SEARCH AND CLASSIFICATION OF "INTERESTING" BUSINESS APPLICATIONS IN THE WORLD WIDE WEB USING A NEURAL NETWORK APPROACH

Karl Kurbel, Kirti Singh, Frank Teuteberg
Europe University Viadrina Frankfurt (Oder), Germany

Abstract

A database of business Internet applications developed at Europe University Viadrina is in the process of being filled with "interesting" WWW applications. As the number of WWW sites is huge and still growing fast, the question is how to find the right applications for the database. In this paper, a neural network approach is proposed to automate the process of searching and selecting candidate applications. 23 configurations of neural networks have been tested: 15 versions of the multi-layer perceptron, four generalized feed-forward networks and four modular networks. Results from training and testing those networks are presented and discussed.

Introduction

A database of business Internet applications (Internet Database, IDB for short) has been developed at Europe University Viadrina in Frankfurt (Oder), Germany. The purpose of IDB is to document the state-of-the-art of business Internet use, with a focus on business-to-business applications. The database was described before in (Kurbel, 1997).

The number and the diversity of WWW applications are huge and still growing fast. As we are primarily interested in innovative solutions for the database, the question is how to find those solutions. One way is, of course, "manual" search, i.e. using a keyword-based search engine (e.g. Alta Vista) or going through company listings in the WWW ("yellow pages", business servers, etc.) and analyzing the respective WWW offers. This is obviously a tiring and time-consuming task. Automation of the search process is desirable.

The term WWW offer is used to describe the general fact that a company offers specific information for business partners via the Internet. The approach proposed in this paper is to have an automated process pre-select WWW offers which might be of interest for the database. The final decision is made by a human. The pre-selection step is based on a neural network approach. It reduces the information overload in the Web by categorizing WWW offers. For this purpose, 15 versions of the multi-layer perceptron with error back-propagation (Wassermann, 1989), four versions of generalized feed-forward networks and four versions of modular networks were trained and tested.

Since the focus of the database is on business-to-business, consumer-oriented WWW offers are considered "not interesting". In the business-to-business field, "interesting" applications are applications where business deals are substantially based on Internet use. More precisely, two classification schemes presented before are employed to distinguish between the two groups of "interesting" and "not interesting" WWW offers. In (Kurbel, 1997), business-to-business applications were categorized under two points of view: a) direct communication between business partners, and b) communication with or via information exchanges.

For the first group (direct communication), six main categories were defined:

1) Providing information
2) Providing information plus contact offer
3) Starting a transaction via Internet
4) Starting and completing a transaction via Internet
5) Business-process interfaces via Internet
6) Inter-business cooperation via Internet

Categories 1) and 2) are typically the ones where a company brochure is put in the Web but the reader cannot do much more than just read it. WWW offers of those categories are declared "not interesting" whereas offers of categories 3) to 6) have the potential of being "interesting" for IDB.

For the second group (information exchanges), four categories were proposed (Kurbel, 1997):

1) Simple catalog of firms and/or products
2) Catalog with search option
3) Product exchange
4) Electronic market

Here the categories 1) and 2) are regarded as the not interesting ones. Categories 3) and 4) contain those offers where the mediator operating the information exchange gives automated support for transactions between business partners. Those categories are considered "interesting".

In the next section, the neural networks used for the classification task are introduced. The subsequent section then describes the process of automated search and the classification approach. Afterwards, the specific configurations of the networks are presented and experimental results are discussed. A summary and an outlook to future improvements are given in the concluding section.

Neural Networks for Classification

Neural networks are attractive for classification problems because they are capable to learn from noisy data and to generalize (Bishop, 1995). The first neural network model (perceptron) was developed by Rosenblatt in the late 1950's (Wassermann, 1989; Rich & Knight, 1991). Since then, several other models have been proposed. Examples are: generalized feed-forward networks, radial basis function networks, the Hopfield model, the multi-layer perceptron, modular networks, etc. These models differ in their architectures and in the way they learn and behave, so they are suitable for different types of problems. In this paper, the multi-layer perceptron, modular neural networks and generalized feed-forward networks will be considered.

The multi-layer perceptron (MLP) is the basic model of the work reported here. It has been applied to classification and prediction problems, for example in time series forecasting, stock market prediction, weather prediction, and pattern recognition (Bishop, 1995). A simple MLP consists of three layers: an input layer, a hidden layer and an output layer. The input layer contains a number of elements which pass weighted inputs to the neurons of the hidden layer, according to the connection weights. Inputs to the neurons in our problem are features of WWW applications, e.g. a term like "online order" or a multimedia property (see below). The neurons of the hidden layer process their inputs and propagate their outputs to the third layer. This is the output layer which produces the response of the network. Outputs in our case are the categories "interesting" and "not interesting". The number of hidden layers may be more than one. It will be varied in different configurations of the MLP described below.

MLP's with two or more hidden layers were found efficient for static pattern classification before (Bishop, 1995). They are capable of approximating any input/output map. A mathematical proof may be given that such networks can solve any classification problem if the number of processing elements in each layer and the training time are not constrained (Khanna, 1990).

Figure 1 illustrates the steps of using a simple MLP for classification of business Internet applications into the categories "not interesting" and "interesting". Before the MLP can by applied to a particular classification task it has to be trained. For this purpose, a set of data is given as input to the network, and the weights of the neural connections are adjusted in a way that the output of the network approximates the desired output. To set the weights, the mean squared error (MSE) is computed. The MSE is the sum of the squared differences between the desired output and the actual output of the output neurons averaged over all training exemplars. A small value, close to zero, indicates that the network has learned well and is suited for the classification problem.

The *modular neural network* is a special type of neural model which uses multiple MLP's. Modular networks process their inputs in parallel MLP's and then recombine the results. This approach tends to foster functional specialization in the modules. In contrast to normal MLP's, modular networks do not have full interconnectivity between their layers and thus require a smaller number of

weights for a network of the same size. In this way, the training time can be shortened and the number of training exemplars can be reduced.

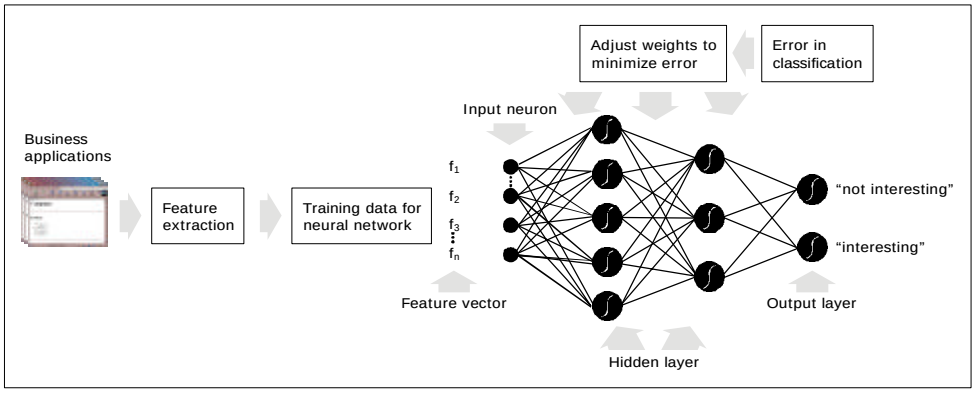

Figure 1: MLP neural network for classification

Generalized feed-forward networks are another variant of the MLP. Here the connections are allowed to cross one or more layers. For example, the output of a neuron in the first hidden layer can be given directly to the output layer, instead of being handed over as input to the next hidden layer.

Overview of the System for Neural Network Training and Classification

227 WWW offers from IDB were classified manually into "interesting" and "not interesting" as described in section 1 and downloaded with the help of an offline browser. A parser, developed in Java, extracts meaningful features for representation of the business applications. These features are coded as input patterns for a neural network. The network is then trained with the input patterns. This means that the connection weights are adjusted in a way that the networks should give the same categories as output as the ones assigned by hand before. In our work, 170 WWW offers are used for training and 57 for validation.

In the classification phase, new applications are searched in the Web. For this purpose, a meta-search tool (WebSeeker) is employed (Teuteberg, 1997; Caceres, 1997). A meta-search tool combines several search engines in a single search. WebSeeker is capable of querying up to 100 search engines simultaneously (including Yahoo, Lycos, Excite, Alta Vista and WebCrawler), indexing the results of any of those engines, and downloading the corresponding pages.

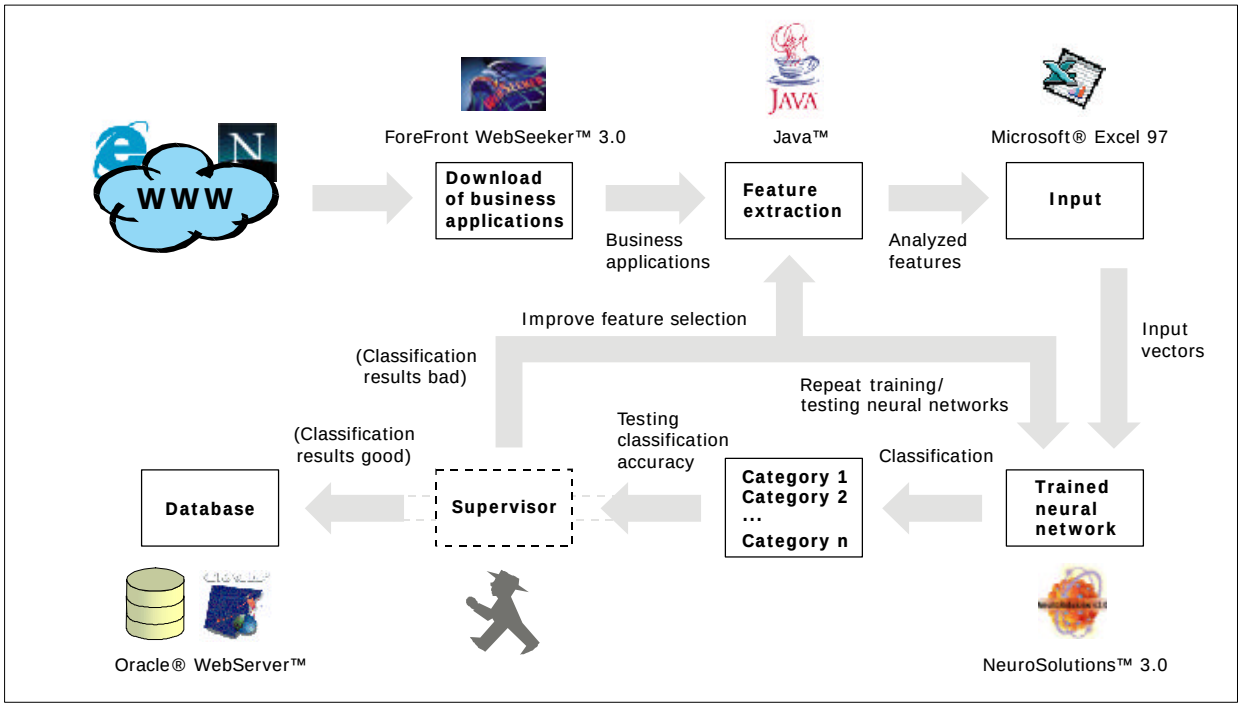

Figure 2: Components of the classification system

The Java parser is used to extract the important features of the downloaded WWW offers. The trained neural network then maps the vector of extracted input features to one of the categories "not interesting" and "interesting". Finally, the new candidate business applications pre-selected in this way can be checked by the IDB administrator and recorded if appropriate.

The process of searching, extracting, and classifying business applications requires support by software tools. An offline browser (Anawave WebSnake Version 1.23) is used to download WWW offers to be used for training the networks. This software is able to follow URL's and download the contents (HTML pages) to the local disk. If a downloaded page contains further links, the process may be continued down to a level specified by the user. The pages are then analyzed locally. For this purpose, the features and the terms used in the pages are extracted by the parser. This program creates an Excel table as input for the neural networks.

For implementation of the networks, a commercial package called NeuroSolutions was used. This software lets users create their own neural networks. It provides several options regarding, for example, the learning algorithm, the number of neurons, the number of hidden layers, and other network parameters. The software can be run within Excel and thus make use of Excel's power and facilities for interpreting the results. In the classification phase, the software tools are mostly the same as in the training phase except for downloading the Web pages. Since *new* pages, i.e. pages whose URL's are not known, have to be downloaded, the offline browser (WebSnake) cannot be used. Instead, a meta-search engine (ForeFront WebSeeker Version 3.0) is applied. This software searches the Web as outlined above and downloads the respective pages (see figure 2).

Feature Extraction, Feature Selection and Coding

A crucial point in using neural networks for classification is to choose the right set of features. This means that the features should represent a business application well. The problem is to select the best subset from N features to reduce the dimensionality of the feature vectors (Dash & Liu, 1997).

Feature type	Feature description	Coding
Keywords in meta tags	Keywords in meta tags are provided by companies if they want robots to index their applications automatically (so they might describe an application well).	Number of occurence and location (hyperlink v_i=1; meta tag v_i=0.75; title tag v_i=0.5; body tag v_i=0.25, v_i=0 if feature is absent).
Keywords as hyperlinks	Keyword points to another page or place (e.g. keyword "order form" as a hyperlink to an order page).	
Keywords in title tags	Keyword was put in the title tag by the author of that page (indicates that the keyword was considered important).	
Keywords in body tags	The body tag contains all information displayed on screen by a browser. Keywords in the body tag are part of the information visible for the user.	
Structural features	Hyperlinks, frames, internal/external links, size of downloaded business applications.	Number of internal/external links; size of business application in KB. If structural feature is present v_i=1, else v_i=0.
Elements of interactivity	Elements which indicate a higher degree of interactivity (e.g. cgi/bin feature, certain Javascript elements (e.g. onSelect).	Feature is present (v_i=1) or absent (v_i=0).
Technological features	Standards, scripts and programming languages used.	If application is realized with a specific technology (e.g. Javascript), this feature is coded with v_i=1, otherwise v_i=0.
Multimedia features	Audio and video files are embedded in WWW offer.	Feature is present (v_i=1) or absent (v_i=0).

Table 1: Extracted and selected features

We use not only keywords as input features but also features like the size and structure of a WWW site, the numbers of internal and external links and where the links go, and the location of a keyword (in meta tag, title tag, body tag, etc.). Furthermore, the technologies employed (e.g. Java, Javascript, VRML, Shockwave, multimedia and audio files) as well as typical elements of interaction (e.g. CGI scripts or certain Javascript keywords) are considered relevant and are also extracted as input features. Accuracy of classification can be expected to improve if advanced features like the ones mentioned, beyond just keywords, are extracted.

To limit the times for download, feature extraction, and training, links are followed down three levels. The offline browser starts with a homepage (e.g. http://www.marketing.com) and then follows all links down to depth 3 (e.g. http://www.marketing.com/product-catalog/product1.html). For our training and testing data base, 227 applications (122 MB, 12,756 HTML files) were downloaded in this way. Table 1 gives an overview of the features used to train and test the networks.

Coding: In the vector space information retrieval paradigm, each document has a vector V where each element v_i is the weight of a word or a feature (Salton & McGill, 1983). In our work, we tried out various ways of coding the weights v_i. Binary coding indicates whether a feature is present ($v_i = 1$) or absent ($v_i = 0$) in a WWW offer. Another type indicates the number of times a feature (e.g. a keyword) is present. The third type of coding exploits meta tags and the specific HTML syntax. A typical feature like "order form" is coded by 0 if it is absent, 0.25 if it is in the body tag, 0.5 if it is in the title tag, 0.75 if it occurs in the meta tag ("content" or "keyword"), and 1 if it is a hyperlink to another page within the same WWW offer (where perhaps an online order facility is provided).

The features outlined in table 1 are coded and represented in Excel spreadsheets. Figure 3 shows an excerpt from such a spreadsheet. Each row describes one WWW offer. The columns contain the extracted features (520 features).

	A	B	C	D	E	F		HT	HU	HV	HW	HX	HY	HZ
1	http:	access	banking	internet	mail	password		size	.gif	.zip	button	frame	javascript	output
2	http://www.1stfed.com/	0.25	0.25	0.25	0.25	0		107034	1	0	1	0	0	0
3	http://www.abfs.com/entrace.htm	0	0	0	0.25	0		8483	1	0	0	0	0	1
4	http://www.airborne-express.com	0	0	0.25	0.25	0		86236	1	0	1	0	0	1
5	http://www.aircanada.ca/cargo/english/n	0	0	0	0	0		10280	1	0	0	0	0	1
6	http://www.baglobal.com/tracking/track	0	0	0	0	0		11744	1	0	0	0	0	1
7	http://www.bbi.co.uk/index.html	0	0	0.5	0	0		13494	1	0	0	0	0	0
8	http://www.bmo.com/	0	0.25	0	0.25	0		26026	1	0	1	1	0	0
9	http://www.canadatrust.com	0.5	0.5	0.25	0.25	0.25		670702	1	0	0	0	0	0
10	http://www.ccx.com/ccxi/home.html	0	0	1	0.25	0		9560	1	0	0	0	0	1
11	http://www.cdworld.com	0.25	0	0.5	0.25	0		94783	1	0	1	0	0	0
12	http://www.centralres.com/	0.25	0	0.25	0.25	0		57994	1	0	1	1	0	1
13	http://www.colonialbank.com/	0.25	0.25	0	0.25	0		138058	1	0	1	0	1	0
14	http://www.commerce.net/	0.25	0	0.25	0.25	0		258535	1	1	0	1	0	1
15	http://www.dealernet.com	0.75	0	0.25	0.5	0		81344	1	0	1	0	0	1
16	http://www.dhl.com/track/track.html	0	0	0	0	0		6196	1	0	0	0	0	1
17	http://www.eupc.com/eupc-e/index.htm	0	0	0.25	0.25	0		3068	1	0	1	1	0	1
18	http://www.export.nl	0.75	0	0.25	1	0		530022	1	0	1	0	0	1
19	http://www.fedex.com	0	0	0	0	0		21905	1	0	0	0	0	1
20	http://www.firstunion.com	0.25	0.75	0.25	0.25	0		216918	1	0	1	1	0	1
21	http://www.flavorohio.com	0	0	0	0.25	0		55260	1	0	0	0	0	0
22	http://www.ichat.com	0	0	0.25	0.25	0		62226	1	0	0	1	1	0
23	http://www.inbb.com/	0.25	0	0.5	0.25	0		15508	1	0	0	1	0	1
24	http://www.intermesse.com	0	0	0	0	0		6930	0	0	0	1	0	1
25	http://www.internet.net	0.25	0	0.25	0.25	0		107263	1	0	1	0	0	0

Figure 3: Excel spreadsheet with extracted features

Experimental Results

The following types of neural networks were tested: 15 versions of the multi-layer perceptron, four generalized feed-forward networks and four modular neural networks. Interpretation of the output neurons as "interesting" and "not interesting" WWW offers was based on a simple distinction of output values: An output ≥ 0.5 is considered "interesting", an output < 0.5 "not interesting". More sophisticated interpretation rules will be applied in the future.

Table 2 shows the best training results from two multi-layer perceptrons, two generalized feed-forward networks, and two modular neural networks, with regard to the mean squared error and based on the training set of 227 business applications. The first column of the table characterizes the type of neural model and the parameters: numbers of epochs, hidden layers (HL), and processing ele-

ments (PE's) in each hidden layer. MSE stands for the mean squared error in the outputs of the training phase.

Network configuration (epochs/HL/PE's)	Mean squared error (MSE)
1) MLP (100/3/50)	0.002510512
2) MLP (100/2/50)	0.005332240
3) Modular network (100/2/50)	0.051718704
4) Modular network (100/3/50)	0.066576041
5) Generalized feed-forward network (100/2/50)	0.144806534
6) Generalized feed-forward network (100/3/150)	0.167463064

Table 2: Mean squared error (MSE) – classification "not interesting"/"interesting"

The MLP with 3 hidden layers and 50 PE's in each hidden layer turned out to be the winner. When this network was run to classify the training set, all 227 applications were classified correctly.

Subsequently, the data set of 227 WWW offers was randomly divided into four subsets of 170 applications (75 %) each for training and 57 applications (25 %) each for validation. The best configurations of each network type from above – i.e. MLP (100/3/50), modular network (100/2/50), generalized feed-forward network (100/2/50) – were then trained and evaluated with the four pairs of subsets. The best classification results of the four runs and the worst results [in brackets] are summarized in table 3.

Network configuration (epochs/HL/PE's)	MSE	Percentage of correctly classified test data
MLP (100/3/50)	0.008454259 [0.014019439]	87.72 % [82.46 %]
Modular network (100/2/50)	0.025483903 [0.013499024]	84.41 % [80.70 %]
Generalized feed-forward network (100/2/50)	0.217074692 [0.389487624]	70.18 % [52.63 %]

Table 3: MSE and classification results from four training and test sets

It should be noted that the MSE varies with the training data and is not fully correlated with accuracy of classification. For example, the worst classification results (80.7 %) of the modular network were obtained for a training subset where the MSE was only 0.0135 while the best classification (84.41 %) produced by that network was for a subset where the MSE in training had been almost twice as big (0.0255). Obviously some training sets are easier to learn than others but this does not mean automatically that the network will produce equally good results when exposed to new cases. The winning network, *MLP (100/3/50)*, on average classified *84.75 %* of the data in the four test sets correctly. These intermediate results are likely to improve with further tuning in the future.

Summary and Future Work

In this paper, an approach for searching and pre-selecting WWW offers to be considered for the Internet Database (IDB) was presented. Our first aim was to classify WWW offers into categories "interesting" and "not interesting". For this purpose, several multi-layer perceptrons, generalized feed-forward networks and modular networks were tried. All networks performed quite well. The winner, an MLP with 3 hidden layers and 50 PE's per hidden layer, exhibited reasonable generalization capability as well. When exposed to new data sets it classified 84.75 % on average correctly.

In order to improve these current results we intend to use larger data sets to train the network. Experiments will also be made with various feature selection methods from the literature (Dash & Liu,

1997), since our system is sensitive to the number of relevant features. Different types of neural networks with their specific capabilities will be combined for this purpose.

Principal component analysis (PCA) networks may help to reduce the feature space. Those networks combine unsupervised and supervised learning in one topology (Bishop, 1995). PCA is an unsupervised linear procedure that finds a set of uncorrelated features from the input vectors. By employing PCA the number of features for the MLP can be reduced significantly, thus the amount of training data and the training times will also decrease.

Furthermore, experiments with self-organizing feature maps (Kohonen, 1995) will be done to restrict the space to representative features. The outcome of the self-organization process can be used as input to a supervised MLP.

The simple distinction between "interesting" and "not interesting" WWW offers as underlying our first approach will be successively refined, e.g. into categories "not interesting", "less interesting", "interesting" and "very interesting". A long-term goal is to classify WWW applications automatically into the six categories of direct business-to-business communication and the four categories of communication via information exchanges as mentioned above. This is, however, very difficult to accomplish.

Another categorization of WWW offers in IDB uses keywords. Those keywords characterize certain application scenarios, like "order tracking", "electronic auction", "business server", etc. In order to fill the database with specific WWW offers for such scenarios, it is necessary to find appropriate offers in the Web first. The three neural network types described above have been applied for this purpose as well. As the preliminary results are promising, we will tune and use the networks for scenario classification in the future, too.

References

Bishop, C. M.: Neural Networks for Pattern Recognition; Oxford 1995.

Caceres Urrutia, F. X.: WebSeeker – A Critical Review; http://www.fis.utoronto.ca/courses/LIS/2108/1997f/assignments/internettools/webseeker/webseeker.html; February 26, 1998.

Dash, M., Liu, H.: Feature Selection for Classification; http://www-east.elsevier.com/ida/browse/0103/ida00013/article.htm; February 26, 1998.

Khanna, T.: Foundations of Neural Networks; New York 1990.

Kohonen, T.: Self-Organizing Maps; Berlin 1995.

Kurbel, K.: Design and Implementation of a Database with Innovative Business-to-business Internet Applications; in: Forcht, K. (Ed.), 1997 IACIS Refereed Proceedings "Expanding Information Horizons in a Global Society", St. Louis, Missouri, Oct. 2-4, 1997, pp. 1-6.

Kurbel, K., Teuteberg, F.: The Current State of Business Internet Use: Results from an Empirical Survey of German Companies; in: Proceedings of European Conference on Information Systems 1998, Aix-en-Provence, France, June 4-6, 1998, pp. 542-556.

Rich, E., Knight, K.: Artificial Intelligence; New York 1991.

Salton, G., McGill, M. J.: Introduction to Modern Information Retrieval; New York 1983.

Teuteberg, F.: Effektives Suchen im World Wide Web: Suchdienste und Suchmethoden; Wirtschaftsinformatik 39 (1997) 4, pp. 373-383.

Wasserman, P. D.: Neural Computation: Theory and Practice; New York 1989.