Combining Counterpropagation Neural Networks and Defeasible Logic Programming for Text Classification

Sergio Alejandro Gómez[†,1] Carlos Iván Chesñevar[‡]

[†]Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)[2]
Depto. de Ciencias e Ingeniería de la Computación — Universidad Nacional del Sur
Av. Alem 1253 – B8000CPB Bahía Blanca – ARGENTINA
TEL/FAX: (+54) (291) 459 5135/5136 – EMAIL: sag@cs.uns.edu.ar

[‡]Grupo de Investigación en Inteligencia Artificial
Departament d'Informàtica — Universitat de Lleida
Campus Cappont – C/Jaume II, 69 – E-25001 Lleida, SPAIN
TEL/FAX: (+34) (973) 70 2764 / 2702 – EMAIL: cic@eup.udl.es

Abstract

The increasing growth of documents available in the World Wide Web has resulted in a difficult situation for those end-users who search for a particular piece of information. A common approach to facilitate search is to perform document classification first, learning the topology of a document base as a set of clusters. Clusters will be labeled as relevant or irrelevant, and determining whether a new document belongs to a given cluster can help determine whether such document corresponds to the user information needs.

We contend that the above clustering technique can be enriched by additional filtering criteria specified in terms of Defeasible Logic Programming (DeLP). In this paper we discuss a combination of Counterpropagation Neural Networks for clustering and DeLP to solve the problem of classifying documents according to user-specified criteria. We present an example of how the proposed approach works.

Keywords: Artificial Intelligence, Machine Learning, Defeasible Argumentation, Counterpropagation neural networks, text mining.

1 Introduction and Motivations

The increasing growth of documents available in the World Wide Web has resulted in a difficult situation for those end-users who search for a particular piece of information. Although search engines provide a useful tool for document retrieval, such procedure is typically based on keyword matching to the elements of a document base. However, the existence (resp., inexistence) of some keyword in some document does not necessarily guarantee the relevance (resp., irrelevance) of the latter with respect to the user's information needs.

A common approach to facilitate search is to perform document classification first, learning the topology of a document base as a set of clusters. Clusters will be labeled as relevant or irrelevant, and determining whether a new document belongs to a given cluster can help determine whether such document corresponds to the user information needs.

In many situations, reasoning with clusters involves complex relationships (e.g. overlapping among clusters), so that commonsense reasoning techniques are required in order to refine the search. *Defeasible argumentation* provides a useful formalization for commonsense reasoning which has been used in many real-world applications. We contend that such clustering technique

[1]Partially supported by the Agencia Nacional de Promoción Científica y Tecnológica (PICT 2002 Nro 13096).
[2]LIDIA is a member of IICyTI Instituto de Investigación en Ciencia y Tecnología Informática.

can be enriched by additional filtering criteria specified in terms of a defeasible argumentation formalism, namely Defeasible Logic Programming (DeLP).

In this article we discuss the problem of document classification by a combination of a Counterpropagation neural network (CPN) model and Defeasible Logic Programming (DeLP). This article characterizes part of a current research in which we are investigating the integration of machine learning and defeasible argumentation techniques (see for instance [6, 5, 8, 7]). The rest of the paper is structured as follows. First in Section 2 the fundamentals of DeLP and CPN are introduced. Then in Section 3 we discuss how to carry out document classification in terms of a combination of a Counterpropagation neural network trained on a set of documents and a domain theory characterized in terms of DeLP. Finally Section 4 discusses the main conclusions that have been obtained and outlines some future research work.

2 Background

Next we will briefly summarize the basic elements of defeasible logic programming and counter-propagation neural networks. For a detailed treatment the reader is referred to the literature cited on these topics.

2.1 Defeasible Logic Programming

Defeasible Logic Programming (DeLP) [4] provides a language for knowledge representation and reasoning that uses *defeasible argumentation* [9, 2] to decide between contradictory conclusions through a *dialectical analysis*. Codifying knowledge by means of a DeLP program provides a good trade-off between expressivity and implementability. Recent research has shown that DeLP provides a suitable framework for building real-world applications (e.g. clustering algorithms [7] and intelligent web search [1]) that deal with incomplete and potentially contradictory information.

In a defeasible logic program $\mathcal{P} = (\Pi, \Delta)$, a set Δ of defeasible $p \prec q_1, \ldots, q_n$, and a set Π of strict rules $p \leftarrow q_1, \ldots, q_n$ can be distinguished. An *argument* $\langle \mathcal{A}, h \rangle$ is a minimal non-contradictory set of ground defeasible clauses $\mathcal{A}$ of Δ that allows to derive a ground literal h possibly using ground rules of Π. Since arguments may be in conflict (concept captured in terms of a logical contradiction), an attack relationship between arguments can be defined. A criterion is usually defined to decide which argument of two conflicting arguments is preferred.In order to determine whether a given argument $\mathcal{A}$ is ultimately undefeated (or *warranted*), a dialectical process is recursively carried out. Given a DeLP program $\mathcal{P}$ and a query h, the final answer to h wrt $\mathcal{P}$ takes such dialectical analysis into account.

2.2 Counterpropagation Neural Networks

The counterpropagation neural network (CPN) [10] learns a mapping $x_i \mapsto f(x_i)$ from a n-dimensional space to a m-dimensional space for a set of example pattern vectors $\{(x_1, f(x_1)), \ldots, (x_n, f(x_n))\}$. The general architecture of the CPN has three layers: an *input layer* composed of a number of fan-out units; one hidden or *Kohonen layer*, containing a set of competitive units, and a linear output or *Grossberg layer*.

The CPN learns to produce $f(x_i)$ given x_i by adapting the weight vectors according to the following learning rule: $w_u(t+1) = w_u(t) + \alpha(x_i - w_u(t+1))$, where u corresponds to the Kohonen winner unit. The winner unit u is selected using some similarity metric. The Kohonen layer is first trained to develop a characterization of the input x_i in terms of a set of clusters.

The Grossberg layer is then trained, learning an average prototype for each hidden or Kohonen unit.

At explotation time, given an input pattern vector x, the most alike unit u is selected by comparing x to each unit prototype and selecting the most similar. Then the Grossberg prototype associated to u is then returned. The latter is considered as $f(x)$.

3 Relating Counterpropagation Neural Networks and DeLP for Text Classification

In this section we will discuss an approach for combining a counterpropagation neural network and a defeasible logic program for text document classification. In the sequel we will use HTML as the underlying model for text documents, although the same approach could be applied to any other Web-based text language which provides tags (e.g. XML, etc.).

3.1 Document Representation

The HTML model lets authors specify *meta data* (i.e., information about the document itself rather than document content) through the use of META tags. For example:

<META Name="Author" Lang="sp" Content="Jorge Luis Borges">
<META Name="Title" Content="El General Quiroga va en coche al muere">
<META Name="Keywords" Lang="sp" Content="poema, Quiroga, muere">
<META Name="Date" Content="12/04/1986">

Note that here the META elements provide information about the author's name, document title, keywords, date, etc. In order to use a document D as an input pattern for a counterpropagation neural network, an internal representation for the document has to be adopted [3]. A vector space representation can be used modeling each document d_i by a feature vector $\langle k_i^1, \ldots, k_i^n \rangle$ of user-provided keywords. Together with this representation we are going to suppose that a numerical rank r_i is given by the user for each document d_i, regarding a degree of the document relevance wrt his information needs. We will also assume that the rank belongs to the real interval $[0, 1]$, and the greater the rank, the greater the relevance of the document.

3.2 Modeling the Neural Network Output in DeLP

A counterpropagation neural network $\mathcal{C}$ will be trained over a set of documents, and will be used for inducing a mapping from that set of documents into real numbers, such that, given a (possibly) previously unseen document, the network $\mathcal{C}$ will be capable of providing a guess on the document relevance wrt the user's information needs.

We want to encode information from the neural network in a DeLP setting. To do so, we will provide an interface in terms of a DeLP program with built-in predicates which invoke computations based on the neural network $\mathcal{C}$. For example:

$$\begin{array}{lcl} relevant(D) & -\!\!\prec & rank(D,R),\ R \geq 0.5 \\ \sim relevant(D) & -\!\!\prec & rank(D,R),\ R < 0.5 \\ rank(D,R) & \leftarrow & kohonen(D,U),\ grossberg(U,R) \end{array}$$

Here we assume the document D is relevant if the rank R assigned to it by the neural network is greater than or equal to 0.5 and that is irrelevant otherwise. The predicates $kohonen(D,U)$ and $grossberg(U,R)$ model the internal network behavior. The predicate $kohonen(D,U)$ retrieves the winner unit U associated to D in the neural net self-organizing Kohonen layer. The predicate $grossberg(U,R)$ retrieve the rank R calculated by the neural net Grossberg layer.

3.3 Providing User Filtering Criteria in DeLP

The previous analysis corresponds to a rather direct translation from the behavior of the underlying neural networks into characterizing relevance. Other, more involved filtering criteria can be extend the previous one, e.g. by adding the following rules:

$$
\begin{array}{lcl}
relevant(D) & \prec & author(D, borges) \\
\sim relevant(D) & \prec & keyword(D, poetry) \\
\sim relevant(D) & \prec & old(D) \\
old(D) & \prec & date(D, DMY), before(DMY, \text{“01/01/1990”})
\end{array}
$$

The above rules specify that a document D is relevant if its author is Borges but it is irrelevant (or not relevant) if the document is about poetry. Besides the document D is also considered irrelevant if it is old, that is, its date is previous to January 1990. This information is gathered by the predicates $author(D, A)$, $keyword(D, K)$, $date(D, DMY)$ that take into account the corresponding document META keyword.

3.4 Classifying Documents Through a Dialectical Analysis

From the above example we can see that user-provided criteria can be contradictory. In order to classify a previously unseen document d a dialectical analysis is needed, and will be performed automatically by the DeLP engine. In the process the DeLP engine will first try to warrant an argument $\langle \mathcal{A}, relevant(d) \rangle$, supporting to the conclusion that the document is relevant. If the former could not be warranted, then the DeLP engine will try to warrant the opposite argument, that is $\langle \mathcal{A}, \sim relevant(d) \rangle$, which supports the opposite.

For the sake of example we will consider some arguments that can be derived in the above context. Given a document d containing a poem written by Jorge Luis Borges, there are two conflicting arguments that can be built from the DeLP program above, namely $\langle \mathcal{A}_1, relevant(d) \rangle$ and $\langle \mathcal{A}_2, \sim relevant(d) \rangle$, where

$$
\mathcal{A}_1 = \{relevant(d) \prec author(d, borges)\}
$$
$$
\mathcal{A}_2 = \{\sim relevant(d_1) \prec keyword(d, poem)\}
$$

Let us also suppose that a CPN $\mathcal{C}$ trained on a sample document set is also used in the classification of this document d. Let us assume that according to $\mathcal{C}$ the rank r of d is greater than or equal to 0.5. In that case another argument $\langle \mathcal{A}_3, relevant(d) \rangle$ supporting the relevance of d can be built[3], where $\mathcal{A}_3 = \{relevant(d) \prec rank(d, r), r \geq 0.5\}$. Similarly, the argument $\langle \mathcal{A}_4, \sim relevant(d) \rangle$ supporting the irrelevance of document d can be derived too, where:

$$
\begin{aligned}
\mathcal{A}_4 = \{ & (\sim relevant(d) \prec old(d)), \\
& (old(d) \prec date(d, \text{“12/04/1986”}), before(\text{“12/04/1986”}, \text{“01/01/1990”}))\}
\end{aligned}
$$

As illustrated above, several conflicting arguments can be obtained from the same set of preference criteria (DeLP program), so that a dialectical analysis is needed in order to determine which arguments are to be ultimately preferred. Such analysis is automatically carried out by DeLP. As pointed out in [9, 4] it must be noted that comparison criterion among arguments is modular, and can be defined in several ways, being independent from the inference procedure for computing warranted arguments.

4 Conclusions and Future Work

In this paper we have outlined how neural networks can be integrated into a DeLP setting in order to perform text classification. The proposed hybrid approach combines the features of

[3] An argument against the document relevance could have also been built if the rank r of d_1 had been less than 0.5.

CPN to obtain clusters from vectors characterizing text documents together with the capabilities of DeLP for performing commonsense reasoning. As we have shown in this paper, the existence of conflicting user preferences can be captured in terms of a dialectical analysis, automatically performed by the DeLP inference engine.

It must be remarked that the approach presented in this paper can be seen as a particular case of how frameworks for defeasible argumentation can be integrated them machine learning techniques. Part of our current research work involves testing the ideas presented in this paper on some sample document collections, using different user-defined criteria encoded as DeLP programs.

Acknowledgements

Research partially supported by Projects CICYT TIC2001-1577-C03-03, TIC2003-00950 and Ramón y Cajal Program funded by the Ministerio de Ciencia y Tecnología (Spain).

References

[1] C. Chesñevar and A. Maguitman. ArgueNet: An Argument-Based Recommender System for Solving Web Search Queries. In *Proc. of Intl. IEEE Conference on Intelligent Systems IS-2004 (to appear)*, June 2004.

[2] Carlos Iván Chesñevar, Ana Maguitman, and Ronald Loui. Logical Models of Argument. *ACM Computing Surveys*, 32(4):337–383, December 2000.

[3] R. Frakes, W.; Baeza-Yates. *Information Retrieval. Data Structures & Algorithms*. Prentice Hall, 1992.

[4] Alejandro J. García and Guillermo R. Simari. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming*, 4(1):95–138, 2004.

[5] Sergio Alejandro Gómez and Carlos Iván Chesñevar. Combining Argumentation and Clustering Techniques in Pattern Classification Problems. In *Proc. of the IX Argentinian Conference in Computer Science (CACIC 2003)*, pages 601–612, 2003.

[6] Sergio Alejandro Gómez and Carlos Iván Chesñevar. Integrating Defeasible Argumentation and Machine Learning Techniques: A Preliminary Report. In *Proc. of the V Workshop of Researchers in Computer Science (WICC 2003)*, pages 320–324, 2003.

[7] Sergio Alejandro Gómez and Carlos Iván Chesñevar. A Hybrid Approach to Pattern Classification Using Neural Networks and Defeasible Argumentation. In The 17th International FLAIRS Conference*, Palm Beach, Florida,USA*, May 2004.

[8] Sergio Alejandro Gómez and Carlos Iván Chesñevar. Integrating Defeasible Argumentation with Fuzzy ART Neural Networks for Pattern Classification. In *Journal of Computer Science and Technology*, volume 4(1), April 2004.

[9] Guillermo R. Simari and Ronald P. Loui. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence*, 53:125–157, 1992.

[10] David Skapura. *Building Neural Networks.* ACM Press, Addison-Wesley, 1996.