

Fuzzy Classification of Web Reports with Mining Text with Linguistic Annotations

Jan Dědek, Peter Vojtáš, Marta Vomlelová

*Department of Software Engineering, Charles University
Institute of Computer Science, Czech Academy of Science
Prague, Czech Republic*

Abstract

In this paper we present 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: comparison of our classifier with other alternatives like decision trees, support vector machines, neural networks etc, extensive evaluation experiments and introduction of combined evaluation measures. Main conclusion is that, even under low precision of linguistic annotation, classification is satisfactory.

Keywords: fuzzy, inductive logic programming

1. Introduction

Our motivating example are messages of accident reports on the web. We would like to have a tool which is able to classify such message with degree of being it a serious accident.

Our solution is based first on information extraction (see emphasized information to be extracted) and second on processing this information to get fuzzy classification rules. In this paper we are concentrated on the second phase – a fuzzy classification.

Main contributions of this paper can be stated as follows:

- formal models for fuzzy classification of information from web reports
- prototype implementation of a fuzzy classification system
- experimental evaluation of the fuzzy classification system

2. Models, methods, design of the system

General schema of our system is in Fig 1. We use our previously developed web information extraction tools based on third party linguistic analyzer (the upper two dashed arrows). The classification is based on fuzzy ILP and its translation to several crisp ILP tasks. We assume that a small amount of learning data are annotated by a human.

Figure 1: Schema of the whole system.

Here we refer to our previous work Dėdek et al. (2008). A long path of tools starting with web crawling and resulting with the extracted structured information is developed in our previous work.

2.1. Fuzzy and GAP induction

In our presentation of Inductive Logic Programming (ILP) we follow Dzeroski and Lavrac (2001) and Muggleton and de Raedt (1994), for fuzzy Inductive Logic Programming (fILP) we follow the paper of T. Horvath and P. Vojtas 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. Hajek Hajek (1998). Formulas are of the form φ, x (φ is syntactically same as in the crisp case) are graded by a truth value $x \in [0, 1]$. A structure \mathcal{M} consist of domain M and relations are interpreted fuzzy (we do not consider function symbols here). Evaluation $\|\varphi\|_{\mathcal{M}}$ of a formula φ uses truth functions of many valued connectives (our logic is extensional and/or truth functional). Satisfaction is defined by

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

Given is a fuzzy set of examples $\mathcal{E} : E \rightarrow [0, 1]$ and a fuzzy background knowledge $\mathcal{B} : B \rightarrow [0, 1]$. The task is to find a fuzzy hypothesis $\mathcal{H} : H \rightarrow [0, 1]$ such that

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

we have

$$\mathcal{E}(e) > \mathcal{E}(f) \Rightarrow \|e\|_{\mathcal{M}} \geq \|f\|_{\mathcal{M}}.$$

That is, it cannot happen that

$$\mathcal{E}(e) > \mathcal{E}(f) \wedge \|e\|_{\mathcal{M}} < \|f\|_{\mathcal{M}},$$

or rephrased, if \mathcal{E} is rating e higher than f , then it can not happen in a model of $\mathcal{B} \cup \mathcal{H}$ that e is rated worse than f .

Typically, \mathcal{E} consists of ground instances of the target predicate which are classified in truth degrees - in our case degree of seriousness of an accident. \mathcal{B} typically consists of several fuzzy predicates (fuzzy relational tables) which describe properties of object which have to be classified - in our case fuzzy properties of accidents - degree of injury, degree of damage, 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 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 crisp connectives and more complex annotation of head 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.

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

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

Assume, our fuzzy sets take values for a finite set of truth values $\{0, 1\} \subseteq T \subseteq [0, 1]$. For each predicate $p(x)$ in B we add an additional attribute for truth value $p(x, t)$. We construct two crisp background knowledge sets \mathcal{B}_T^{raw} and \mathcal{B}_T^{mon} as follows:

First is a direct coding of the fuzzy value by an additional attribute:

If $\mathcal{B}(p(x)) = t \in T$, then for we add $p(x, t') \in B_T^{raw}$.

Second is obtained by a process called monotonicization:

If $\mathcal{B}(p(x)) = t \in T$, then for all $t' \in T, t' \leq t$ we add $p(x, t') \in B_T^{mon}$.

Also example sets are constructed in two ways.

For all $t \in T$ we create a crisp example set $E_t = P_t \cup N_t$, where $e \in P_t$ iff $\mathcal{E}(e) = t$ and N_t is the rest of E .

For all $t \in T$ we create a crisp example set $E_{\geq t} = P_{\geq t} \cup N_{< t}$, where $e \in P_{\geq t}$ iff $\mathcal{E}(e) \geq t$ and N_t is the rest of E .

These two translation create two ILP tasks, first is purely crisp and second can be understood (and translated back to) FILP.

First *raw ILP task* is for each $t \in T$ given by E_t and B_T^{raw} , as a result we get a set of hypothesis H_t .

Second, for each $t \in T$ we create a crisp ILP task $E_{\geq t}, B_T^{mon}$ and get a hypothesis $H_{\geq t}$ guaranteeing examples of degree at least t . Note that variable boundings in B have no boundings on truth value attribute, which was added to each predicate, and hence there are no variable boundings in $H_{\geq t}$ on truth value attribute. To predicates in E we did not add the additional truth value attribute

Now we sketch the translation of second ILP task to GAP (FILP) rules. Let us assume C is the target predicate in the domain of \mathcal{E} . We define \mathcal{H} with domain consisting of one GAP rule $C(y) : u(x_1, \dots, x_m) \leftarrow B_1 : x_1 \& \dots \& B_n : x_m$, here $B_1 : x_1 \& \dots \& B_n : x_m$ is enumeration of all predicates in B .

Assume $B_1(y_1, t_1), \dots, B_n(y_n, t_n)$ are some predicates in B (for simplicity enumerated from 1 to $n \leq m$). Then for each rule $R = C(y) \leftarrow B_1(y_1, t_1), \dots, B_n(y_n, t_n)$ in H_t we give a constraint in definition of u as follows

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

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

We claim, that if all H_t were correctly learned by an crisp ILP system then for u the minimal solution of all constraints U_R for all $R \in H_t$, for all $t \in T$, the rule

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

is a correct solution to fuzzy ILP task given by \mathcal{E} and \mathcal{B} . Our presentation is here a little bit simplified and we freely switch between fuzzy and GAP programs, which are know to be equivalent, see Krajci et al. (2004).

3. The system prototype and our experiment

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 semantic information. We use web reports of

Table 1: Accident attributes.

fire departments of several regions of the Czech Republic. These reports are written in Czech language and can be accessed through the web of General Directorate of the Fire and Rescue Service of the Czech Republic¹.

For the present 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 section 3.1. To each report we have also assigned a value of overall ranking of seriousness of presented accident, which is the target of the classification.

For the seriousness classification we have used two inductive logic approaches – Classical ILP and Fuzzy ILP (as described above). Technically the difference between the approaches consist in different setting of *ILP task*. Both can be done with a classical ILP tool. We have used “*A Learning Engine for Proposing Hypotheses*” (Aleph v5²), which seems very practical to us. It use quite effective method of *inverse entailment* Muggleton (1995) and keeps all handy features of Prolog system (supports YAP and SWI) in its background.

We have compared results of the two approaches (fuzzy and classical) and we could see that the fuzzy approach made better results than the classical one. See section 4 for details of the results.

3.1. Features of accidents

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. In some cases value of some attribute is unknown. We have decided to make evidence of this and keep the values **unknown** in a knowledge base. Short explanation of each attribute follows.

size is a file size of text part of a report.

damage is an amount (in CZK – Czech Crowns) of summarized damage arisen during an accident.

dur_minutes is time taken to handle an accident.

fatalities and **injuries** are numbers of fatalities (and injuries) taken by

¹<http://www.hzscr.cz>

²<http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/>

Crisp learning examples

Monotonized learning examples

Figure 2: Learning examples.

damage \rightarrow damage_atl

Figure 3: Monotonization of attributes.

an accident.

cars is number of cars damaged during an accident (especially during car accidents).

professional_units and **amateur_units** are numbers of fireman units sent for an accident.

pipes is number of used fire pipes.

lather, **aqualung** and **fan** (ventilator) indicates whether these devices were used.

3.2. Seriousness ranking

Values of overall seriousness ranking attribute were stated from “general impression” from report’s texts with respect to the particular attributes. Values of seriousness ranking have evolved to 14 distinct values in range from 0.5 to 8. We have divided the values into four approximately equipotent groups and logic rules were learned for each group separately³.

3.3. Data transformation

As already described in previous section, we have two possibilities to organize crisp ILP tasks, one with raw data and second with monotonized data. For this we have to translate extracted data.

For the construction of the E_t example set in our application we encode it in the predicate **serious_t**, For the construction of the $E_{\geq t}$ example set in our application we encode it in the predicate **serious_atl_t**, see Fig 2.

For the construction of monotonized set of background knowledge B_T^{mon} we use rules, here illustrated on predicates **damage** and **damage_atl**, see Fig 3.

³We do not have a binary rule which would return an exact value of the rating, but we have one “true/false” rule for each of the four categories.

Figure 4: Crisp & monotonized hypothesis

Figure 5: Evaluation results

Here, first rules deals with **unknown** values and the second constructs the translation.

4. Results

The Fig 4 summarizes sets of obtained rules from two experiments:

1. experiments with E_t and B_T^{raw}
2. experiments with $E_{\geq t}$, B_T^{mon} .

In both experiments learning and testing examples and the background knowledge had their origin in the same data (the same accidents) but they differ in the form of the ILP task (crisp and monotonized). The Fig 5 shows the success of these rules on the test data (not used during learning). We have used standard performance measures⁴ often used in information retrieval. There are also exact numbers of *true positives* TP and *false positives* FP from which the performance measures were computed (with respect to the numbers of all testing examples⁵).

Naturally monotonized hypotheses $H_{\geq t}$ were tested on monotonized examples $E_{\geq t}$ (double-line frame in the Fig 4) and similarly for the crisp case H_t, E_t (dashed-line frame in the Fig 4).

We wanted to compare raw and monotonized learning tasks. As they run on different example sets we had to translate results of one learning (rules with head **serious_t**) to results of second learning (rules with head **serious_atl_t**). Logic programming translation rules are depicted on the Fig 6. Here the negation “**not(serious_atl.3(ID))**” means a standard Prolog *negation as failure*.

Then the comparison of both learnings is possible, see the numbers in white areas of Fig 5.

⁴http://en.wikipedia.org/wiki/Precision_and_recall

⁵These numbers are written in the Fig 5 in small frames under each test set label.

crisp \rightarrow monotone

monotone \rightarrow crisp

Figure 6: Conversion of results

5. Conclusion

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

Main contributions are formal models and prototype implementation of the system and evaluation experiments with the Fuzzy ILP classification method.

Experiments have shown better results of fuzzy approach. We see the difference in the fact that monotonization leads to the extension of the learning domain.

Acknowledgment

This work was partially supported by Czech projects: IS-1ET100300517, GACR-201/09/H057, GAUK 31009 and MSM-0021620838.

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.

Dzeroski, S., Lavrac, N. (Eds.), 2001. Relational Data Mining. Springer, Berlin.

URL <http://www-ai.ijs.si/SasoDzeroski/RDMBook/>

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

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.

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

- Muggleton, S., 1995. Inverse entailment and prolog. *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 Grundle. Math.* 25, 45–52, 119–134, 447–464.