and potential conflicts using a very small dataset, it suffers from a number of limitations. In what follows, we highlight and discuss these limitations as pointers for further improvement of our approach.

6.1 Deontic Logic Paradoxes

The conflict identification process we use in our approach is based on a naive deontic logic that suffers from well-known paradoxes. Most of them occur with the O-operator (obligation) under logical consequence, which deontic logic states that "$if \vdash A \rightarrow B \ then \vdash OA \rightarrow OB$". This logical equation means that if we have a logical consequence where a variable proposition A implies B, we can say that an obligation of A implies in an obligation of B (Carmo and Jones, 2002). We describe below some of these paradoxes, better described by Carmo and Jones (2002).

Ross Paradox ($\vdash OA \rightarrow O(A \vee B)$):

This paradox allows that the following sentence be valid in deontic logic:

"If it is obligatory to pay the rent, then it is obligatory to pay the rent or to burn the house"

Analysing this paradox, one can say that since the implication of A implies in the obligation of A or B, if it is not the case of A (if we perform B), we have a violation.

Free Choice Permission paradox

The paradox arises due to the fact that in standard deontic logic it is not a logical consequence that $P(A \vee B) \rightarrow (PA \wedge PB)$, whereas one can understand that if A or B is permitted, then A and B are permitted. Besides, we cannot add $P(A \vee B) \rightarrow (PA \wedge PB)$ as a deontic logic axiom because together with $\vdash PA \rightarrow PA \vee PB$, it would imply that $PA \rightarrow (PA \wedge PB)$ and, thus, the permission to do A implies in the permission of doing A and B, where B can be anything.

Good Samaritan paradox:

This paradox arises due to the deontic logic theorem: $\vdash O(A \wedge B) \rightarrow OB$. Therefore the following sentence is a valid one:

> "if it is obligatory that John helps Jack who had an accident, than it is obligatory that Jack has an accident"

The problem here is that once we have an obligation of a conjunction, both logical variables become obligations. In this case, the second variable states a condition that cannot be obligatory due its definitions. Since our work only borrows the concepts of deontic meanings, such as obligation, permission, and prohibition in a semi-formal way, our approach does not suffer from these paradoxes.

6.2 Corpus Size

The lack of a substantial corpus of contracts containing conflicting clauses constitutes a key limitation of the experimentation of our approach. After an extensive search, we found no suitable corpus to evaluate our approach, and consequently had to rely on a semi-automatically constructed corpus containing 111 conflicts between norms. Given the relatively small corpus, the number of conflict types present in the corpus is also limited. Consequently, our current research relies on an open web tool to help collect a variety of contracts in natural language in order to expand such evaluation.

6.3 Reparations and Contrary-to-Duty

The relatively simple deontic formalism used by our approach entails that it does not deal with reparation norms from contrary-to-duties. We decided to use a norm structure divided into three components, namely: party name, modal verb (deontic meaning), and norm action. This structure limits the analysis of conditions and, thus, the possibility of contrary-to-duty identification. We acknowledge that the absence of a contrary-to-duty structure makes our approach prone to misleading norm conflicts. As a future improvement, we aim to detect norms and classify them according to their structure, allowing us to use specialised norm structures to look for conflicts in specific cases.

6.4 Conflict Cases

As we describe in Section 4, our approach detects conflicts between norms following three rules:

- Both norms are applied to **the same party**;
- Both norms have **conflicting deontic meanings**; and
- Both norms refer to **the same act**, i.e., the same consequent in which the party must fulfil.

Although these rules ensure the identification of many conflicting cases, they are unable to identify more complex conflicts. Since we assume that each norm has only one party, in cases where the norm refers to two or more parties, we only consider the last one (the one closer to the modal verb) in our norm comparison. Finally, the absence of a definition for various conditional statements for the norm means that conditional norms with explicit activation and expiration clauses are reasoned about as norms without conditions. Nevertheless, norms with similar actions and conditions will naturally be semantically similar, and may be classified as potential conflicts. We chose this norm structure (with party name, modal verb, and norm action) since we noticed that many norm sentences follow this well-known pattern from natural language (subject, verb, and object). We aim to address more complex cases of norm conflicts in future work.

7 Conclusion

In this work, we developed an approach to identify potential conflicts between norms in contracts. In this paper, we have four main contributions: first, we create a norm identifier that allows us to potentially identify norm sentences in a contract; second, we create a party identifier that identifies the parties and their aliases in contracts with two parties; third, we propose a prototypical representation to norms with three components, namely party name, modal verb, and norm action; finally, we suggest that measures the semantic similarity between sentences will be of benefit and put forward an algorithm to do so. Along this work, we also curated dataset that is available for other researchers[3] (Aires et al, 2017), including the sets of norm sentences and common sentences used in the norm identification phase and the small contract corpus with a set of artificially created conflicting norms used to evaluate our approach.

As future work, we plan to generalise our approach in order to include other contract structures. The current approach deals with contracts with at most two parties and follows a certain pattern of parties description. As an alternative, we can either create a more general rule for the party identification or invest in a machine learning approach to recognise parties and aliases together with named entity recognition techniques. We plan to improve the semantic similarity algorithm, since it considers extra aspects that are often unnecessary to the norm similarity calculus, such as long lists of synonyms. Even knowing that conflicts are rare in real contracts we want to create a richer conflict corpus, larger and with a better variation of study cases of conflicts. We also intend to consider approaches of extrinsic evaluation of this work in form of contract writing support.

[3] `https://github.com/JoaoPauloAires/norm-dataset`

8 Acknowledgements

We would like to thank HP PROFCSI for funding our research. Felipe thanks CNPq for support within grant number 306864/2016-4 under the PQ fellowship project.

References

Aires JP, Pinheiro D, Meneguzzi F (2017) Norm Dataset: Dataset with Norms and Norm Conflicts. DOI 10.5281/zenodo.345411, URL `https://doi.org/10.5281/zenodo.345411`

Athan T, Boley H, Governatori G, Palmirani M, Paschke A, Wyner A (2013) Oasis legalruleml. In: Proceedings of ICAIL, pp 3–12

Axelrod R (1986) An evolutionary approach to norms. The American Political Science Review 80(4):pp. 1095–1111

Azzopardi S, Gatt A, Pace GJ (2016a) Integrating natural language and formal analysis for legal documents. In: Language Technologies & Digital Humanities

Azzopardi S, Gatt A, Pace GJ (2016b) Reasoning about partial contracts. In: Legal Knowledge and Information Systems - JURIX 2016: The Twenty-Ninth Annual Conference, pp 23–32, DOI 10.3233/978-1-61499-726-9-23, URL `http://dx.doi.org/10.3233/978-1-61499-726-9-23`

Bird S (2006) NLTK: the natural language toolkit. In: ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006, URL `http://aclweb.org/anthology/P06-4018`

Carmo J, Jones AJI (2002) Deontic Logic and Contrary-to-Duties, Springer Netherlands, Dordrecht, pp 265–343. DOI 10.1007/978-94-010-0387-2_4, URL `http://dx.doi.org/10.1007/978-94-010-0387-2_4`

Curtotti M, Mccreath E (2010) Corpus based classification of text in australian contracts. In: Proceedings of the Australasian Language Technology Association Workshop, Melbourne, Australia, pp 18–26

Curtotti M, McCreath EC (2011) A corpus of australian contract language: Description, profiling and analysis. In: Proceedings of the 13th International Conference on Artificial Intelligence and Law, ACM, New York, NY, USA, ICAIL '11, pp 199–208, DOI 10.1145/2018358.2018387

Fenech S, Pace GJ, Schneider G (2009a) Automatic conflict detection on contracts. In: International Colloquium on Theoretical Aspects of Computing, Springer, pp 200–214

Fenech S, Pace GJ, Schneider G (2009b) CLAN: A tool for contract analysis and conflict discovery. In: Liu Z, Ravn AP (eds) Automated Technology for Verification and Analysis, 7th International Symposium, ATVA 2009, Macao, China, October 14-16, 2009. Proceedings, Springer, Lecture Notes in Computer Science, vol 5799, pp 90–96, DOI 10.1007/978-3-642-04761-9_8

Figueiredo KS, da Silva VT (2013) An algorithm to identify conflicts between norms and values. In: Coordination, Organisations,Institutions and Norms in Multi-Agent Systems, pp 259–274

Gabbard J, Sukkarieh JZ, Silva F (2015) Writing and reviewing contracts: Don't you wish to save time, effort, and money? In: Proceedings of the 15th International Conference on Artificial Intelligence and Law, ACM, New York, NY, USA, ICAIL '15, pp 229–230, DOI 10.1145/2746090.2746534, URL `http://doi.acm.org/10.1145/2746090.2746534`

Gao X, Singh MP (2013) Mining contracts for business events and temporal constraints in service engagements. Services Computing, IEEE Transactions on PP(99):1–1, DOI 10.1109/TSC.2013.21

Gao X, Singh MP (2014) Extracting normative relationships from business contracts. In: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems, International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, AAMAS '14, pp 101–108

Gao X, Singh MP, Mehra P (2012) Mining business contracts for service exceptions. IEEE Transactions on Services Computing 5(3):333–344

Gorín D, Mera S, Schapachnik F (2011) A software tool for legal drafting. In: Proceedings Fifth Workshop on Formal Languages and Analysis of Contract-Oriented Software, FLACOS 2011, Málaga, Spain, 22nd and 23rd September 2011., pp 71–86, DOI 10.4204/EPTCS.68.7, URL `https://doi.org/10.4204/EPTCS.68.7`

Governatori G (2005) Representing business contracts in *ruleml*. International Journal of Cooperative Information Systems 14(2-3):181–216

Jenks CW (1953) The conflict of law-making treaties. BYIL 30:401

Jiang JJ, Conrath DW (1997) Semantic similarity based on corpus statistics and lexical taxonomy. CoRR URL `http://arxiv.org/abs/cmp-lg/9709008`

Jones AJI, Sergot MJ (1992) Deontic logic in the representation of law: Towards a methodology. Artificial Intelligence and Law 1(1):45–64

Kollingbaum MJ, Norman TJ, Preece A, Sleeman D (2007) Norm conflicts and inconsistencies in virtual organisations. In: Coordination, Organizations, Institutions, and Norms in Agent Systems II, Springer, pp 245–258

Leacock C, Chodorow M (1998) Combining local context and wordnet similarity for word sense identification. WordNet: An electronic lexical database 49(2):265–283

Li Y, McLean D, Bandar Z, O'Shea J, Crockett KA (2006) Sentence similarity based on semantic nets and corpus statistics. IEEE Transactions on Knowledge and Data Eng 18(8):1138–1150, DOI 10.1109/TKDE.2006.130

Lin D (1998) An information-theoretic definition of similarity. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998, pp 296–304

Pace GJ, Schapachnik F (2012) Contracts for interacting two-party systems. In: Proceedings Sixth Workshop on Formal Languages and Analysis of Contract-Oriented Software, FLACOS 2012, Bertinoro, Italy, 19 September 2012., pp 21–30, DOI 10.4204/EPTCS.94.3, URL https://doi.org/10.4204/EPTCS.94.3

Palmer FR (2001) Mood and modality (2nd Edition). Cambridge University Press

Peters W, Wyner AZ (2016) Legal text interpretation: Identifying hohfeldian relations from text. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016., URL http://www.lrec-conf.org/proceedings/lrec2016/summaries/253.html

Resnik P (1995) Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes, pp 448–453

Rosso P, Correa S, Buscaldi D (2011) Passage retrieval in legal texts. The Journal of Logic and Algebraic Programming 80(3-5):139–153, DOI 10.1016/j.jlap.2011.02.001

Rousseau DM, McLean Parks J (1993) The contracts of individuals and organizations, vol 15. JAI PRESS LTD

Sadat-Akhavi A (2003) Methods of Resolving Conflicts Between Treaties. Graduate Institute of International Studies (Series), V. 3, M. Nijhoff

Steedman MJ (1977) Verbs, time, and modality. Cognitive science 1(2):216–234

Vasconcelos WW, Kollingbaum MJ, Norman TJ (2009) Normative conflict resolution in multi-agent systems. Autonomous Agents and Multi-Agent Systems 19(2):124–152, DOI 10.1007/s10458-008-9070-9, URL http://dx.doi.org/10.1007/s10458-008-9070-9

Vranes E (2006) The definition of 'norm conflict' in international law and legal theory. European Journal of International Law 17(2):395–418

Winikoff M, Padgham L, Harland J, Thangarajah J (2002) Declarative & procedural goals in intelligent agent systems. In: Proceedings of the Eights International Conference on Principles and Knowledge Representation and Reasoning (KR-02), Toulouse, France, April 22-25, 2002, pp 470–481

von Wright GH (1951) Deontic Logic, New Series, vol 60. Oxford University Press on behalf of the Mind Association

Wu Z, Palmer M (1994) Verbs semantics and lexical selection. In: Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '94, pp 133–138, DOI 10.3115/981732.981751

Wyner AZ, Peters W (2011) On rule extraction from regulations. In: Legal Knowledge and Information Systems - JURIX 2011: The Twenty-Fourth Annual Conference, University of Vienna, Austria, 14th-16th December 2011, pp 113–122, DOI 10.3233/978-1-60750-981-3-113, URL https://doi.org/10.3233/978-1-60750-981-3-113

A Instructions used by human volunteers for conflict creation

The following text was used to guide the volunteers during the norm conflict insertion. The document consists of a description for each step of the system.

This README explains how to execute `contract_data_structure.py` to introduce conflicts randomly in contracts from our corpus.[4] The main goal is to create contracts containing norm conflicts independently from a conflict detection algorithm.

Execute:

- To execute, run the following command:
  `python -B contract_data_structure.py`

Options:

- After login, there will be two initial options:
  (1) Pick a random contract; and
  (4) Finish.

(1) This option chooses a contract from the corpus at random. This step may return an error due to the choice of a contract without norms; if that happens, ignore the error and press (1) again. If no error occurs, the program will display information, such as the total number of norms extracted and the parties identified;
(4) This option just clears the output folder and exits.

- When the user selects a contract, the program adds a new option:
  (1) Pick a random contract;
  (2) Pick a random norm; and
  (4) Finish.

- Option (1) restarts the process with a new contract;
- Option (2) chooses a random norm among the extracted ones.

- When the user selects a norm, the program adds yet another option:
  (1) Pick a random contract;
  (2) Pick a random norm;
  (3) Make a conflict; and
  (4) Finish.

- Options (1,2,4) are the same as before;
- Option (3) displays the last chosen norm and asks you to alter it in order to create a conflict.

Process:

- In this manual conflict insertion, you are intended to follow a series of steps.
  1. Execute `contract_data_structure.py`;
  2. Insert your first name, and pick a contract with option (1);
  3. Choose a random norm using option (2);
  4. Choose the option to create a conflict (3).

- You have to create between 70 and 100 conflicts of 3 types. These types are:
  - Permission × Obligation (33%);
  - Permission × Prohibition (33%);
  - Obligation × Prohibition (33%).

- A regular norm has the following structure:

- **Example 1:**
  - "Purchaser must pay the product taxes."

[4] This code is available at `https://github.com/JoaoPauloAires/potential-conflict-identifier`

Given a regular norm, you will choose option 3, which allows you to alter such norm. Then you have to alter it in order to generate a conflict, e.g., if you got the Example 1, you may choose to create either a Permission × Obligation conflict or an Obligation x Prohibition conflict. In the first case (Permission × Obligation), a possible modification can be described as follows:

- **Example 2:**
  - "Purchaser MAY pay the product taxes."

To ensure that you are really making a conflict, use Table 7 as a guide:

Modal Verb	**Deontic Meaning**
can	Permission
may	Permission
must	Obligation
ought	Obligation
shall	Obligation
will	Obligation
modal_verb not	Prohibition

Table 7 Modal verbs and their deontic modalities

You may also modify the structure after the modal verb creating a conflict and altering the conflict structure (obviously maintaining the same meaning), as the Example 3 shows.

- **Example 3:**
  - "Purchaser may choose to pay the taxes related to the product."
- We recommend you to use more than three contracts to create conflicts, it allows us to test our approach in different contexts.
- At the end of the process, choose option (4) and that's it!

Thanks in advance.

B Performance measures used in the paper

Below, we summarize the performance measures from the literature used in this paper.

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{2}$$

$$precision = \frac{tp}{tp + fp} \tag{3}$$

$$recall = \frac{tp}{tp + fn} \tag{4}$$

$$f - score = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

$$tpr = \frac{tp}{tp + fn} \tag{6}$$

$$specificity = \frac{tn}{tn + fp} \tag{7}$$

$$fpr = \frac{fp}{fp + tn} \tag{8}$$