Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2018"

Moscow, May 30—June 2, 2018

DISCOVERING AND ASSESSING HEATED ARGUMENTS AT THE DISCOURSE LEVEL

Galitsky B. (boris.galitsky@oracle.com)[1,2],
Taylor R. (ray.taylor@oracle.com)[1]

[1]Oracle Inc., Redwood Shores CA, USA
[2]Higher School of Economics, Moscow, Russia

The problem of detecting heated arguments in text such as political debates and customer complaints is formulated as tree kernel learning of discourse structures. Affective argumentation structure is discovered in the form of discourse trees extended with edge labels for communicative actions. Extracted argumentation structures are then encoded as defeasible logic programs and are subject to dialectical analysis, to establish the validity of the main claim being communicated. We evaluate the accuracy of each step of this affect processing pipeline as well as overall performance.

1. Introduction

When an author attempts to provide a logical or affective argument in for something, a number of argumentation patterns can be employed. One of such patterns is to make a claim emotionally loaded, heated, escalated, associated with a confrontation. In text, argumentation patterns are associated with certain discourse features, and heated arguments are expressed in text by discourse means attempting to amplify the strength of these arguments.

The basic points of argumentation are reflected in rhetoric structure of text. A text without an argument, with a heated argument and with a logical one would have different rhetoric structures (Moens et al., 2007). When an author uses an affective argument instead of logical, it does not necessarily mean that his argument is invalid [Galitsky et al 2009). The goal of this study is to explore when a heated argumentation is valid. We introduce the notion of *heated argumentation* to circumscribe a special class of argumentation associated with strong emotions and sentiments.

Frequently people say of a politician's speech, *Oh, that's just rhetoric*, assuming that the words of politicians are empty verbiage or hot air. Frequently politicians do their most to sound impressive but indeed are saying nothing with real meaning.

Sometimes politicians are making promises his listeners believe he has no intention of keeping. The use of rhetoric in an intuitive sense in speeches: both bad, dishonest and good ones is only the most visible use of rhetoric. In this work we attempt to treat the intuitive notion of rhetoric computationally with a special focus on heated rhetoric. We expect the strongest, heated arguments to have a more prominent underlying rhetoric structure.

We select the Rhetoric Structure Theory (RST, [Mann and Thompson 1988]) as a means to represent discourse features associated with heated argumentation. Nowadays, a performance of both rhetoric parsers and argumentation reasoners has dramatically increased, and a discourse structure of text to be learned is formed from text automatically (Galitsky 2017a). Taking into account the discourse structure of conflicting dialogs, one can judge on the authenticity and validity of these dialogs in terms of validity of heated argumentation. In this work we will evaluate the *combined* argument validity assessment system that includes both the *discourse structure extraction* and *reasoning about it* with the purpose of validation of the complainant's claim.

Most of the modern techniques treat computational argumentation as specific discourse structures and perform detection of arguments of various sorts in text, such as classifying text paragraph as argumentative or non-argumentative ([Sardianos et al., 2015], [Stab and Gurevych, 2014], [Bondarenko, et al., 1997]). In this paper we intend to build the *whole heated argumentation pipeline*, augmenting argument extraction from text with its logical analysis. This pipeline is necessary to deploy an argumentation analysis in a practical decision support system:

1) Extract syntactic features;
2) Compute segmentation into elementary discourse units;
3) Build discourse trees;
4) Label their nodes with extracted communicative actions;
5) Form logical representation for clauses extracted from discourse tree;
6) Identify the main claim;
7) Given the logical representation as a Defeasible Logic Program, confirm or reject the main claim;
8) Produce a decision on whether argumentation for the main claim is acceptable or not.

Building this pipeline, we leverage two research areas: argument-mining, which is a linguistic-based, and logical validation of an argument, which is logic based. To the best of our knowledge, nowadays the former research area supports extracting various kinds of arguments from text on a scale, and the latter research area focuses on logical argumentation analysis of limited manually constructed argumentation structures. The contribution of this paper, the pipeline which implements the algorithms discovered in both these research areas, allows to perform logical analysis of a high quantity of heated arguments extracted from text. Therefore, industrial applications of mining and reasoning about heated arguments become possible. Since the paper combines linguistic and logical analyses, knowledge of both these domains is required from the reader to follow the whole pipeline of understanding heated arguments.

The concept of automatically identifying argumentation schemes was first discussed in (Walton et al., 2008). In (Ghosh et al., 2014) authors investigate argumentation discourse structure of the specific type of communication—online interaction threads. Identifying argumentation in text is connected to the problem of identifying truth, misinformation and disinformation on the web (Pendyala and Figueira, 2015, Galitsky 2015, Pisarevskaya et al 2015). In (Lawrence and Reed, 2015) three types of argument structure identification are combined: linguistic features, topic changes and machine learning.

To represent the linguistic features of text, we use the following sources:

1) *Rhetoric relations* between the parts of the sentences, obtained as a *discourse tree*.
2) *Speech acts, communicative actions*, obtained as verbs from the VerbNet resource (the verb signatures with instantiated semantic roles).

To assess the logical validity of extracted argument, we apply Defeasible Logic Program (DeLP, Garcia and Simari 2004), part of which is built on the fly from facts and clauses extracted from these sources. We integrate heated argumentation detection and validation components into a decision support system that can be deployed, for example, in a Customer Relationship Management (CRM) domain. To evaluate our approach to extraction and reasoning about argumentation, we choose the dispute resolution / customer complaint validation task because complainants frequently use heated argumengts to bring their point across. Most complainants are in a strong emotional distress due to a disparity between what they expected and what they received. Moreover, heated arguments appear in response to how companies communicate the issues with complainants. Most complaint authors report incompetence, flawed policies, ignorance, indifference to customer needs and misrepresentation from the customer service personnel. The complainants have frequently exhausted conventional communicative means available to them, confused, seeking recommendation from other users and advise others on avoiding particular financial service. Multiple affective argumentation patterns are used in complaints; the most frequent is an intense description by a complainant on a deviation of what has actually happened from what was expected, according to a common sense. This pattern covers both valid and invalid argumentation.

2. Representing Discourse for Heated Argumentation

We provide an example of conflicting agents providing their interpretation of certain events.

We show an example of a discourse tree (DT) for a heated argumentation of a customer treated badly by a credit card company American Express (amex) in 2007 (Fig. 1). Text split into logical chunks is as follows:

[I 'm another one of the many][that has been carelessly mistreated by American Express .] [I have had my card since 2004 and never late .] [In 2008][they reduced my credit limit from $16,600 to $6,000][citing several false excuses .] [Only one of their excuses was true—other credit card balances .] [They also increased my interest rate by 3 %][at the same time .] [I have never been so insulted by a credit card company .] [I used to have a credit score of 830 , not anymore , thanks to their unfair credit practices .] [They screwed my credit score .] [In these bad economic times you 'd think][they would appreciate consistent paying customers like us][but I guess][they are just so full of themselves .] [I just read today][that their CEO stated][that they will be hurt less than their competitors][because 80 percent of their revenues][are generated from fees.That][explains their callous , arrogant , unacceptable credit practices .] [It seems][they have to screw every cardholder][they can before the new law becomes effective .] [Well America , let 's learn from our appalling experience][and stop using our American Express credit card][so we can pay it off !].

We first explain how a traditional discourse tree encodes information flow in a paragraph of text. Text is split into logical chunks (elementary discourse units, EDUs) according to the order the entities of text are being introduced, attributes attached to them, and inter-relationships established. The author first introduces her opponent, describes how this opponent treats herself and others unfairly (according to her viewpoint) and enumerates different steps of this treatment.

DT is a tree where the EDUs are the labels of the terminal nodes. The other nodes are labeled with rhetorical relations encoding the type of logical links between the EDUs, such as *elaboration* (default), *attribution, cause* and others. Rhetorical relations hold not only between EDUs but also between the higher level logical chunks which might in turn include the lower level ones. That is how logical flow of text such as a heated argument can be visualized hierarchically. At the highest level, this text is split into two parts following the presentation sequence: 1) what happened; 2) what I think about it. This presentation style is covered by the rhetorical relation *topic-comment* (shown at the top of hierarchy).

In this study, to demonstrate the discourse features associated with heated argumentation, we augment the information on logical flow which is encoded in a traditional discourse tree and extend it by two components:

1) Communicative actions, showing how some elementary discourse units are being communicated [Galitsky 2017b];
2) Sentiment associated with some elementary discourse units.

It turns out that to differentiate a heated argumentation from a default, logical argumentation, 1) and 2) are essential. We refer to an extension of DT as a Communicative DT, CDT. CDT is a DT with labels for edges that are the VerbNet expressions for verbs (which are communicative actions, CA). Arguments of verbs are substituted from text according to VerbNet frames. The first and possibly second argument is instantiated by agents and the consecutive arguments—by noun or verb phrases which are the subjects of CA.

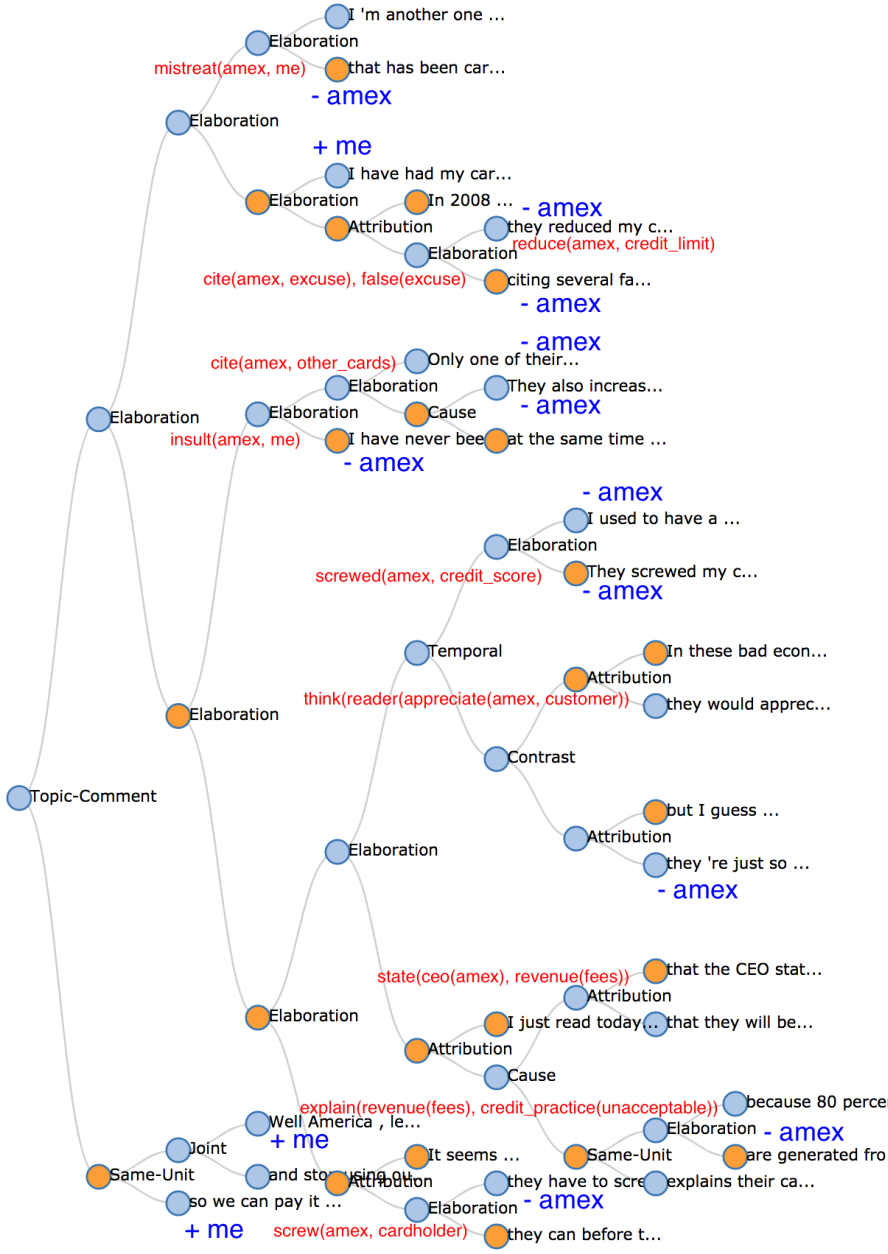

Fig. 1: A communicative discourse tree that includes labels for communicative actions and sentiments. Visualization of [Joty et al 2013] is used

In Fig. 1, the verbs communicative actions such as *mistreat(amex, me)* augment DT with necessary information about the text to match with other similar DTs. The

sequence of communicative actions provides information on the structure of a dialogue between a proponent and an opponent of a given argument. This information is complementary to what DT encodes for logical chunks provided irrespectively of how the entities from these chunks were communicated. Communicative actions are labels of the edges of the DT leading to the terminal nodes; sentiments are labels of these edges as well. We denote a sentiment polarity as + or—and the subject of this sentiment as the proponent (*me*) and opponent (here, *amex*). Naturally, an author provides an argument for how she is right and hew opponent is wrong, therefore one expects the positive sentiments with the author's EDUs and the negative ones with her opponents' EDUs. For each label, it is attached to the CDT edge nearest to the right end of the label expression.

Argumentation analysis needs a systematic approach to learn associated discourse structures. The features of CDTs could be represented in a numerical space argumentation detection can be conducted; however structural information on DTs would not be leveraged. Also, features of argumentation can potentially be measured in terms of maximal common sub-DTs, but such nearest neighbor learning is computationally intensive and too sensitive to errors in DT construction [Galitsky et al., 2015].

Therefore a CDT-kernel learning approach is selected which applies SVM learning ([Joty and Moschitti 2014], [Wang et al., 2010]) to the feature space of all sub-CDTs of the CDT for a given text where a heated argument is being detected.

We combined Stanford NLP parsing, coreferences, entity extraction, DT construction (discourse parser, [Surdeanu et al., 2016] and [Joty et al., 2013]), VerbNet and Tree Kernel builder into one system available at https://github.com/bgalitsky/relevance-based-on-parse-trees.

Our second example is a regular discourse tree for a text advising on how to behave communicating a heated argument [Inspiyr 2018].

> When you are in the middle of an argument, it can be easy to get caught up in the heat of the moment and say something that makes the situation even worse. Nothing can make someone more frenzied and hysterical than telling them to calm down. It causes the other person to feel as if you are putting the blame for the elevation of the situation on them. Rather than actually helping them calm down, it comes off as patronizing and will most likely make them even angrier.

A default DT for text such as a work of fiction or a scientific article, introducing and explaining a subject, would have default rhetoric relation of elaboration (as well as joint, attribution, background). This DT in addition has topic-comment on the top level, and also enablement, which indicates a peculiar logic flow. We will apply machine learning approach with extensive dataset of DT examples to observe a specific features of DTs associated with heated argumentation. In our earlier studies [Galitsky et al 2018] we developed a technique to extract and learn logical argumentation from text, and now we will apply it to heated arguments.

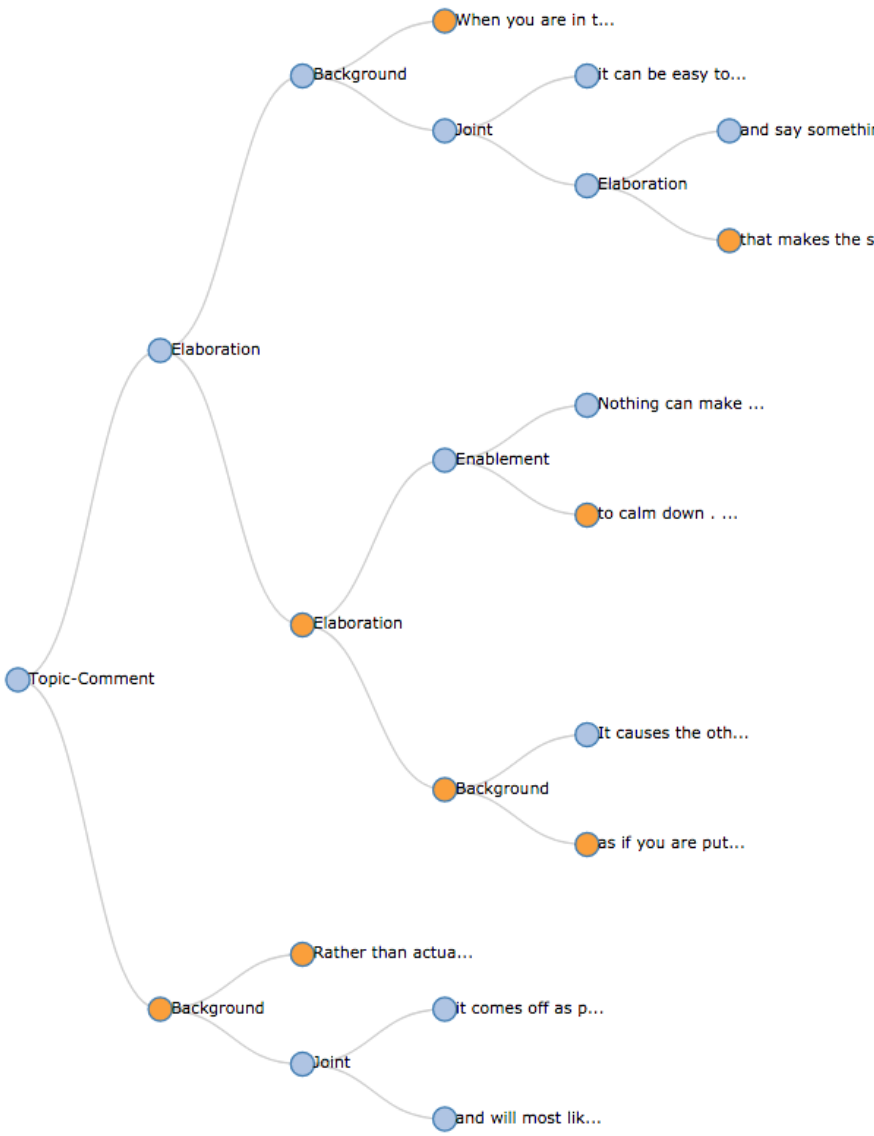

Fig. 2: The DT for a text advising on how to behave communicating an argument

3. Assessing Validity of Extracted Argument Patterns via Dialectical Analysis

To convince an addressee, a message needs to include an argument and its structure needs to be valid. Once an argumentation structure extracted from text is represented via CDT, we need to verify that the main point (target claim) communicated by the author is not logically attacked by her other claims. For a given domain, this claim is known (such as innocent or guilty, winning or loosing case, complaint is valid or not, violation has occurred or not). Most facts and clauses are pre-specified in a vertical domain ontology (the static part) and some of them are extracted from text via CDT (those can be less reliable).

To assess the validity of the argumentation, Defeasible Logic Programming (DeLP) approach is selected, an argumentative framework based on logic programming ([García and Simari 2004], [Alsinet et al 2008]), and present an overview of the main concepts associated with it.

A DeLP is a set of facts, strict rules P of the form (A:-B), and a set of defeasible rules D of the form A-<B, whose intended meaning is "if B is the case, then usually A is also the case". Let P=(P, D) be a DeLP program and L a ground literal.

Defeasible Rules Prepared In Advance
rent_receipt -< rent_deposit_transaction.
rent_deposit_transaction -< contact_tenant.
¬rent_deposit_transaction -<contact_tenant, three_days_notice_is_issued.
¬rent_deposit_transaction -< rent_is_overdue.
¬repair_is_done -< rent_refused, repair_is_done. repair_is_done -<
rent_is_requested.
¬rent_deposit_transaction -< tenant_short_on_money, repair_is_done.
¬repair_is_done -< repair_is_requested.
¬repair_is_done -<rent_is_requested.
¬repair_is_requested -< stay_unrepaired.
¬repair_is_done -< stay_unrepaired.
Target Claim to be Assessed
?—rent_receipt
Clauses Extracted from text
repair_is_done -< rent_refused.
Facts from text
contact_tenant. rent_is_requested. rent_refused. remind_about_repair.
three_days_notice_is_issued.
rent_is_overdue. stay_unrepaired.

Fig. 3a: An example of a Defeasible Logic Program for heated argument extracted from text

Let us now build an example of a DeLP for legal reasoning about facts extracted from text (Fig. 3). A judge hears an eviction case and wants to make a judgment on whether rent was provably paid (deposited) or not (denoted as *rent_receipt).*

An input is a text where a defendant is expressing his point. Underlined words form the clause in DeLP, and the other expressions formed the facts (Fig. 3b).

> *The landlord contacted me, the tenant, and the rent was requested. However, I refused the rent since I demanded repair to be done. I reminded the landlord about necessary repairs, but the landlord issued the tree-day notice confirming that the rent was overdue. Regretfully, the property still stayed unrepaired.*

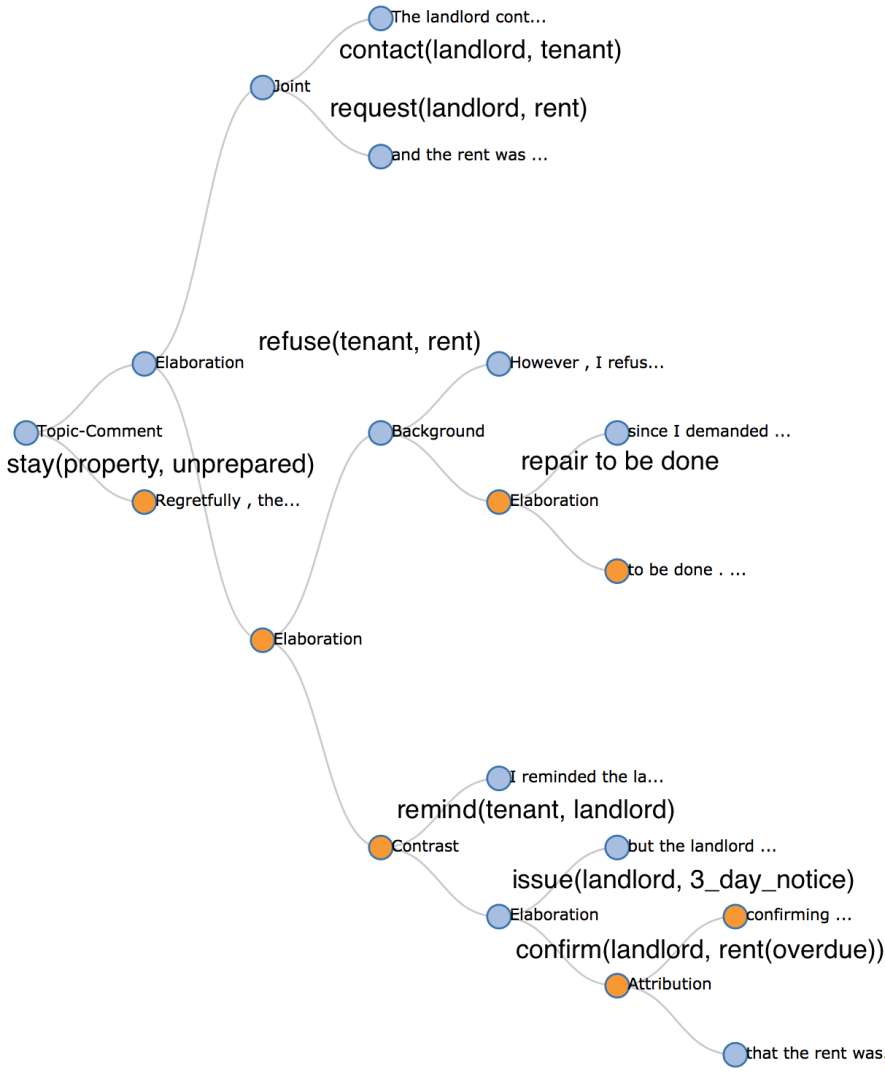

Fig 3b: Text of a complaint and its CDT

A *defeasible derivation* of L from P consists of a finite sequence $L_1, L_2, \ldots, L_n = L$ of ground literals, such that each literal L_i is in the sequence because:

(a) L_i is a fact in Π, or
(b) there exists a rule R_i in P (strict or defeasible) with head L_i and body $B_1, B_2, \ldots, B_k$ and every literal of the body is an element L_j of the sequence appearing before L_j ($j < i$).

Let h be a literal, and P=(Π, Δ) a DeLP program. We say that <A, h> is an *argument* for h, if A is a set of defeasible rules of Δ, such that:

1. there exists a defeasible derivation for h from =(Π ∪ A);
2. the set (Π ∪ A) is non-contradictory; and
3. A is minimal: there is no proper subset A_0 of A such that A_0 satisfies conditions (1) and (2).

Hence an argument <A, h> is a minimal non-contradictory set of defeasible rules, obtained from a defeasible derivation for a given literal h associated with a program P.

We say that $<A_1, h_1>$ *attacks* $<A_2, h_2>$ iff there exists a sub-argument <A, h> of $<A_2, h_2>$ ($A \subseteq A_1$) such that h and h_1 are inconsistent (i.e. $\Pi \cup \{h, h_1\}$ derives complementary literals). We will say that $<A_1, h_1>$ *defeats* $<A_2, h_2>$ if $<A_1, h_1>$ attacks $<A_2, h_2>$ at a sub-argument <A, h> and $<A_1, h_1>$ is strictly preferred (or not comparable to) <A, h>. In the first case we will refer to $<A_1, h_1>$ as a *proper defeater*, whereas in the second case it will be a *blocking defeater*. Defeaters are arguments which can be in their turn attacked by other arguments, as is the case in a human dialogue. An *argumentation line* is a sequence of arguments where each element in a sequence defeats its predecessor. In the case of DeLP, there are a number of *acceptability* requirements for argumentation lines in order to avoid fallacies (such as circular reasoning by repeating the same argument twice).

Target claims can be considered DeLP queries which are solved in terms of dialectical trees, which subsumes all possible argumentation lines for a given query. The definition of dialectical tree provides us with an algorithmic view for discovering implicit self-attack relations in users' claims. Let $<A_0, h_0>$ be an argument (target claim) from a program P. A *dialectical tree* for $<A_0, h_0>$ is defined as follows:

1. The root of the tree is labeled with $<A_0, h_0>$
2. Let N be a non-root vertex of the tree labeled $<A_n, h_n>$ and $\Lambda = [<A_0, h_0>, <A_1, h_1>, \ldots, <A_n, h_n>]$ (the sequence of labels of the path from the root to N). Let $[<B_0, q_0>, <B_1, q_1>, \ldots, <B_k, q_k>]$ all attack $<A_n, h_n>$. For each attacker $<B_i, q_i>$ with acceptable argumentation line $[\Lambda, <B_i, q_i>]$, we have an arc between N and its *child* N_i.

A labeling on the dialectical tree can be then performed as follows:

1. All leaves are to be labeled as U-nodes (undefeated nodes).
2. Any inner node is to be labeled as U-node whenever all its associated children nodes are labeled as D-nodes.
3. Any inner node is to be labeled as D-node whenever at least one of its associated children nodes is labeled as U-node.

After performing this labeling, if the root node of the tree is labeled as a U-node, the original argument at issue (and its conclusion) can be assumed as *justified* or *warranted*.

In our DeLP example, the literal *rent_receipt* is supported by

<*A, rent_receipt*> = <{ (*rent_receipt -< rent_deposit_transaction*), (*rent_deposit_transaction -< tenant_short_on_money*)}, *rent_receipt*> and there exist three defeaters for it with three respective argumentation lines: <B_1, ┐*rent_deposit_transaction*> = <{(┐*rent_deposit_transaction -< tenant_short_on_money, three_days_notice_is_issued*)}, *rent_deposit_transaction*>.

<B_2,┐*rent_deposit_transaction*> = <{(┐ *rent_deposit_transaction -< tenant_short_on_money, repair_is_done*), (*repair_is_done -< rent_refused*)}, *rent_deposit_transaction*>.

<B_3, ┐*rent_deposit_transaction*> = <{(┐*rent_deposit_transaction -< rent_is_overdue*)}, *rent_deposit_transaction*>.

The first two are proper defeaters and the last one is a blocking defeater. Observe that the first argument structure has the counter-argument, <{*rent_deposit_transaction -< tenant_short_on_money*}, *rent_deposit_transaction)*, but it is not a defeater because the former is more specific. Thus, no defeaters exist and the argumentation line ends there. B_3 above has a blocking defeater

<{(*rent_deposit_transaction -< tenant_short_on_money*)}, *rent_deposit_transaction*>

which is a disagreement sub-argument of <*A, rent_receipt*> and it cannot be introduced since it gives rise to an unacceptable argumentation line. B_2 has two defeaters which can be introduced:

<C_1, ┐*repair_is_done* >, *where* C_1 = {(┐*repair_is_done -< rent_refused, repair_is_done*), (*repair_is_done -< rent_is_requsted*)}, a proper defeater, and <C_2, ┐*repair_is_done* >,

where C_2={(┐*repair_is_done -< repair_is_requested*)} is a blocking defeater. Hence one of these lines is further split into two; C_1 has a blocking defeater that can be introduced in the line

<D_1, ┐*repair_is_done* >, *where* D_1= <{(┐*repair_is_done -< stay_unrepaired*)}.

D_1 and C_2 have a blocking defeater, but they cannot be introduced, because they make the argumentation line inacceptable. Hence the state *rent_receipt* cannot be reached, as the argument supporting the literal *rent_receipt* is not warranted. The dialectical tree for *A* is shown in Fig. 4.

Having shown how to build dialectic tree, we now ready to outline the algorithm for validation the domain-specific claim for arguments extracted from text:

1. Build a DT from input text;
2. Attach communicative actions to its edges to form CDT;
3. Extract subjects of communicative actions attached to CDT and add to 'Facts' section;

4. Extract the arguments for rhetoric relation *contrast* and communicative actions of the class *disagree* and add to 'Clauses Extracted FromText' section;
5. Add domain-specific section to DeLP;
6. Having the DeLP formed, build a dialectical tree and assess the claim.

We used [Tweety 2017] system for DeLP implementation. The Tweety package contains several classes for dealing with abstract argumentation frameworks which can be imported programmatically using specific methods. Tweety supports reasoning relying on the extension-based approaches of grounded, stable, complete, preferred, ideal, semistable, CF2, and stage semantics as well as the ranking-based approaches of [Grossi & Modgil, 2015].

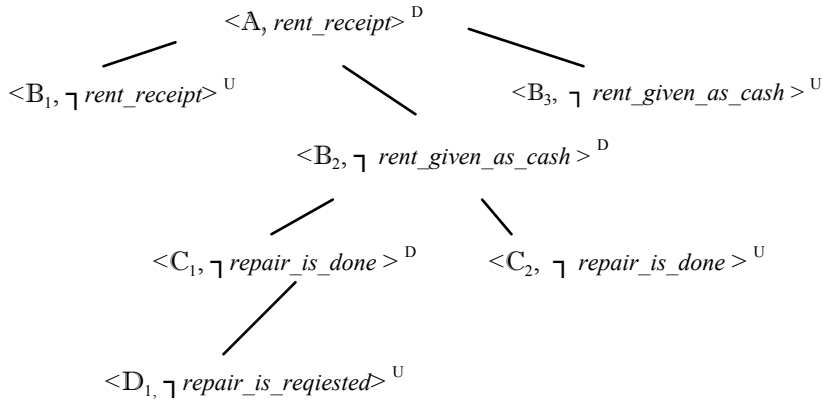

Fig. 4: Dialectical tree for target claim rent_receipt

4. Evaluation of Detection and Validation of Affective Arguments

The objective of argument detection task is to identify all kinds of arguments, not only ones associated with customer complaints. We formed the *positive* dataset from textual customer complaints dataset (Galitsky et al., 2009, and https://github.com/bgalitsky/relevance-based-on-parse-trees/blob/master/src/test/resources/opinions-FinanceTagged.xls.zip, scraped from consumer advocacy site PlanetFeedback.com. This dataset is used for both argument detection and argument validity tasks.

Table 1: Evaluation results for argument detection

Method / sources	P	R	F1
Bag-of-words	57.2	53.1	55.07
WEKA-Naïve Bayes	59.4	55.0	57.12
SVM TK for RST and CA (full parse trees)	77.2	74.4	75.77
SVM TK for DT	63.6	62.8	63.20
SVM TK for CDT	82.4	77.0	79.61

For the *negative* dataset, only for the affective argument detection task, we used Wikipedia, factual news sources, and also the component of [Lee, 2001] dataset that includes such sections of the corpus as: ['tells'], instructions for how to use software; ['tele'], instructions for how to use hardware", and [news], a presentation of a news article in an objective, independent manner, and others. Further details on the data set are available in [Galitsky et al 2015].

A baseline approach relies on keywords and syntactic features to detect argumentation (Table 1). Frequently, a coordinated pair of communicative actions (so that at least one has a negative sentiment polarity related to an opponent) is a hint that logical argumentation is present. This naïve approach is outperformed by the top performing TK learning CDT approach by 29%. SVM TK of CDT outperforms SVM TK for RST+CA and RST + full parse trees [Galitsky 2017] by about 5% due to noisy syntactic data which is frequently redundant for argumentation detection.

SVM TK approach provides acceptable F-measure but does not help to explain how exactly the affective argument identification problem is solved, providing only final scoring and class labels. Nearest neighbor maximal common sub-graph algorithm is much more fruitful in this respect [Galitsky et al 2015]. Comparing the bottom two rows, we observe that it is possible, but infrequent to express an affective argument without CAs.

Assessing logical arguments extracted from text, we were interested in cases where an author provides invalid, inconsistent, self-contradicting cases. That is important for CRM systems focused on customer retention and facilitating communication with customer [Galitsky et al 2009]. The domain of residential real estate complains was selected and DeLP thesaurus was built for this domain. Automated complaint processing system is essential, for example, for property management companies in their decision support procedures [Constantinos et al 2003].

Table 2: Evaluation results for argument validation

Types of complaints	P	R	F1 of validation	F1 of total
Single rhetoric relation of type *contrast*	87.3	15.6	26.5	18.7
Single communicative action of type *disagree*	85.2	18.4	30.3	24.8
Two or three specific relations or communicative actions	80.2	20.6	32.8	25.4
Four and above specific relations or communicative actions	86.3	16.5	27.7	21.7

In our validity assessment we focus on target features related to how a given complaint needs to be handled, such as *compensation_required, proceed_with_eviction, rent_receipt* and others.

Complaint validity assessment results are shown in Table 2. In the first and second rows, we show the results of the simplest complaint with a single rhetoric relation such as *contrast* and a single CA indicating an extracted argumentation attack

relation respectively. In the third row we assess complaints of average complexity, and in the bottom row—most complex, longer complaints in terms of their CDT. The third column shows detection accuracy for invalid argumentation in complaints in a standalone argument validation system. Finally, the fourth column shows the accuracy of the integrated argumentation extraction and validation system.

For decision support systems, it is important to maintain a low false positive rate. It is acceptable to miss invalid complaints, but for a detected invalid complain, confidence should be rather high. If a human agent is recommended to look at a given complaint as invalid, her expectations should be met most of the times. Although F1 measure of the overall argument detection and validation system is low in comparison with modern recognition systems, it is still believed to be usable as a component of a CRM decision support system.

5. Conclusions

We observed that relying on discourse tree data, one can reliably detect patterns of affective argumentation. Communicative discourse trees then become a source of information to form a defeasible logic program to validate an argumentation structure. Although the performance of the former being about 80% is significantly above that of the latter (29%), the overall pipeline can be useful for detecting cases of invalid heated argumentation, which are important in decision support for CRM. Once it is possible to extract amplified, heated arguments, in our future studies we will proceed to combining mining and reasoning about general arguments, not necessarily accented by a sentiment.

We anticipate the difficulties in adopting the argumentation pipeline in industry. Today, sentiment analysis is extensively used by companies to understand which features of products and services are appreciated by customers and which need improvement. Deeper understanding of customer complaints, implemented in this study, would reveal shady corporate practices and would put a blame on certain company management individuals responsible for respective product limitations and customer support deficiencies. Internal corporate policies and internal conflicts of interest between management structures could potentially be affected by findings produced by the argumentation pipeline, and a significant number of corporate management members might oppose obtaining these findings. Hence a series of issues outside of the technology area might prevent argumentation pipeline from being deployed in industry. [Galitsky 2016] addressed the corporate conflict of interest models from the standpoint of multiagent systems; the results show that a corporate multiagent system can involve into behavioral forms distant from being rational or competent.

In this paper we attempted to combine the best of both worlds, argumentation mining from text and reasoning about the extracted argument. Whereas applications of either technology are limited, the whole argumentation pipeline is expected to find a broad range of applications. In this work we focused on a very specific legal area such as customer complaints, but it is easy to see a decision support system employing the proposed argumentation pipeline in other domains of CRM.

An important finding of this study is that argumentation structure can be discovered via the features of extended discourse representation, combining information on how an author organizes her thoughts with information on how involved agents communicate these thoughts. Once a communicative discourse tree is formed and identified as being correlated to argumentation, a defeasible logic program can be built from this tree and the dialectical analysis can validate the main claim.

References

1. *Mann, William and Sandra Thompson* (1988). Rhetorical structure theory: Towards a functional theory of text organization. Text-Interdisciplinary Journal for the Study of Discourse, 8(3):243–281.
2. *Alsinet, T., Carlos Iván Chesñevar, Lluis Godo, Guillermo Ricardo Simari* (2008). A logic programming framework for possibilistic argumentation: Formalization and logical properties. Fuzzy Sets and Systems 159(10): 1208–1228
3. *Mikolov, Tomas, Chen, Kai, Corrado; G.S., Dean; Jeffrey* (2015). Computing numeric representations of words in a high-dimensional space. US Patent 9,037,464, Google, Inc.
4. *Wang, W., Su, J., Tan, C. L.* (2010). Kernel Based Discourse Relation Recognition with Temporal Ordering Information. In *ACL*.
5. *Wei Feng, Vanessa and Graeme Hirst* (2014). A linear-time bottom-up discourse parser with constraints and post-editing. In ACL.
6. *Joty, Shafiq R, Giuseppe Carenini, Raymond T Ng, and Yashar Mehdad* (2013). Combining intra- and multi-sentential rhetorical parsing for document-level discourse analysis. In *ACL (1)*, pages 486–496.
7. *Joty, Shafiq R and A. Moschitti* (2014). Discriminative Reranking of Discourse Parses Using Tree Kernels. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).
8. *Kipper, K. Korhonen, A., Ryant, N. and Palmer, M.* (2008). A large-scale classification of English verbs. Language Resources and Evaluation Journal, 42, pp. 21–40.
9. *Surdeanu, Mihai, Thomas Hicks, and Marco A. Valenzuela-Escarcega* (2015) Two Practical Rhetorical Structure Theory Parsers. Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics—Human Language Technologies: Software Demonstrations (NAACL HLT).
10. *Galitsky, B., MP González, CI Chesñevar* (2009). A novel approach for classifying customer complaints through graphs similarities in argumentative dialogue. Decision Support Systems, 46–3, 717–729.
11. *Galitsky, B.* (2013). Machine learning of syntactic parse trees for search and classification of text. *Engineering Application of AI* , 26(3) 1072–91.
12. *Galitsky, B.* (2015). Detecting Rumor and Disinformation by Web Mining. AAAI Spring Symposium, 2015.
13. *Galitsky, B., Ilvovsky, D. and Kuznetsov S. O.* (2015). Rhetoric Map of an Answer to Compound Queries. ACL-2, 681–686.
14. *Galitsky, B.* (2016). Computational Autism. Springer, London UK.

15. *Galitsky, B.* (2017a). Matching parse thickets for open domain question answering, Data & Knowledge Engineering, Volume 107, January 2017, Pages 24–50.
16. *Galitsky, B.* (2017b). Using Extended Tree Kernel to Recognize Metalanguage in Text. In Uncertainty Modeling, Volume 683 of the series Studies in Computational Intelligence pp. 71–96, Springer.
17. *Constantinos J. Stefanou, Christos Sarmaniotis, Amalia Stafyla.* (2003). CRM and customer-centric knowledge management: an empirical research, Business Process Management Journal, Vol. 9 Issue: 5, pp.617–634.
18. *Tweety* (2016). Last downloaded Dec 12, 2016. https://javalibs.com/artifact/net.sf.tweety.arg/delp.
19. *Thimm, M.* (2014) Tweety—A Comprehensive Collection of Java Libraries for Logical Aspects of Artificial Intelligence and Knowledge Representation. In Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14). Vienna, July, 2014
20. *Garcia, Alejandro and Guillermo R. Simari.* (2004) Defeasible Logic Programming: An Argumentative Approach. Theory and Practice of Logic Programming 4(1–2):95–138, 2004.
21. *Amgoud, Leila, Philippe Besnard, Anthony Hunter.* (2015). Representing and Reasoning About Arguments Mined from Texts and Dialogues. ECSQARU 2015: 60–71.
22. *Bondarenko, A., Dung, P., Kowalski, R., Toni, F.* (1997) An abstract, argumentation-theoretic approach to default reasoning. Artificial Intelligence 93, 63–101.
23. *Inspiyr* (2018). 5 Things You Should Never Say During A Heated Argument. https://inspiyr.com/heated-argument/.
24. *Vishnu S. Pendyala, Silvia Figueira* (2015) Towards a truthful world wide web from a humanitarian perspective. Global Humanitarian Technology Conference (GHTC), 2015 IEEE, Issue Date: 8–11 Oct. 2015.
25. *Grossi, D., Modgil, S.* (2015) On the graded acceptability of arguments. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15). pp. 868–874.
26. *Toni, F.* (2014). A tutorial on assumption-based argumentation. Argument & Computation 5(1), 89–117.
27. *Moens, Marie-Francine, Erik Boiy, Raquel Mochales Palau, and Chris Reed* (2007). Automatic detection of arguments in legal texts. In Proceedings of the 11th International Conference on Artificial Intelligence and Law, ICAIL '07, pages 225–230, Stanford, CA, USA.
28. *Christos Sardianos, Ioannis Manousos Katakis, Georgios Petasis, and Vangelis Karkaletsis* (2015). Argument extraction from news. In Proceedings of the 2nd Workshop on Argumentation Mining, pages 56–66, Denver, CO, USA.
29. *Stab, C. and Gurevych, I.* (2016) Recognizing the absence of opposing arguments in persuasive essays. ACL 2016.
30. *Walton, D., Reed, C., Macagno, F.* (2008) Argumentation Schemes. Cambridge University Press. .
31. *Pisarevskaya, Dina, Tatiana Litvinova, Olga Litvinova* (2017) Deception Detection for the Russian Language: Lexical and Syntactic Parameters. Proceedings of the 1st Workshop on Natural Language Processing and Information Retrieval / RANLP.