Fuzzy ILP Classification of Web Reports after Linguistic Text Mining

Jan Dědek[a], Peter Vojtáš[a], Marta Vomlelová[b]

[a] *Department of Software Engineering, Charles University, Prague, Czech Republic*
[b] *Department of Theoretical Computer Science and Mathematical Logic, Charles University, Prague, Czech Republic*

Abstract

In this paper we study the problem of classification of textual web reports. We are specifically focused on the issue when structured information extracted from the reports is used for the classification. We present an experimental classification system, which is based on usage of third party linguistic analyzers, our previous work on web information extraction and fuzzy inductive logic programming (fuzzy ILP). A detailed study of so called 'Fuzzy ILP Classifier' is the main contribution of the paper. The study includes formal models, prototype implementation, extensive evaluation experiments and comparison of the classier with other alternatives like decision trees, support vector machines, neural networks, etc.

Keywords: Fuzzy, Inductive Logic Programming, Information Extraction, Natural Language Processing, Machine Learning

1. Introduction

Big amount of information on the web increases the need of automated processing. Especially machine processing and machine understanding of textual information is very difficult. In this paper we study the problem of classification of textual web reports. We are specifically focused on the issue when structured information extracted from the reports is used for the classification. We present an experimental classification system, which is

Email addresses: `dedek@ksi.mff.cuni.cz` (Jan Dědek), `vojtas@ksi.mff.cuni.cz` (Peter Vojtáš), `marta@ktiml.mff.cuni.cz` (Marta Vomlelová)

based on usage of third party linguistic analyzers, our previous work on web information extraction and fuzzy inductive logic programming (fuzzy ILP).

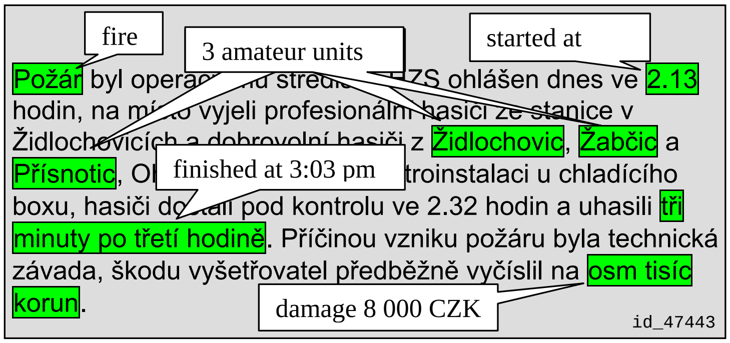

Figure 1: Example of analyzed web report.

The paper is based on a case study of seriousness classification of accident reports. Fig. 1 presents such an example of a message of an accident report from the web. We would like to have a tool, which is able to classify such message by a degree of being it a serious accident.

Our solution is based on information extraction (see emphasized pieces of information that could be extracted from a report in Fig. 1) and on a fuzzy-based machine learning procedure that provides rules for classification of the reports.

Our experiments deal with texts in the Czech language but our method is general and it can be used with any structured linguistic representation. In this paper we do not provide many details about the information extraction part of the solution. We concentrate on the classification part and present a detailed study of the fuzzy ILP classification method (so called 'Fuzzy ILP Classifier').

In our application we are facing the challenge of induction and/or mining on several places. First we need an inductive procedure when extracting attributes of an accident from web reports texts. Second (the subject of this paper) we need an inductive procedure when trying to explain the degree of seriousness of an accident by the attributes of the accident. On both places ILP is used as the inductive procedure.

- During the information extraction phase we exploit the fact that ILP can work directly with multirelational data e.g. deep syntactic (tectogrammatical) trees built form sentences of processed text.

- During the classification phase we are experimentally using the Fuzzy ILP Classifier, because it performs quite well on our dataset (see in Section 6.1) and it is naturally capable to handle fuzzy data, which occurs when the information extraction engine returns also confidence probability values along with the extracted data. But in this paper we do not describe this situation and only the approach is fuzzy in the present demonstration.

The main contributions of this paper are formal models, prototype implementation and experimental evaluation of the Fuzzy ILP Classifier.

The paper is organized as follows: In Section 2 we introduce some closely related works. In Section 3 design of the experimental system is presented, including short description of the information extraction method and linguistic analyzers. Section 4 provides details about our case study, which is later used for illustration. Section 5 presents formal models of the system (including ILP, fuzzy ILP and several translations of a fuzzy ILP task to classical ILP) followed by a description of implementation of the models in the system. In Section 6 we present main results of the work, we evaluate and compare the methods with other well-known classifiers. Section 7 concludes the paper.

2. Related Work

There are plenty of systems dealing with text mining and text classification. In (Reformat et al., 2008) the authors use ontology modeling to enhance text identification. The authors of (Chong et al., 2005) use preprocessed data from National Automotive Sampling System and test various soft computing methods to modeling severity of injuries (some hybrid methods showed best performance). Methods of Information Retrieval (IR) are very numerous, with extraction mainly based on key word search and similarities. The Connection of IR and text mining techniques with web information retrieval can be found in Chapter Opinion mining in the book of Bing Liu (Liu, 2007).

The Fuzzy ILP Classifier can be seen as an ordinary classifier for data with the monotonicity constraint (the target class attribute has to be monotonizable – a natural ordering has to exist for the target class attribute). There are several other approaches addressing the classification problems with the monotonicity constraint.

The CART-like algorithm for decision tree construction does not guarantee to provide a monotone tree even on a monotone dataset. The algorithm can be modified (Bioch and Popova, 2009) to provide a monotone tree on the dataset by adding the corner elements of a node with an appropriate class label to the existing data whenever necessary.

An interesting approach is presented in (Kotlowski and Slowinski, 2009) first, the dataset is "corrected" to be monotone (a minimal number of target class labels is changed to get a monotone dataset), then a learning algorithm is applied – linear programming boosting in the cited paper.

Several other approaches to monotone classification have been presented – instance based learning (Lievens et al., 2008), rough sets (Bioch and Popova, 2000) and others.

3. Design of the system

A general schema of our experimental system is shown on Fig. 2. We use our previously developed web information extraction tools based on third party linguistic analyzers (the upper two dashed arrows; this is not a subject of this paper). We extract some structured information from the web and the extracted information is then translated to an ILP knowledge base and along with a user rating it is used for the classification. (We assume that a small amount of learning data is annotated by a human user.) The classification is based on ILP and it could be *fuzzy* or *crisp* (crisp denotes a strait forward application of ILP and fuzzy is for the fuzzy method, see in next sections.)

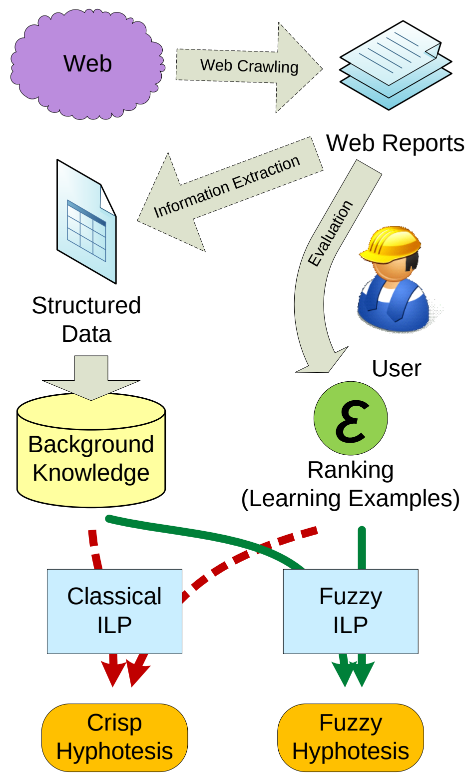

Figure 2: Schema of our system.

3.1. Linguistic Analysis

In this section we will briefly describe the linguistic tools that we have used to produce linguistic annotations of texts.

These tools are being developed in the Institute of Formal and Applied Linguistics in Prague, Czech Republic. They are publicly available – they have been published on a CDROM under the title PDT 2.0 (Hajič et al. (2006) – first five tools) and in (Klimeš (2006) – Tectogrammatical analysis). These tools are used as a processing chain. At the end of the chain they produce tectogrammatical (Mikulová et al., 2006) dependency trees built up from the text.

Tool 1. Segmentation and tokenization consists of tokenization (dividing the input text into words and punctuation) and segmentation (dividing a sequences of tokens into sentences).

Tool 2. Morphological analysis assigns all possible lemmas and morphological tags to particular word forms (word occurrences) in the text.

Tool 3. Morphological tagging consists in selecting a single pair lemma-tag from all possible alternatives assigned by the morphological analyzer.

Tool 4. Collins' parser – Czech adaptation. Unlike the usual approaches to the description of English syntax, the Czech syntactic descriptions are dependency-based, which means, that every edge of a syntactic tree captures the relation of dependency between a governor and its dependent node. Collins' parser gives the most probable parse of a given input sentence.

Tool 5. Analytical function assignment assigns a description (analytical function – in linguistic sense) to every edge in the syntactic (dependency) tree.

Tool 6. Tectogrammatical analysis produces linguistic annotation at the tectogrammatical level, sometimes called "layer of deep syntax". An example of a tectogrammatical tree can be seen on the Fig. 3. Annotation of a sentence at this layer is closer to the meaning of the sentence than its syntactic annotation and thus information captured at the tectogrammatical layer is crucial for machine understanding of natural language (Klimeš, 2006).

3.2. Web Information Extraction

After the content of a web resource is analyzed by the above mentioned linguistic tools, the output linguistic data is stored in the form of tectogrammatical trees. To achieve our objectives we have to extract information from

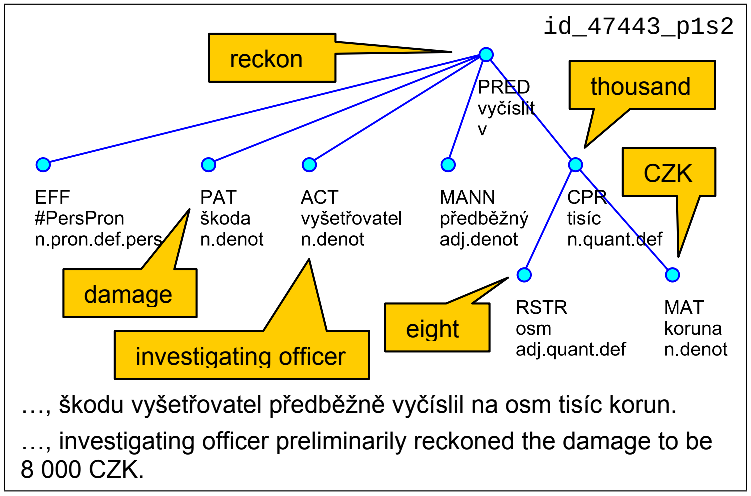

Figure 3: Example of a linguistic tree of one of analyzed sentences.

this representation. Here we refer to our previous work (Dědek and Vojtáš, 2008; Dědek et al., 2008). A long path of tools starting with web crawling and resulting with the extracted structured information has been developed in our previous works. In the Fig. 3 nodes of the tectogrammatical tree are decorated. A piece of information about the damage of 8000 CZK can be found there (the three nodes on the right). We have used ILP to learn rules, which are able to detect these nodes. The extraction process requires a human assistance when annotating the training data.

Note that our method is general and is not limited to Czech and can be used with any structured linguistic representation.

4. The case study – accident seriousness classification

The main experiment presented in this paper leads to the seriousness classification of an accident presented on a web report, which is one of possible utilizations of the extracted information. We use web reports of fire departments of several regions of the Czech Republic. These reports are written Czech and can be accessed through the web of the General Directorate of the Fire and Rescue Service of the Czech Republic[1].

[1] http://www.hzscr.cz

attribute name	distinct values	missing values	monotonic
size (of file)	49	0	yes
type (of accident)	3	0	no
damage	18	30	yes
dur_minutes	30	17	yes
fatalities	4	0	yes
injuries	5	0	yes
cars	5	0	yes
amateur_units	7	1	yes
profesional_units	6	1	yes
pipes	7	8	yes
lather	3	2	yes
aqualung	3	3	yes
fan	3	2	yes
ranking	14	0	yes

Table 1: Accident attributes.

For our experiment we have selected a collection of 50 web reports. We have identified several features presented in these reports and manually extracted corresponding values. This will be described in more detail in the section 4.1. To each report we have also assigned a value of overall ranking of seriousness of the presented accident, which is the target of the classification. The whole dataset can be downloaded from our Fuzzy ILP classifier's web page[2].

In this experiment we have not used an information extracted by our automated information extraction tools. In this paper we are concentrated on the classification and the actual source of the information is not so important. The integration step is still waiting to be done.

4.1. Features of accidents

The table 1 summarizes all features (or attributes) that we have obtained from accident reports. Except the attribute `type` (type of an accident – `fire`, `car_accident` and `other`) all the attributes are numerical and so monotonizable. There are cases when the value of an attribute is unknown. We have decided to make evidence of this and keep the values `unknown` in our knowledge base. A brief explanation of each attribute follows.

- `size` is the file size of the text of the particular report.

[2] `http://www.ksi.mff.cuni.cz/~dedek/fuzzyILP/`

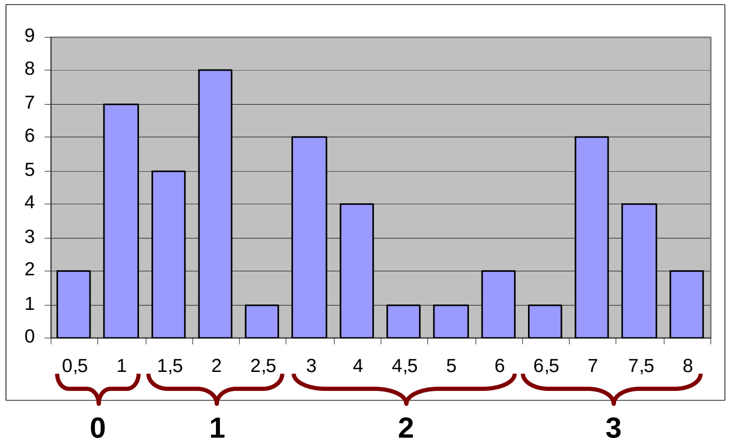

Figure 4: Frequencies of the seriousness ranking.

- `damage` is the amount (in CZK – Czech Crowns) of the summarized damage arisen during the reported accident.
- `dur_minutes` is the time taken to handle the accident.
- `fatalities` and `injuries` are the numbers of fatalities and injuries taken by the accident.
- `cars` is the number of cars damaged during the accident (important during car accidents).
- `professional_units` and `amateur_units` are the numbers of the fireman units sent for the accident.
- `pipes` is the number of used fire pipes.
- `lather`, `aqualung` and `fan` (ventilator) indicates whether these devices were used.

4.2. Seriousness ranking

Values of the overall seriousness ranking attribute were stated from "a general impression" from texts of the reports texts with respect to particular attributes. The values have evolved to 14 distinct values in the range from 0.5 to 8. A histogram with frequencies of all the values is on the Figure 4. We have divided the values into four approximately equipotent groups (see in the Fig. 4) and these groups determine the target class attribute in our classification task.

5. ILP Models and methods

To make the paper readable we present a short description of the ILP techniques bellow.

5.1. Classical ILP

In our presentation of ILP we follow Dzeroski and Lavrac (2001) and Muggleton and de Raedt (1994).

Definition 1 (Classical ILP task)**.** Given is a set of examples $E = P \cup N$, where P contains positive and N negative examples, and background knowledge B. The task of ILP is to find a hypothesis H such that

$$(\forall e \in P)(B \cup H \models e)$$

and

$$(\forall e \in N)(B \cup H \not\models e).$$

Typically, E consists of ground instances of the target predicate, in our case accident seriousness (see examples in Fig. 5). B typically consists of several predicates (relational tables), which describe properties of an object, in our case properties of an accident (see examples in Fig. 6). The background knowledge can contain also some rules. A hypothesis H typically consists of logic programming rules (see examples in Fig. 9). H added to B entails all positive examples and no negative examples. Main advantage of ILP is its multirelational character, namely B can reside in several relational tables.

5.2. Fuzzy and GAP induction

In our presentation of fuzzy ILP we follow the paper of T. Horváth and P. Vojtáš (Horváth and Vojtáš, 2007) about fuzzy inductive logic programming. We use the approach of the fuzzy logic in narrow sense developed by J. Pavelka (Pavelka, 1979) and P. Hájek (Hájek, 1998). Formulas are of the form φ, x (φ is syntactically the same as in the classical case), they are graded by a truth value $x \in [0, 1]$. A structure $\mathcal{M}$ consist of a domain M and relations are interpreted fuzzy (we do not consider function symbols here). The evaluation $\|\varphi\|_{\mathcal{M}}$ of a formula φ uses truth functions of many valued connectives (our logic is extensional and/or truth functional). The satisfaction $\models_f$ is defined by

$$\mathcal{M} \models_f (\varphi, x) \ iff \ \|\varphi\|_{\mathcal{M}} \geq x.$$

Definition 2 (Fuzzy ILP task)**.** Given are a fuzzy set of examples $\mathcal{E} : E \longrightarrow [0,1]$ and a fuzzy background knowledge $\mathcal{B} : B \longrightarrow [0,1]$. The task of fuzzy ILP is to find a fuzzy hypothesis $\mathcal{H} : H \longrightarrow [0,1]$ such that

$$(\forall e_1, e_2 \in E)(\forall \mathcal{M})(\mathcal{M} \models_f \mathcal{B} \cup \mathcal{H})$$

we have

$$\mathcal{E}(e_1) > \mathcal{E}(e_2) \Rightarrow \|e_1\|_{\mathcal{M}} \geq \|e_2\|_{\mathcal{M}}.$$

That is, it cannot happen that

$$\mathcal{E}(e_1) > \mathcal{E}(e_2) \wedge \|e_1\|_{\mathcal{M}} < \|e_2\|_{\mathcal{M}},$$

or rephrased: if $\mathcal{E}$ is rating e_1 higher than e_2, then it cannot happen that e_1 is rated worse than e_2 in a model of $\mathcal{B} \cup \mathcal{H}$.

Typically, $\mathcal{E}$ consists of ground instances of the target predicate, which are classified in truth degrees – in our case a degree of seriousness of an accident. $\mathcal{B}$ typically consists of several fuzzy predicates (fuzzy relational tables), which describe properties of an object, in our case fuzzy properties of an accident – a degree of injury, a degree of damage, etc. A hypothesis $\mathcal{H}$ typically consists of a fuzzy logic program, which, when added to $\mathcal{B}$, prevents of misclassification (better can not be declared to be worse, nevertheless can be declared as having the same degree — for more detailed discussion on this definition of fuzzy ILP we refer to the paper (Horváth and Vojtáš, 2007)).

Moreover, in practice, we use GAP – Generalized Annotated Programs, so graded formulas will be sometimes understood as annotated (with classical connectives and with a more complex annotation of the heads of rules). This is possible, because in (Krajci et al., 2004) we have shown that (some extension of) fuzzy logic programming is equivalent to (some restriction of) generalized annotated programs.

5.3. Translation of fuzzy ILP task to several classical ILP tasks

As far as there is no implementation of fuzzy ILP or GAP, we have to use a classical ILP system. Fortunately any fuzzy ILP task can be translated to several classical ILP tasks (subject to some rounding and using a finite set of truth values).

In following text assume that all fuzzy sets take truth values only from a finite set of truth values $T : \{0,1\} \subseteq T \subseteq [0,1]$.

Definition 3 (Transformation of background knowledge)**.** Given is a fuzzy background knowledge $\mathcal{B} : B \longrightarrow [0,1]$. For each predicate $p(x)$ in B we add an additional attribute t to express the truth value, thus we have created a new predicate $p(x,t)$. We construct two classical background knowledge sets B_T^{raw} and B_T^{mon} as follows:

- The first (B_T^{raw}) is a direct coding of a fuzzy value by an additional attribute:
  If $\mathcal{B}(p(x)) = t,\ t \in T$, then we add $p(x,t)$ to B_T^{raw}.
- The second (B_T^{mon}) is obtained by a process called monotonization:
  If $\mathcal{B}(p(x)) = t,\ t \in T$, then for all $t' \in T,\ t' \leq t$ we add $p(x,t')$ to B_T^{mon}.
  This correspond to natural meaning of truth values t.

Also example sets are constructed in two ways.

Definition 4 (Transformation of examples)**.** Given is a fuzzy set of examples $\mathcal{E} : E \longrightarrow [0,1]$. For all $t \in T$ we construct two classical sets of examples E_t and $E_{\geq t}$ as follows:

- $E_t = P_t \cup N_t$, where $e \in P_t \ \ iff \ \ \mathcal{E}(e) = t$ and N_t is the rest of E.
- $E_{\geq t} = P_{\geq t} \cup N_{\geq t}$, where $e \in P_{\geq t} \ \ iff \ \ \mathcal{E}(e) \geq t$ and N_t is the rest of E.

These two translations create two classical ILP tasks for each truth value $t \in T$, the first one (*raw*) is crisp and the second one (*mon*) can be understood as (and translated back to) fuzzy ILP.

- The *raw ILP task* is given by B_T^{raw} and E_t for each $t \in T$. As a result it produces a set of hypotheses H_t.
- The *mon ILP task* is given by B_T^{mon} and $E_{\geq t}$ for each $t \in T$. As a result it produces a set of hypotheses $H_{\geq t}$ guaranteeing examples of a degree of at least t.

Note that among variable boundings in B there are no boundings on the truth value attribute, which was added to each predicate, and hence there are no variable boundings on the truth value attribute in $H_{\geq t}$. We did not add an additional truth value attribute to the predicates in E.

Now we sketch the translation of the *mon ILP task* to GAP (fuzzy ILP) rules.

Theorem 1 (Translation of the *mon ILP task*)**.** Given is a fuzzy ILP (or equivalent GAP) task given by $\mathcal{E}$ and $\mathcal{B}$. Let us assume that C is the target predicate in the domain of $\mathcal{E}$ and for each $t \in T$ $H_{\geq t}$ is a correctly learned solution of the corresponding *mon ILP task* according to the definitions above. We define $\mathcal{H}$ consisting of one GAP rule:

$$C(y) : u(x_1, \ldots, x_m) \leftarrow B_1 : x_1 \& \ldots \& B_m : x_m,$$

here $B_1 : x_1 \& \ldots \& B_m : x_m$ is the enumeration of all predicates in B.

Assume that $B_1(y_1, t_1), \ldots, B_n(y_n, t_n)$ are some of the predicates in B (for simplicity enumerated from 1 to n, $n \leq m$). Then for each rule

$$R = C(y) \Leftarrow B_1(y_1, t_1), \ldots, B_n(y_n, t_n)$$

in $H_{\geq t}$ we give a constraint in definition of u as follows:

$$U_R = u(x_1, \ldots, x_m) \geq t \text{ if } x_1 \geq t_1, \ldots, x_n \geq t_n.$$

Note that $x_{n+1}, \ldots, x_m$ have no restrictions.

We claim, that if all $H_{\geq t}$ were correctly learned by a classical ILP system then for the minimal solution u of all constraints U_R the rule

$$C(y) : u(x_1, \ldots, x_m) \leftarrow B_1 : x_1 \& \ldots \& B_n : x_m$$

is a correct solution of the fuzzy ILP task given by $\mathcal{E}$ and $\mathcal{B}$, for all $R \in H_t$ and $t \in T$.

Our presentation is here a little bit simplified and we freely switch between fuzzy and GAP programs, which are known to be equivalent, see in (Krajci et al., 2004).

5.4. Implementation

In our experimental system we are using two inductive logic approaches – Crisp ILP and Fuzzy ILP (as described above). Technically the difference between the approaches consists in different setting of the *ILP task*. Both can be done with a classical ILP tool. We have used "*A Learning Engine for Proposing Hypotheses*" (Aleph v5[3]), which we consider very practical. It uses quite effective method of *inverse entailment* (Muggleton, 1995) and keeps all handy features of a Prolog system (it is supported by YAP and SWI Prolog) in its background.

We have compared results of the Crisp and Fuzzy ILP approaches with other classification methods and in our experiment the fuzzy approach made better results than the crisp one and also than many other methods. See the section 6 for the details.

Crisp learning examples

```
serious_2(id_47443). %positive

serious_0(id_47443). %negative
serious_1(id_47443). %negative
serious_3(id_47443). %negative
```

Monotonized learning examples

```
serious_atl_0(id_47443). %positive
serious_atl_1(id_47443). %positive
serious_atl_2(id_47443). %positive

serious_atl_3(id_47443). %negative
```

Figure 5: Learning examples.

```
size(id_47443, 427).
type(id_47443, fire).
damage(id_47443, 8000).
dur_minutes(id_47443, 50).
fatalities(id_47443, 0).
injuries(id_47443, 0).
cars(id_47443, 0).
amateur_units(id_47443, 3).
profesional_units(id_47443, 1).
pipes(id_47443, unknown).
lather(id_47443, 0).
aqualung(id_47443, 0).
fan(id_47443, 0).
```

Figure 6: B_T^{raw} – crisp attributes.

```
damage_atl(ID,N) :- %unknown values
        damage(ID,N), not(integer(N)).
damage_atl(ID,N) :- %numeric values
        damage(ID,N2), integer(N2),
        damage(N), integer(N), N2>=N.
```

Figure 7: Monotonization of attributes (damage_atl ← damage).

To use ILP for the classification task we have to translate the input data to the Prolog-like logic representation. As already described in previous

[3] http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/

```
serious_0(ID) :- serious_atl_0(ID),
                 not(serious_atl_1(ID)),
                 not(serious_atl_2(ID)),
                 not(serious_atl_3(ID)).
serious_1(ID) :- serious_atl_1(ID),
                 not(serious_atl_2(ID)),
                 not(serious_atl_3(ID)).
serious_2(ID) :- serious_atl_2(ID),
                 not(serious_atl_3(ID)).
serious_3(ID) :- serious_atl_3(ID).
```

Figure 8: Conversion rules for monotonized hypotheses (`serious_t` ← `serious_atl_t`).

sections, the transformations in the crisp and in the fuzzy case are different. Here we will describe the implementation details of the construction of the crisp and fuzzy knowledge bases and example sets.

For the construction of the crisp example set E_t we encode the target (class) predicate as `serious_t`. The letter `t` stands for the actual seriousness degree. We use multiple unary predicates `serious_0`, `serious_1`, etc., instead of one binary predicate `serious(ID,Degree)`. These two cases are equivalent and we have decided to use the unary one because it is better for visual distinction of the multiple ILP tasks.

For the construction of the fuzzy (or monotonized) example set $E_{\geq t}$ we encode the target predicate as `serious_atl_t`, see in the Fig. 5.

For the construction of the crisp background knowledge B_T^{raw} we use a simple translation of the data to the predicates. It is illustrated on the Fig. 6).

For the construction of the monotonized background knowledge B_T^{mon} we reuse the crisp background knowledge and add monotonization rules. An example of the monotonization rules for a crisp predicate of `damage` is shown on the Fig. 7. The first rule deals with `unknown` values and the second serves for the translation.

All the negations we use (Fig. 7 and Fig. 8) are the standard Prolog *negations as failure*.

Once we have the learning examples and the background knowledge, we can run the ILP inductive learning procedure and obtain learned rules (a learned hypothesis). According to the kind of the ILP task (crisp or monotonized) we obtain corresponding kind (crisp or monotonized) of rules (see e.g. in the Fig. 9). But these rules cannot be used directly to solve the classification task. There are common cases when more then one rule is applicable

```
serious_0(A) :- fatalities(A,0), injuries(A,0), cars(A,1), amateur_units(A,0), lather(A,0).
serious_0(A) :- fatalities(A,0), cars(A,0), amateur_units(A,0), professional_units(A,1).
serious_1(A) :- amateur_units(A,1).
serious_1(A) :- damage(A,300000).
serious_1(A) :- type(A,fire), amateur_units(A,0), pipes(A,2).
serious_1(A) :- type(A,car_accident),dur_minutes(A,unknown),fatalities(A,0),injuries(A,1).
serious_2(A) :- lather(A,unknown).
serious_2(A) :- cars(A,0), lather(A,0), aqualung(A,1), fan(A,0).
serious_2(A) :- amateur_units(A,2).
serious_3(A) :- fatalities(A,2).
serious_3(A) :- type(A,fire), dur_minutes(A,unknown), cars(A,0), fan(A,0).
serious_3(A) :- injuries(A,2), cars(A,2).
serious_3(A) :- fatalities(A,1).

serious_atl_0(A).
serious_atl_1(A) :- injuries_atl(A,1).
serious_atl_1(A) :- dur_minutes_atl(A,21), pipes_atl(A,1), aqualung_atl(A,0).
serious_atl_1(A) :- damage_atl(A,8000), amateur_units_atl(A,3).
serious_atl_1(A) :- dur_minutes_atl(A,197).
serious_atl_1(A) :- dur_minutes_atl(A,unknown).
serious_atl_2(A) :- dur_minutes_atl(A,50), pipes_atl(A,3).
serious_atl_2(A) :- size_atl(A,1364), injuries_atl(A,1).
serious_atl_2(A) :- fatalities_atl(A,1).
serious_atl_2(A) :- size_atl(A,1106), professional_units_atl(A,3).
serious_atl_3(A) :- fatalities_atl(A,1).
serious_atl_3(A) :- damage_atl(A,1500000).
```

Figure 9: Crisp and monotonized hypotheses.

to a single instance, which is to be classified. So we have to select which one to use. For the monotonized hypothesis we select the maximum from all the possibilities. It is illustrated on the Fig. 8. For the crisp hypothesis there is not such clear criterion, so we simply use the fist applicable rule.

On the other hand in the crisp case there are often many instances, which cannot be classified because there is no applicable rule. (In our experiment there was about 51% of unclassified instances, see in the next section.) It could be caused by the lack of training data, but the monotonized case does not suffer from this shortage. We can always select the bottom class and – what is probably more important – the set of positive training examples is extended by monotonization.

6. Results

The Fig. 9 summarizes obtained hypotheses learned from our data:

- a crisp hypothesis learned from E_t and B_T^{raw} and
- a monotonized hypothesis learned from $E_{\geq t}$ and B_T^{mon}.

In both cases the learning examples and the background knowledge have the origin in the same data (the same accidents). They differ in the form of the ILP task (crisp and monotonized). The crisp hypothesis uses only the crisp predicates, the monotonized hypothesis uses only the monotonized predicates.

6.1. Evaluation

We have evaluated both ILP methods and compared them with other machine learning procedures used in data mining. To make the comparison clear and easy to perform, we have implemented an interface between the ILP methods and the Weka data mining software[4] (Hall et al., 2009). This interface makes it possible to use the ILP methods as an ordinary Weka classifier for any[5] classification task inside the Weka software. Our implementation is publicly available. The data, source codes and an platform independent installer of the Crisp and Fuzzy ILP classifiers for Weka can be downloaded from our Fuzzy ILP classifier's web page[6]. This makes our experiment repeatable according to the SIGMOD Experimental Repeatability Requirements (Manolescu et al., 2008).

For our experiment we used the Weka Experimenter and performed an experiment in which the Crisp and Fuzzy ILP classifiers were compared with five additional classifiers:

- Multilayer Perceptron (Bishop, 1996),
- Support Vector Machine classifier SMO (Keerthi et al., 2001),
- J48 decision tree (Quinlan, 1993),
- JRip rules (Cohen, 1995) and
- Additive logistic regression LogitBoost (Friedman et al., 2000).

We have evaluated all the methods two times by 10-fold cross validation on our data. The obtained results (average values) are described by the graph

[4] `http://www.cs.waikato.ac.nz/ml/weka/`

[5] For the fuzzy ILP method, there is a requirement on the target (class) attribute: it has to be monotonizable (e.g. numeric).

[6] `http://www.ksi.mff.cuni.cz/~dedek/fuzzyILP/`

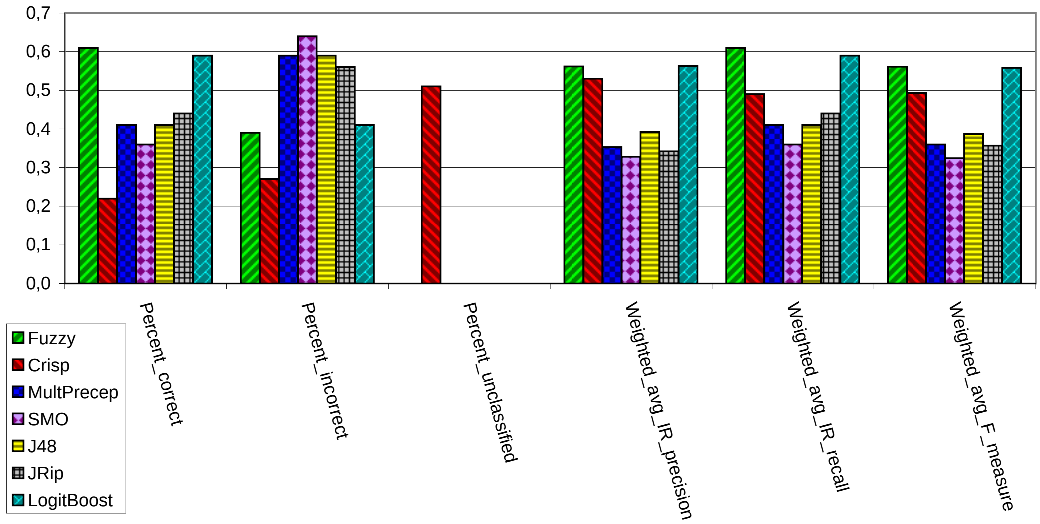

Figure 10: Evaluation of the methods – average values.

	Fuzzy	Crisp	MultPerc	SMO	J48	JRip	LBoost
Corr	0.61±.19	.22±.17 •	.41±.19 •	.36±.24 •	.41±.22 •	.44±.17 •	.59±.26
Incor	.39±.19	.27±.24	.59±.19 ∘	.64±.24 ∘	.59±.22 ∘	.56±.17 ∘	.41±.26
Uncl	.00±.00	.51±.29 ∘	.00±.00	.00±.00	.00±.00	.00±.00	.00±.00
Prec	.56±.24	.53±.37	.35±.20 •	.33±.26	.39±.22	.34±.21 •	.56±.28
Rec	.61±.19	.49±.32	.41±.19 •	.36±.24 •	.41±.22 •	.44±.17 •	.59±.26
F	.56±.20	.49±.33	.36±.19 •	.32±.24 •	.39±.21	.36±.19 •	.56±.27

∘, • statistically significant increase or decrease

Legend:

Fuzzy .... czsem.ILP.FuzzyILPClassifier ”
Crisp .... czsem.ILP.CrispILPClassifier ”
MultPerc ..... functions.MultilayerPerceptron ’-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a’
SMO .... functions.SMO ’-C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K \”functions.supportVector.PolyKernel -C 250007 -E 1.0\”’
J48 .... trees.J48 ’-C 0.25 -M 2’
JRip .... rules.JRip ’-F 3 -N 2.0 -O 2 -S 1’
LBoost .... meta.LogitBoost ’-P 100 -F 0 -R 1 -L -1.7976931348623157E308 -H 0.1 -S 1 -I 10 -W trees.DecisionStump’

Corr .... Percent correct
Inor .... Percent incorrect
Uncl .... Percent unclassified
Prec .... IR precision, weighted average from all classes
Rec .... IR recall, weighted average from all classes
F .... F measure, weighted average from all classes

Table 2: Evaluation of the methods in 2 times 10-fold cross validation.

on the Fig. 10 and in the Table 2 (with standard deviations and decorated statistically significant values).

There is no clear winner in our experiment. But the Fuzzy ILP classifier proved better results than majority of the methods on our data and the results are statistically significant in many cases. Very good results obtained also the LogitBoost.

UCI datasets

	Fuzzy	Crisp	MultPerc	SMO	J48	JRip	LBoost	train	test
car	.39±.03	.36±.03 •	.53±.02 ∘	.57±.01 ∘	.50±.02 ∘	.51±.03 ∘	.54±.02∘	173	1554
wine	.44±.03	.42±.02 •	.48±.02 ∘	.46±.02 ∘	.47±.02 ∘	.48±.03 ∘	.52±.02∘	160	1439
cmc	.79±.02	.77±.03 •	.89±.02 ∘	.81±.01 ∘	.88±.02 ∘	.82±.03 ∘	.85±.02∘	147	1325
tae	.50±.12	.39±.11 •	.59±.11 ∘	.55±.12 ∘	.50±.12	.37±.11 •	.55±.11∘	135	15
pop	.66±.09	.54±.17 •	.57±.13 •	.70±.06 ∘	.70±.07 ∘	.70±.06 ∘	.66±.10	80	9
nurs	.79±.04	.68±.06 •	.81±.04 ∘	.73±.04 •	.80±.04 ∘	.79±.06	.83±.02∘	52	12907

∘, • statistically significant improvement or degradation

Legend:

train .... average number (±1) of training instances in each run
test .... average number (±1) of testing instances in each run

car ....`http://archive.ics.uci.edu/ml/datasets/Car+Evaluation`
wine .... red wine dataset `http://archive.ics.uci.edu/ml/datasets/Wine+Quality`
cmc ....`http://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice`
tae ....`http://archive.ics.uci.edu/ml/datasets/Teaching+Assistant+Evaluation`
pop ....`http://archive.ics.uci.edu/ml/datasets/Post-Operative+Patient`
nurs ....`http://archive.ics.uci.edu/ml/datasets/Nursery`

Learning parameters for both ILP methods:
set(noise,20). set(i,3). set(clauselength,13). set(search,heuristic). set(evalfn,wracc). set(samplesize,3).

Table 3: Evaluation of the methods on UCI datasets, **percent correct**, average values form 100 repetitions.

The Fuzzy ILP classifier performed quite well on our dataset, but the next question is: How is it with other data, with more learning instances and what about the time complexity? To answer these questions we have done another experiment. We have selected several datasets form the University of California, Irvine, Repository of Machine Learning Databases (UCI) (Frank and Asuncion, 2010) and evaluated all the methods against them.

The list of selected datasets can be found in the legend of Table 3. All the dataset are monotonizable (the target attribute can be naturally ordered), so the fuzzy classifier could take advantage of that. Learning settings are the same as before (Table 2) except settings of both ILP classifiers, which on majority of the datasets performed a little bit better with modified settings (see in the legend of Table 3).

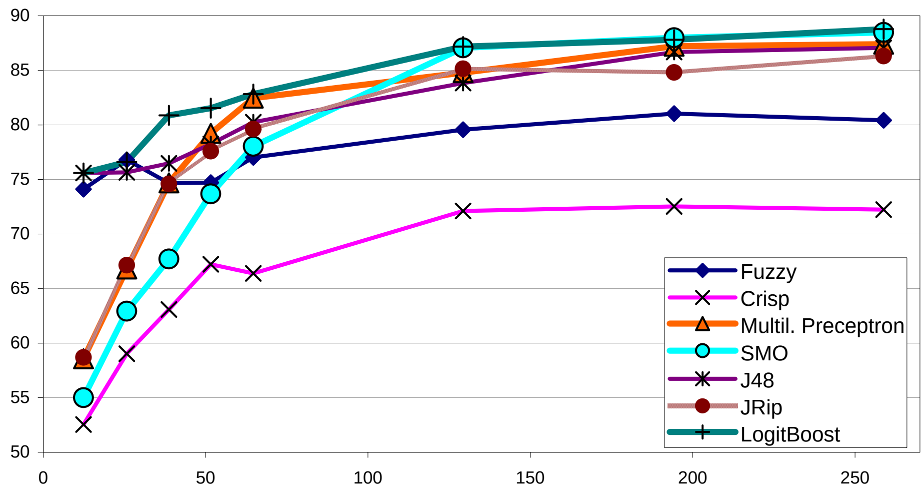

Figure 11: x-axis: number of training instances, y-axis: percent of correctly classified instances, average values form 10 repetitions, 'nursery' dataset.

Table 3 compares the numbers of correctly classified instances on all the datasets. Last two columns show numbers of training and testing instances. The numbers of training instances are quite low; this is because the ILP classifiers are probably not capable to fully exploit higher numbers of training instances and the difference between ILP classifiers and the others would be even a bit bigger. This is demonstrated on Fig. 11 (for 'nursery' dataset only). It can be seen that when the number of training instances was under about 40, the fuzzy classifier performed better than some of the others (SMO, JRip and Multilayer Perceptron), but from about 60 training instances the ILP classifiers performed worst on the 'nursery' dataset.

Fig. 12 demonstrates time complexity of the classifiers in the same experiment as in Fig. 11. Despite the fact that Fuzzy ILP classifier was several times slower than the Crisp ILP classifier and even more than the others it is still computable on current processors (e.g. P9600, 2.66 GHz that we have used) and the curve of time complexity did not grow any rapidly during the experiment. Because ILP is a heuristic and iterative method the time complexity can be quite directly managed by the setting of learning parameters.

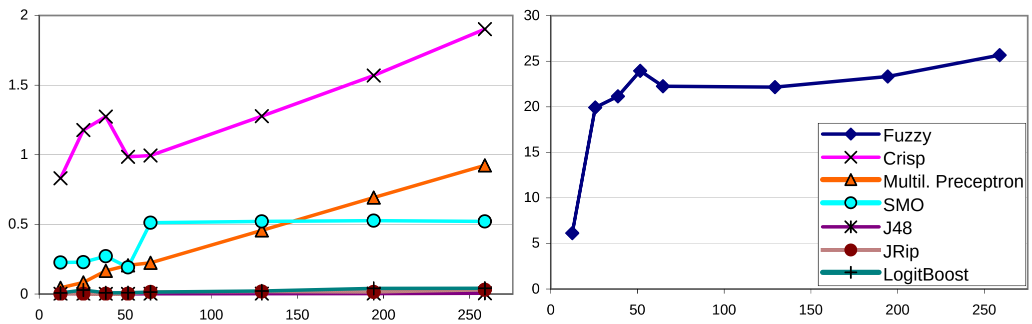

Figure 12: x-axis: number of training instances, y-axis: training time in seconds, average values form 10 repetitions, 'nursery' dataset.

7. Conclusion

In this paper we have presented a fuzzy system, which provides a fuzzy classification of textual web reports. Our approach is based on usage of third party linguistic analyzers, our previous work on web information extraction and fuzzy inductive logic programming.

Main contributions are formal models, prototype implementation of the presented methods and an evaluation experiment. Our data and our implementation is publicly available on the Web. The experiment has shown the performance of the presented methods and compared them with other machine learning procedures used in data mining. The Fuzzy ILP classifier proved better results than majority of the methods on our data. The results are statistically significant in many cases. We see the advantage of the Fuzzy ILP classifier in the fact that monotonization leads to the extension of the learning domain and it utilizes the fact that the domain is or can be monotonically ordered.

We have not performed any additional experiments with other data sets so far because we are primarily interested in our data. This is to be done in the future works and thanks to the public implementation it could be done by anyone interested.

Acknowledgments

This work was partially supported by Czech projects: GACR P202/10/0761, GACR-201/09/H057, GAUK 31009 and MSM-0021620838.

References

Bioch, J., Popova, V., 2009. Monotone decision trees and noisy data. Research Paper ERS-2002-53-LIS, Erasmus Research Institute of Management (ERIM), ERIM is the joint research institute of the Rotterdam School of Management, Erasmus University and the Erasmus School of Economics (ESE) at Erasmus University Rotterdam.
URL `http://econpapers.repec.org/RePEc:dgr:eureri:2002206`

Bioch, J. C., Popova, V., 2000. Rough sets and ordinal classification. In: ALT '00: Proceedings of the 11th International Conference on Algorithmic Learning Theory. Springer-Verlag, London, UK, pp. 291–305.

Bishop, C. M., January 1996. Neural Networks for Pattern Recognition, 1st Edition. Oxford University Press.
URL `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0198538642`

Chong, M., Abraham, A., Paprzycki, M., 2005. Traffic accident analysis using machine learning paradigms. Informatica 29, 89–98.

Cohen, W. W., 1995. Fast effective rule induction. In: In Proceedings of the Twelfth International Conference on Machine Learning. pp. 115–123.
URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.50.8204`

Dědek, J., Eckhardt, A., Vojtáš, P., 2008. Experiments with czech linguistic data and ILP. In: Železný, F., Lavrač, N. (Eds.), ILP 2008 (Late Breaking Papers). Action M, Prague, Czech Republic, pp. 20–25.

Dědek, J., Vojtáš, P., 2008. Computing aggregations from linguistic web resources: a case study in Czech Rep. sector/traffic accidents. In: Dini, C. (Ed.), 2nd International Conference on Advanced Engineering Computing and Applications in Sciences. IEEE Computer Society, pp. 7–12.
URL `http://www2.computer.org/portal/web/csdl/doi/10.1109/ADVCOMP.2008.17`

Dzeroski, S., Lavrac, N. (Eds.), 2001. Relational Data Mining. Springer, Berlin.
URL `http://www-ai.ijs.si/SasoDzeroski/RDMBook/`

Frank, A., Asuncion, A., 2010. UCI machine learning repository.
URL http://archive.ics.uci.edu/ml

Friedman, J., Hastie, T., Tibshirani, R., 2000. Additive logistic regression: a statistical view of boosting. Annals of statistics 28 (2), 337–374.

Hájek, P., 1998. Metamathematics of Fuzzy Logic. Kluwer.

Hajič, J., Hajičová, E., Hlaváčová, J., Klimeš, V., Mírovský, J., Pajas, P., Štěpánek, J., Vidová-Hladká, B., Žabokrtský, Z., 2006. Prague dependency treebank 2.0 cd–rom. Linguistic Data Consortium LDC2006T01, Philadelphia 2006.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H., 2009. The weka data mining software: an update. SIGKDD Explor. Newsl. 11 (1), 10–18.

Horváth, T., Vojtáš, P., 2007. Induction of fuzzy and annotated logic programs. ILP: 16th International Conference, ILP 2006, Santiago de Compostela, Spain, August 24-27, 2006, Revised Selected Papers, 260–274.

Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., Murthy, K. R. K., 2001. Improvements to platt's smo algorithm for svm classifier design. Neural Computation 13 (3), 637–649.

Klimeš, V., 2006. Transformation-based tectogrammatical analysis of czech. In: Proceedings of the 9th International Conference, TSD 2006. No. 4188 in Lecture Notes In Computer Science. Springer-Verlag Berlin Heidelberg, pp. 135–142.

Kotlowski, W., Slowinski, R., 2009. Rule learning with monotonicity constraints. In: Danyluk, A. P., Bottou, L., Littman, M. L. (Eds.), ICML. Vol. 382 of ACM International Conference Proceeding Series. ACM, p. 68.

Krajci, S., Lencses, R., Vojtas, P., 2004. A comparison of fuzzy and annotated logic programming. Fuzzy Sets and Systems 144, 173–192.

Lievens, S., Baets, B. D., Cao-Van, K., 2008. A probabilistic framework for the design of instance-based supervised ranking algorithms in an ordinal setting. Annals OR 163 (1), 115–142.

Liu, B., 2007. Web Data Mining. Springer-Verlag.
URL http://dx.doi.org/10.1007/978-3-540-37882-2

Manolescu, I., Afanasiev, L., Arion, A., Dittrich, J., Manegold, S., Polyzotis, N., Schnaitter, K., Senellart, P., Zoupanos, S., Shasha, D., 2008. The repeatability experiment of sigmod 2008. SIGMOD Rec. 37 (1), 39–45.

Mikulová, M., Bémová, A., Hajič, J., Hajičová, E., Havelka, J., Kolářová, V., Kučová, L., Lopatková, M., Pajas, P., Panevová, J., Razímová, M., Sgall, P., Štěpánek, J., Urešová, Z., Veselá, K., Žabokrtský, Z., 2006. Annotation on the tectogrammatical level in the prague dependency treebank. annotation manual. Tech. Rep. 30, ÚFAL MFF UK, Prague, Czech Rep.

Muggleton, S., 1995. Inverse entailment and progol. New Generation Computing, Special issue on Inductive Logic Programming 13 (3-4), 245–286.
URL http://citeseer.ist.psu.edu/muggleton95inverse.html

Muggleton, S., de Raedt, L., 1994. Inductive logic programming: Theory and methods. Journal of Logic Programming 19, 629–679.

Pavelka, J., 1979. On fuzzy logic i, ii, iii. Zeitschr. Math. Logik und Grundl. Math. 25, 45–52, 119–134, 447–464.

Quinlan, J. R., 1993. C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Reformat, M., Yager, R. R., Li, Z., 2008. Ontology enhanced concept hierarchies for text identification. Journal Semantic Web Information Systems 4 (3), 16–43.