22

Self-organizing Maps in Web Mining and Semantic Web

Emil Şt. Chifu and Ioan Alfred Leţia
Technical University of Cluj-Napoca
Romania

1. Introduction

The nature inspired approaches represent a new trend in computer science in general and in the Semantic Web, due to their scalability and robustness. Neural networks represent one category of nature inspired solutions. The self-organizing map (SOM) is a very popular unsupervised neural network model (Kohonen, et al., 2000). It is a data mining and visualization method for complex high dimensional data sets.

In the first part of the chapter, we present how the SOM model can be applied in Web mining, by giving sets of documents as input data space for SOM. The result of applying SOM on a set of documents is a map of documents, which is organized in a meaningful manner so that documents with similar content appear at nearby locations on the two-dimensional map display. From the information retrieval point of view, our implemented SOM-based system creates document maps that are readily organized for browsing. A document map also clusters the data, resulting in an approximate model of the data distribution in the high dimensional document space. Some experimental results are included, where a couple of meaningful clusters have been discovered by our system in a subset of the "20 newsgroups" data set (Lang, K., 1995). The clustering capability of our system allows users to find out quickly what is new in a Web site of interest by comparing the clusters obtained from the site at different moments in time.

In the rest of the chapter, we focus on how a more complex SOM based unsupervised neural network model is used for enriching a domain ontology. Building complete and reliable domain ontologies is the basis for the success of the Semantic Web. The ontology enrichment process consists in the addition of new concepts which will be attached as hyponyms for the existent nodes of the ontology (Pekar and Staab, 2002). The names of the new concepts are terms represented linguistically by common noun phrases. The enrichment process can also add new instances to existent concepts of the ontology. In this case, the process is also known in the literature as ontology population or named entity classification, where the named entities are represented linguistically by proper names of people, organizations, locations etc. (Cimiano and Völker, 2005). In both cases, the process is algorithmically the same, the only difference being the grammatical category of the linguistic entities to be classified: common noun phrases representing terms for new concepts to be added or proper noun phrases representing named entities, i.e. new instances for the existent

concepts. The noun phrases representing terms and named entities are extracted from a domain text corpus by a text mining process. For every noun phrase, the mining acquires a vector that encodes contextual content information, in a distributional vector space. The enrichment behaves like a classification of the terms or named entities into the taxonomy of the given ontology, based on a similarity metric in the distributional vector space.

The growing hierarchical self-organizing map (GHSOM) model consists of a set of SOM maps arranged as nodes in a hierarchy and it is able to discover hierarchical clusters (Dittenbach et al., 2002). The SOM's in the nodes can grow horizontally during the training by inserting either one more row or one more column of neurons. The SOM's in the nodes can also grow vertically during the training, by giving rise to successor nodes. Like the classical Kohonen SOM model, GHSOM is an unsupervised neural network. Unsupervised hierarchical neural models in general start the growing of a dynamic tree-like topology from a single initial node. We propose a new neural network model, called Enrich-GHSOM, as an extension of the GHSOM system, which allows the growing to start from an initial tree. The growth of the hierarchy proceeds along with the predefined paths of the given hierarchy. Consequently, our model allows a classification of the data items into an existing taxonomic structure that plays the role of an initial state for the tree-like neural network model. In order to apply our model in ontology enrichment, the taxonomy that is subject to enrichment is given as the initial state of the hierarchical self-organizing map. So, an essentially symbolic knowledge structure – taxonomic tree – is converted into a neural representation as an initial state of the hierarchical self-organizing map. The actual taxonomy enrichment takes place via an unsupervised training of the neural network by exposing the initialized hierarchical self-organizing map to the vector representation of the terms extracted from the domain corpus. A reverse, neural-symbolic translation is done after this enrichment process. This is the knowledge extraction step, and the output is the final enriched taxonomy. Our taxonomy enrichment framework is a hybrid one, as it has to deal with neural-symbolic integration. The neural-symbolic translations in both directions have been naturally achieved, since our framework merely operates upon the taxonomic structure of the ontology, which is in agreement with the hierarchical structure of the self-organizing neural network.

The ontology enrichment experiments that will be presented are in the "Lonely Planet" tourism domain. The taxonomies, the corpus, and the named entities are the ones proposed in the PASCAL ontology learning and population challenge (Grobelnik et al., 2006). The evaluation of the enrichment is based on cross-validation and on gold standard ontologies. Our experimental results prove that the quality of the ontology enrichment is considerably improved by using our semantics based vector representations for the classified (newly added) terms, like the document category histograms and the document frequency times inverse term frequency (DF-ITF) weighting scheme.

2. Self-organizing Maps

The self-organizing maps have been created by Teuvo Kohonen as a particular kind of neural networks (Kohonen, et al., 2000). There are multiple views on SOM; the different definitions are the following. SOM is a model of specific aspects of biological neural nets (the ordered "maps" in the cortex). SOM is a model of unsupervised machine learning and an adaptive knowledge representation scheme. SOM is a tool for statistical analysis and

visualization: it is both a projection method which maps a high dimensional data space into a lower dimensional one and a clustering method so that similar data samples – represented as vectors of numerical attribute values – tend to be mapped on nearby neurons. The resulting lower dimensional output space is a two-dimensional grid of arrays (the SOM map) which visualizes important relationships among the data, – which are latent in the input data set – in an easily understandable way. This dimensionality reduction maintains the topology of the input vectors, i.e. inputs that are close to each other – in other words, similar – in the input space are also close to each other in one of the clusters of the map.
In short, SOM is a data mining and visualization method for complex high dimensional data sets. Even though there are no explicit clusters in the input data set, important relationships are nevertheless latent in the data. SOM can discover and illustrate these latent structures of an arbitrary data set. SOM can describe different aspects of a phenomenon in any domain, provided that the data in the domain can be represented by vectors of numerical attributes.
The map learns by a self-organization process. No a priori knowledge about the membership of any input data item (vector) in a particular class or about the number of such classes is available. Hence, the training proceeds with unlabeled input data like any unsupervised learning. The clusters (classes) are instead discovered and described with gradually detected characteristics during the training process.
The map consists of a regular two-dimensional (rectangular) grid of processing units – the neurons. Each unit has an associated model of some multidimensional observation, represented as a vector of attribute values in a domain. SOM learning is an unsupervised regression process which consumes at every iteration one available observation represented as a vector of values for the attributes in a given domain. The role of a learned map is to represent all the available observations with optimal accuracy by using a restricted set of model vectors associated to the map units.

2.1 The Learning Algorithm

The initial values for the model vectors – also referred to as reference vectors or weight vectors – of the map units can either be chosen depending on the problem domain or they can be taken randomly. Each iteration of the learning algorithm processes one input (training) vector (one sample) $\mathbf{x}(t)$ as follows. Like usually for unsupervised neural networks, some form of a competitive learning takes place: the winner unit index c, which best matches the current input vector, is identified as the unit where the model vector is the most similar to the current input vector in some metric, e.g. Euclidean:

$$\| \mathbf{x}(t) - \mathbf{m}_c(t) \| \leq \| \mathbf{x}(t) - \mathbf{m}_i(t) \| , \quad (1)$$

for any unit index i. Then all the model vectors or a subset of them that correspond to units centered around the winner unit c – i.e. units in the neighbourhood area of c –, including the winner itself, are adjusted in the direction of the input vector, as follows:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \mathbf{h}_{ci}(t) * [\mathbf{x}(t) - \mathbf{m}_i(t)] , \quad (2)$$

where h_{ci} is the neighbourhood function, which is a decreasing function on the distance between the i-th and c-th units on the map grid, and whose maximum value corresponds to

$i = c$. In practice, the neighbourhood area is chosen to be wide in the beginning of the learning process, and both its width and height decrease during learning. The updating in (2) forms a globally ordered map in the process of learning.

A map unit has six immediate neighbours in a hexagonal map topology, which is usually the preferred topology. This is only a hexagonal lattice type of the two-dimensional array (grid) of neurons, so the SOM map continues to be a planar rectangle. The hexagonal neighbourhood topology effect is gained by shifting correspondingly rows number 1, 3, 5, … of the rectangular map to the right and keeping rows 0, 2, 4, … untouched. A rectangular topology corresponds to a rectangular lattice, and a map unit has only four immediate neighbours. Consequently, the number of neighbour units affected during the learning is only four as compared to six in the hexagonal topology.

After the training of a map, its reference vectors have converged to stationary values and the result is a topology-preserved map. Similar reference vectors become close to each other, and dissimilar ones become far from each other on the map. Moreover, two input data items which are close to each other in the input data space are mapped onto the same or neighbouring neurons on the map. Each neuron together with its own reference vector represents similar data items of the input space, and a set of neighbouring neurons with similar reference vectors creates a cluster.

2.2 Cluster Visualization

A subset of data items which are close to each other in a high dimensional input data space – and thus defines a cluster in the input space – are arranged to a map area consisting of neurons close to each other also in the two-dimensional SOM display. As a consequence, the problem of discovering a cluster in a high dimensional data set with the help of the self-organizing maps reduces to the problem of discovering the map area whose neurons to contain all the data in the cluster. Actually, we have to find the boundaries of the map cluster. Finding the boundaries of a SOM map cluster is based on applying the unified-distance matrix (U-matrix) algorithm on a SOM map (Ultsch, 1993).

U-matrix visualizes the map in grey-levels, in order to express how similar or dissimilar adjacent neurons are (Wilppu, 1997; Hautaniemi, et. al., 2003). In a hexagonal self-organizing map topology, six hexagons (extra neurons) around each neuron separate geometrically the neuron from its six immediate neighbours and show its similarity with each of them. The lighter a separating hexagon, the bigger the similarity of the reference vectors of the two separated neurons, and the darker the hexagon, the bigger the dissimilarity of the reference vectors. This way, SOM map clusters can be discovered visually as “valleys” or “depressions” (light areas) separated by “hills” (dark areas or borders). Moreover, the higher (i.e. darker) a hill separating two clusters, the more dissimilar the clusters in the multidimensional input data space.

In (Ultsch, 1993), an older (in fact the original) version of the U-matrix algorithm is used, by calculating at each map unit the sum of the distances of the reference vector of that neuron to the reference vectors of the immediate neighbouring neurons.

3. Web Mining with Self-organizing Maps

Applying SOM on natural language data means doing data mining on text data, for instance Web documents (Lagus, 2000). The role of SOM is to cluster numerical vectors given at input and to produce a topologically ordered result. The main problem of SOM as applied to natural language is the need to handle essentially symbolic input such as words. If we want SOM to have words as input then SOM will arrange the words into word categories. But what about the input (training) vector associated to each input word? What should be the vector components, i.e. the attributes of a word? Similarity in word appearance is not related to the word meaning, e.g. "window", "glass", "widow".

We have chosen to classify words by SOM, creating thus word category maps. The attributes of the words in our experiments were the count of the word occurrences in each document in a collection of documents. Consequently, we have chosen to represent the meaning of each word as related to the meanings of text passages (documents) containing the word and, symmetrically, the semantic content of a document as a bag-of-words style function of the meanings of the words in the document. The lexical-semantic explanation of this contextual usage meaning of words is that the set of all the word contexts in which a given word does and does not occur provides a set of mutual constraints that captures the similarity of meaning of words and passages (i.e. documents, contexts) to each other. The measures of word-word, word-passage and passage-passage relations are well correlated with several cognitive phenomena involving semantic similarity and association (Landauer, et. al., 1998). The meaning of semantically similar words is expressed by similar vectors.

After training a SOM on all the words in a collection of documents – where the vectorial coding of words represents the contextual usage –, the result self-organizing map groups the words in semantic categories. There are also other possibilities to code words, which lead to grammatical or semantic word categories (Honkela, 1997; Kohonen, et.al., 1996; Kohonen, et. al., 2000).

3.1 System Architecture

The architecture of our system is based on two self-organizing maps. The first one creates a semantically ordered spread of all the word forms in a large collection of Web documents. This is also called the map of word categories or level 1 SOM. The second SOM (called document map or level 2 SOM) represents a semantically ordered spread of all the documents in the collection, where the documents are codified as vectors that are histograms of word categories. The word categories are the ones as already induced into the word category map units. For every word category, the histogram representation of a document contains the number of word form occurrences in the document which belong to that word category. This way we have reduced the dimensionality of the document vectors from thousands of components which would correspond to thousands of different word forms in a classical bag-of-words approach. The dimensionality is reduced to around 200 or 300 components which correspond to 200 or 300 different word categories, enough to express the number of different concepts in a shallower or wider domain. Thus the reduced dimensionality removes the noise caused by the variability in word usage; since the number of dimensions is much smaller than the number of word forms, minor differences in terminology will be ignored. Our category based approach is able to solve the terminology problem in information retrieval, i.e. the problem of possibly different terminologies used in

the documents and in a user query for one and the same concept, in other words, the problem of synonymy or near-synonymy.
The aim of our system is to classify the document collection by using the criterion of semantic similarity. Hence the graphical browsing interface of our system is in essence a document map (level 2 SOM).

3.2 System Implementation

The system is written in C and bash script. We have used the LEX software package (Lesk, and Schmidt, 1975) for implementing the preprocessing module, which reads and counts the word occurrences in all the documents in a collection, by ignoring all the HTML tags. The preprocessing module also ignores 450 common words, i.e. English words having no semantic load. These words have been taken from the information retrieval software package GTP (Giles, et. al., 2003). Finally, the preprocessing also means a stemming phase that uses a morphological analyzer for English, which is part of the GATE system (Cunningham, et. al., 2002). The stemming is done in order to reduce the number of word forms by keeping only their stem.
The SOM_PAK (Kohonen, et. al., 1996) system is used for the training of all our SOM maps. The result of training the document SOM is a text file containing for every document category a list of document names that belong to that category, i.e. the list of documents managed into the corresponding map unit. The format of this text file is exemplified with seven document categories in Fig. 1, where each row corresponds to a different map unit. The first two integer numbers in each row represent the rectangular coordinates (x and y) of the current unit. The document category name follows the coordinates of the unit and becomes the identification label of the unit. The document category name is given by the name of the first document in the (training) data set that "hit" the unit during the training process. This name occurs as the last in the enumeration of document names in the category, after the colon.
All the seven document categories in Fig. 1 are semantically related as they all contain as documents emails from one and the same newsgroup (*talk.politics.mideast*) in the "20 newsgroups data set" (Lang, 1995). The seven corresponding map units are neighbours on the document map and they constitute together an area or *cluster*. The aggregation of the neurons in this cluster is noticeable from their coordinates and from the hexagonal topology adopted.

3.2.1 Graphical User Interface

The graphical interface has been implemented by using the PHP language. The system creates the graphical interface as an interactive graphical display that is implemented as a dynamically created HTML file. This PHP module reads the text file of document categories (created by the document SOM and exemplified in Fig. 1) and translates this document classification into a dynamical HTML file which is the graphical display of the document map itself. Every map unit is labelled with the associated document category name, which is found out as explained at the beginning of the current section (Section 3.2). A better alternative would be to label a map unit with the most relevant and representative document in the corresponding category, i.e. the name of the document whose vector is closest to the model vector of that unit (Lagus, et.al., 1999). A second label on each map unit

represents the number of documents in the corresponding category. For instance, the map unit for the last document category in Fig. 1, having coordinates 9, 5 on the map, is also labelled 6.

```
 8  3 75381  :  75381
 8  4 75382  :  75369  75382
 9  4 75394  :  75371  75389  75394
10  4 75393  :  75393
11  4 75400  :  75370  75392  75400
 8  5 75395  :  75395
 9  5 75399  :  176854  75366  75387  75388  75390  75399
```

Fig. 1. Example document categories.

The interface allows the navigation on the document map from any Web browser. The aim of this browsing is the retrieval of relevant documents in two steps. Click on a map unit gives access to the index of documents in that unit, which is also a dynamically generated HTML file, containing a list of links, each of which having as text the document name and pointing to the document itself. Then click on a document name in this list allows viewing that document.

3.3 Experimental Evaluation

The experiments reported here take as test data the "20 newsgroups data set" (Lang, 1995). This data set contains 20,000 UseNet news postings having the form of email messages. The 20,000 messages were collected at random from 20 different Netnews newsgroups, 1000 messages from each newsgroup (Lang, 1995). The data set is "labelled", by being already partitioned into twenty categories. This labelling helped us in evaluating the clustering results of the same set of email documents as discovered and visualized by our document SOM. In one of our most successful experiments, we have selected randomly 40 documents from each newsgroup, summing up a total set of 800 message documents. This balanced subset of the original "20 newsgroups" data set has been taken as input data space for our SOM-based system in order to arrive at an email document SOM map.

An important question in this experiment was to choose a size for the SOM map, in order to arrive at a map with the highest degree of visual expressiveness for clustering (Wilppu, 1997). The map size means the total number of neurons of the rectangular grid. For a given data set, different map sizes mean different granularity levels, in terms of the average number of data items to belong to a neuron. If the map is too small, it is too rough and consequently it might hide some important differences that should be detected in order to separate the clusters. This is because too many unsimilar data items could belong to the same neuron. When the map is too big, then it is too detailed and, besides the important differences, the map displays also too small differences, which are often unimportant for

clustering. This is because data items which are very similar could belong to different neurons, when normally we expect them to belong to one single neuron.
We have chosen a map size of 16 (columns) times 12 (rows) considered as suitable for the input data space of 800 data items (800 email documents). This also conforms to the suggestions in (Wilppu, 1997), where a suite of experiments with input data sets of different cardinality and different SOM map sizes is described. Fig. 2 shows the result email document SOM map image, where grey levels occur as an effect of applying the U-matrix algorithm for cluster visualization. The U-matrix algorithm used here is included in the SOM_PAK program package (Kohonen, et. al., 1996), which is part of our system. The algorithm conforms to the description in Section 2.2.

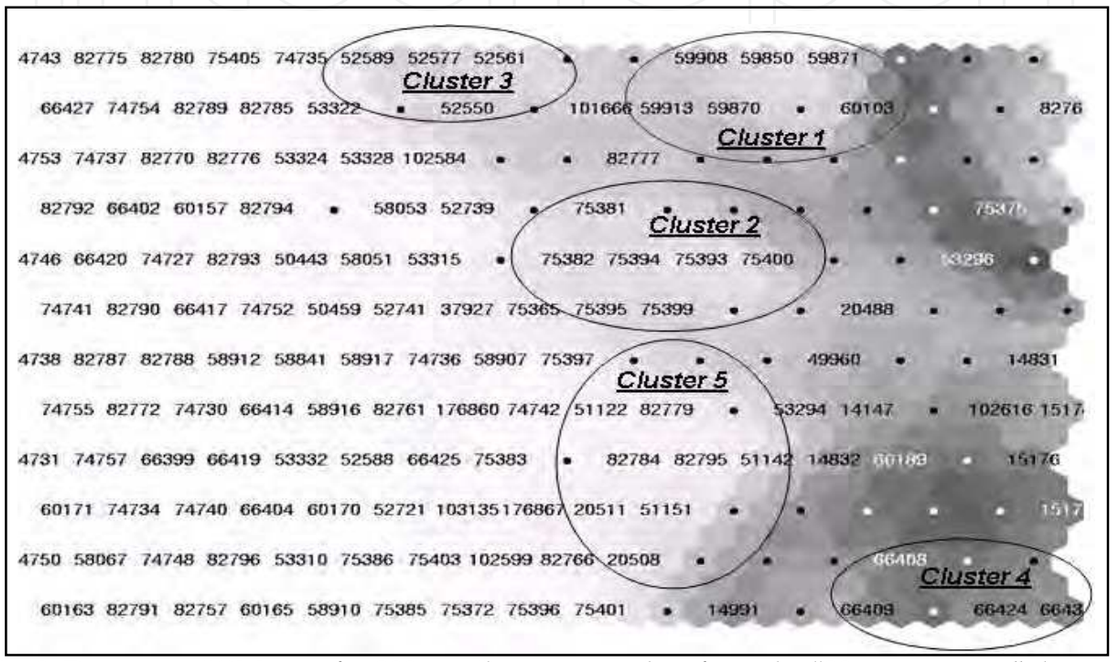

Fig. 2. Document SOM map for 800 email messages taken from the "20 newsgroups" data set

3.3.1 Clustering Results

The document map in Fig. 2 clearly illustrates four clusters discovered by the map. Neurons in Cluster 1 contain 15 email messages, which all belong to the newsgroup *science.space*. Only 15 out of a total of 40 messages in the *science.space* newsgroup in the input space were discovered to belong to Cluster 1. Hence, even if the *accuracy* of this cluster is 100%, its coverage is only 15/40, so 37.5%.
All the 17 documents in Cluster 2 belong to the newsgroup *talk.politics.mideast*, but there are 41 messages in the input data that belong to this newsgroup. Cluster 2 contains seven neurons whose explicit description in terms of email messages grouped as document categories in each neuron is given in Fig. 1. Actually only 40 input messages are "officially" labelled by the authors of the "20 newsgroups" data set to belong to the newsgroup *talk.politics.mideast*. One more message, named 176854, and found out by our map to belong to Cluster 2, has been "abusively" put by the authors into another newsgroup, namely

talk.politics.misc. The header of this email indicates explicitly `Newsgroups: talk.politics.mid-east, misc.headlines, talk.politics.misc`.
Similarly, Cluster 3 contains 12 messages, 11 of them from the newsgroup *rec.sport.hockey*. This cluster is less clearly bordered on the map, because of the semantic overlap with other messages some of them form the related newsgroup *rec.sport.baseball*. In fact, the only message in Cluster 3, which is outside of the expected newsgroup *rec.sport.hockey*, is from the related newsgroup *rec.sport.baseball*. Finally, Cluster 4 on the map represents 11 messages, 10 of them from the newsgroup *comp.windows.x*, and one from the related newsgroup *comp.sys.ibm.pc.hardware*. Table 1 shows the classification quality parameters accuracy and coverage associated with the four clusters.

Cluster No.	**Newsgroup**	**Accuracy** (Correct/Predicted)	**Coverage** (Correct/Actual)
1	*Science.space*	100% (15 / 15)	37.5% (15 / 40)
2	*talk.politics.mideast*	100% (17 / 17)	41.5% (17 / 41)
3	*rec.sport.hockey*	91.5% (11 / 12)	27.5% (11 / 40)
4	*comp.windows.x*	90.9% (10 / 11)	25% (10 / 40)
5	Combination of *talk.religion.misc*, *soc.religion.christian*, *alt.atheism*	80.8% (21 / 26)	17.5% (21 / 120) where 120 = 3*40

Table 1. Classification accuracy and coverage associated with document clusters on Fig. 2

3.3.2 Discussion of Results

There are some more results found out from our document map induced from 800 news messages, and illustrated in Fig. 2. For instance, there is one more cluster, Cluster 5, also mentioned in Table 1, which contains 26 email messages, 21 of them being a mixture of messages from three different newsgroups: *talk.religion.misc*, *soc.religion.christian*, and *alt.atheism*. The first two newsgroups are obviously related to each other, and they are also semantically related with the third, even if this relation sounds more like an antonymy. Similar topics are nevertheless discussed in messages about religion and atheism.
About 85% of the 800 email messages are contained in about the left half of the map, which is completely white, and constitutes a huge cluster. Such a cluster has no clear semantic content, because it contains messages from all the 20 newsgroups, including the messages left out from the five clusters already mentioned. The technical explanation for this phenomenon is that the document SOM map was unable to display semantic differences in this big cluster. The differences in the semantic content of the messages could be too small when the authors of the messages use too few words specific to the domain of the newsgroup or sometimes when they communicate announcements with no bearing with the domain of the newsgroup.
Another explanation for the huge cluster is that the majority of the email messages in the "20 newsgroups" data set are addressed to many different real newsgroups. The more newsgroups a message is addressed to, the more arbitrary its inclusion (by the authors of the "20 newsgroups" data set) in one of the 20 groups, and the fewer semantic differences discernable by our SOM-based system for such messages.
Our clustering results were worse when we didn't ignore the 450 stop words mentioned in Section 3.2, because these words with no semantic load introduced noise that reduced the

capability of displaying semantic differences between email message documents. We have also examined some word category SOM maps. One of our first interesting results with word maps is that, when we didn't ignore the stop words, the word category maps contained isolated (unclustered) neurons representing stop words, which were situated only near the margins of the map. The explanation is that the word map separates the stop words from the other content-rich words, the latter being contained in the interior of the map.

4. Growing Hierarchical Self-organizing Maps

GHSOM is an extension of the Self-organizing Map (SOM, also known as Kohonen map) learning architecture (Kohonen, et al., 2000). Data spaces contain some latent structuring in the form of clusters. SOM maps can discover and illustrate this clustering. However, some *hierarchical structures* are also latent in data sets. To give an interesting example in the present context, a thesaurus is a data space consisting of terms in a language, represented as a lexical database. The main relation between the terms in a thesaurus is the taxonomic relation. However, because of their essentially flat topology, SOM maps have a limited capability to discover and illustrate hierarchical clusters in data sets. A solution for this problem is represented by the *hierarchical SOM maps*.

The growing hierarchical self-organizing map model consists of a set of SOM maps arranged as nodes in a hierarchy and it is able to discover hierarchical clusters (Dittenbach, et. al., 2002). The SOM's in the nodes can grow horizontally during the training by inserting either one more row or one more column of neurons. This happens iteratively until the average data deviation (quantization error) over the neurons in the SOM map decreases under a specified threshold τ_1. For one neuron, the quantization error is the dissimilarity of all the vectors of the data items mapped into the neuron versus the weight vector of the neuron.

The SOM's in the nodes can also grow vertically during the training, by giving rise to successor nodes. Each neuron in the SOM map could be a candidate for expansion into a successor node SOM map (see Fig. 3). The expansion takes place whenever the data deviation on the current neuron is over a threshold τ_2. This sounds like a zoom into the data subspace mapped into the parent neuron, because the successor SOM map is trained merely with data items in that subspace. Further node expansions continue recursively on successor nodes, and the training of the whole GHSOM model finally stops (converges) when both thresholds are satisfied. The training begins with a single-neuron SOM map having the whole input data set mapped into its only neuron. This becomes the root of the final, completely trained GHSOM model.

The thresholds τ_1 and τ_2 control the granularity of the hierarchy learned by GHSOM in terms of depth and branching factor. A low τ_1 with a much lower τ_2 leads to a deep hierarchy with an increased number of neurons into the SOM nodes, and consequently an increased branching factor also. A high τ_1 with a lower τ_2 leads to deep hierarchies with small SOM nodes (with few neurons), and consequently a reduced branching factor corresponding to the reduced number of neurons in SOM nodes. When both thresholds are low and comparable, then the hierarchy is flat with a high branching factor. If both thresholds are high and comparable, then the hierarchy is flat with a low branching factor.

Each level in a learned GHSOM model displays a more detailed clustering of the data space as compared to the parent level. This corresponds to a top-down process of hierarchical clustering of the input data space items.

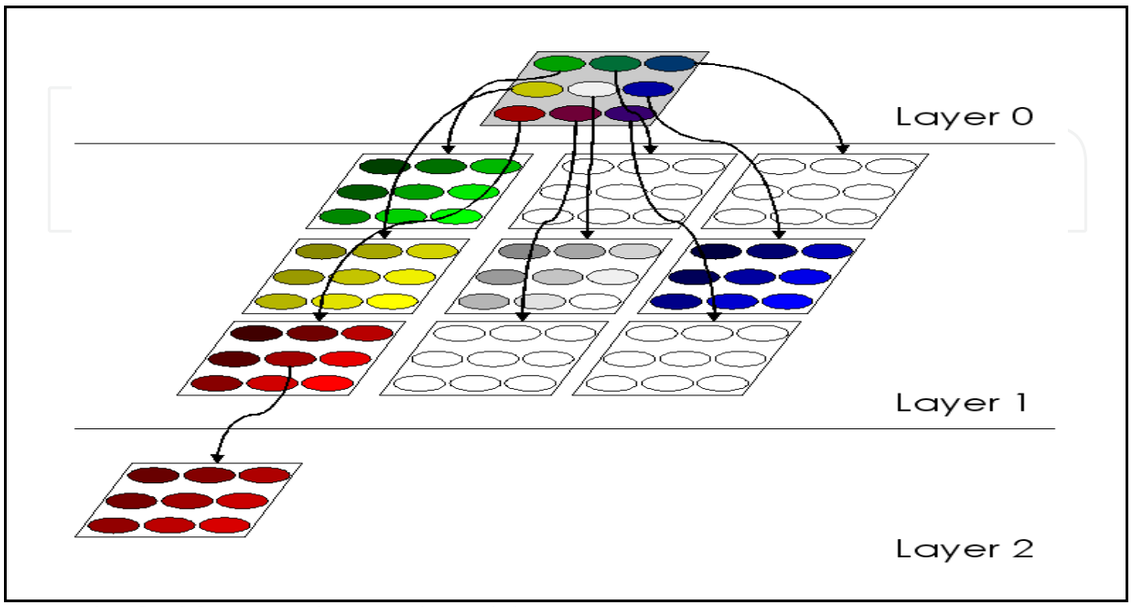

Fig. 3. The GHSOM neural network model.

5. Enrich-GHSOM

The growth of a GHSOM is a completely unsupervised process, being only driven by the unlabeled input data items themselves together with the two thresholds and some additional learning parameters. There is no way to suggest from outside any initial paths for the final learnt hierarchy. We have extended the GHSOM model with the possibility to force the growth of the hierarchy along with some predefined paths of a given hierarchy. Our new extended model, *Enrich*-GHSOM, is doing a classification of the data items into an existing tree hierarchy structure. This initial tree plays the role of an initial state for the tree-like neural network model. The classical GHSOM model grows during the training by only starting from a single node. The top-down growth in our extended model starts from a given initial tree structure and inserts new nodes attached as successors to any of its intermediate and leaf nodes.

In Enrich-GHSOM, the nodes of the predefined hierarchy are labelled with some data item labels from the input data space used for training. The training data items propagate top-down throughout the given tree hierarchy structure. When the propagation process hits a parent SOM of a tree node, then the weight vector of the corresponding parent neuron in that parent SOM is initialized with the data item vector of that successor node label. The weight vectors of the SOM neurons with no successor are initialized with random values. Then the training of that SOM proceeds by classifying the training data items against the initialized neurons. Training data items that are similar (distributionally similar as vectors) to the predefined initialized neurons are propagated downwards to the associated successor SOM nodes to continue the training (recursively) on that predefined successor SOM. Data

items that are not similar to the initialized neurons are mapped to other, non-initialized, neurons in the same SOM, and they are not propagated downwards into the predefined hierarchy. They remain as mapped into that SOM, and are considered as classified into the parent neuron of that SOM, i.e. as successor of that parent.
For instance, consider the parent neuron of a current SOM node is labelled *mammal*, and there are two predefined successor nodes labelled *feline* and *bear*, which correspond to two predefined initialized neurons in the current SOM. Then the training data item vector *dog* is not similar to any of the two neuron initializer weight vectors associated to *feline* and *bear* (see Fig. 4, where the neuron initializers are marked with bold).

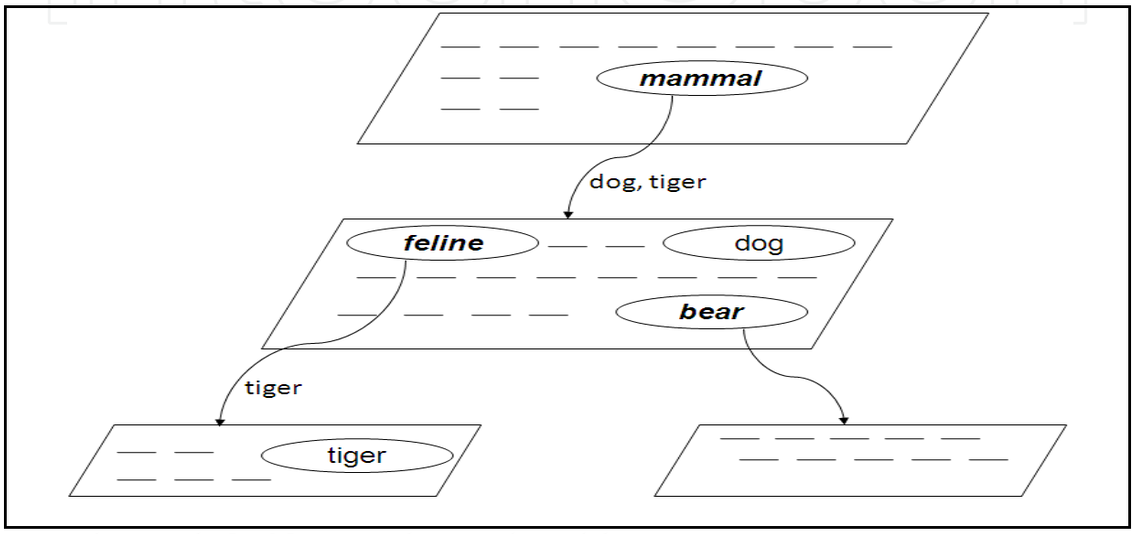

Fig. 4. The Enrich-GHSOM neural network model.

So *dog* will remain as classified into that SOM – mapped on another, non-initialized neuron – i.e. as successor (hyponym) of *mammal* and twin of the existent nodes *feline* and *bear*. Also, a data item labelled *tiger* – similar with the weight vector of the predefined "*feline*" neuron – will be propagated into the associated predefined successor SOM map together with other terms that correspond to *felines*, which will all become direct or indirect hyponyms of the concept *feline*. The process continues top-down for all the SOM nodes in the predefined initial tree hierarchy, ending at the leaves. The data item vector representations of the labels of the given initial tree play the role of *predefined initializer weight vectors* of our neural model.

6. Text-Based Ontology Enrichment Using Hierarchical Self-organizing Maps

The most important prerequisite for the success of the Semantic Web research is the construction of complete and reliable domain ontologies. Building ontologies is still a time consuming and complex task, requiring a high degree of human supervision and being still a bottleneck in the development of the semantic web technology.
The process of domain ontology enrichment has two inputs, an existing ontology – which plays the role of background knowledge – and a domain text corpus. The aim of our work is to automatically adapt the given ontology according to a domain specific corpus. We enrich

the hierarchical backbone of the existing ontology, i.e. its taxonomy, with new domain-specific concepts extracted from the corpus (Pekar and Staab, 2002).
Our framework for taxonomy enrichment is based on an extended model of hierarchical self-organizing maps, which represent an unsupervised neural network architecture. The candidates for labels of newly inserted concepts are terms collected by mining a text corpus. The term extraction process is based on recognizing linguistic patterns (noun phrases) in the domain corpus documents. Each term encodes contextual content information, in a distributional vector space. The context features of a term are the frequencies of its occurrence in different documents of the corpus. The classification of the extracted terms into the taxonomy of the given ontology proceeds by associating every term to one target node of the taxonomy, based on a similarity in the distributional vector space. That term becomes a new concept added to the taxonomy, and it is attached as hyponym (successor) under the target node.
Unsupervised hierarchical neural models in general start the growing of a dynamic tree-like topology from a single initial node. Our neural network model, called *Enrich-GHSOM*, is an extension of one of these existent systems, GHSOM (Dittenbach, et al., 2002), and it allows the growing to start from an initial tree. This is suitable to the knowledge structure to be enriched – a taxonomy, i.e. an *is-a* hierarchy of concepts. The taxonomy that is subject to enrichment is given as the initial state of the hierarchical self-organizing map. So, an essentially symbolic knowledge structure – taxonomic tree – is converted into a neural representation as an initial state of the hierarchical self-organizing map. The actual taxonomy enrichment takes place via an unsupervised training of the neural network by exposing the initialized hierarchical self-organizing map to the vector representation of the terms extracted from the domain corpus. A reverse, neural-symbolic translation is done after this enrichment process. This is actually the knowledge extraction step whose output is the final enriched taxonomy. Our taxonomy enrichment framework is a hybrid one, as it has to deal with neural-symbolic integration. The neural-symbolic translations in both directions have been naturally achieved, since our framework merely operates upon the taxonomic structure of the ontology, which is in agreement with the hierarchical structure of the self-organizing neural network.

6.1 Related Work on Taxonomy Enrichment

There are two main categories of approaches for taxonomy enrichment (Buitelaar, et al., 2005b): methods based on *distributional similarity* and *classification of terms into an existing taxonomy* on one hand, and approaches using *lexico-syntactic patterns*, also known as Hearst patterns (Hearst, 1992), on the other hand. Our enrichment approach belongs to the former category.
In the term classification approach, the terms extracted from a domain specific corpus of text are classified into an existent taxonomy (Pekar and Staab, 2002; Cimiano and Völker, 2005; Alfonseca and Manandhar, 2002; Witschel, 2005; Widdows, 2003). In a top-down variant of this classification (Pekar and Staab, 2002; Alfonseca and Manandhar, 2002; Witschel, 2005), there is a top-down search on the existent taxonomy in order to find a node under which a new term is to be inserted as a successor (hyponym). The classification of the terms is made according to a similarity measure in a distributional vector space. Each term is represented as a vector with information about different contexts of its occurrences in the corpus.

The top-down classification behaviour in our framework is modelled by a growing hierarchical self-organizing map (GHSOM) architecture (Dittenbach,et al., 2002) extended with the possibility to set an initial state for the tree-like neural network. In our new extended neural model, called Enrich-GHSOM, the given taxonomy is set as the initial state of the neural network. The model allows to classify the extracted terms into the existing taxonomy by attaching them as hyponyms for the intermediate and leaf nodes of the taxonomy. Details of this process are given in section 7.2.
A similar, although non top-down approach is (Widdows, 2003). There is a search for a node to attach a new concept as a hyponym of, by finding a place in the existent taxonomy where the corpus derived semantic neighbours of the candidate concept are most concentrated. He supposes that at least some of the semantic neighbours are already in the taxonomy, and he defines a function to compute the class label for the set of neighbours – a hypernym for all the neighbours. This class label becomes the concept under which to attach the new term as hyponym. The similarity measure to find neighbours is based on a latent semantic analysis vector space (Landauer and Dumais, 1997).

7 A Neural Model for Unsupervised Taxonomy Enrichment

The architecture of our framework for taxonomy enrichment is implemented as a pipeline with several linguistic and machine learning processing stages. The whole processing can be divided in two main steps: the *term extraction* step and the *taxonomy enrichment* step.

7.1 Extraction of Terms

The candidates for the labels of new concepts inserted during the taxonomy enrichment are terms representing *noun phrases*, identified by mining the domain text corpus. In order to identify the terms by a linguistic analysis of the corpus documents, our framework relies on several processing resources offered by the ANNIE module for analyzing English texts in the GATE framework (Cunningham, et al., 2002): morphological analyzer (stemmer), tokenizer, sentence splitter, the Hepple part-of-speech tagger, and a JAPE (Cunningham, et al., 2002) transducer. The transducer has the role to identify noun phrase constructs, based on regular expressions over different parts of speech of the component words.

7.2 Taxonomy Enrichment

The terms extracted from the domain text corpus are mapped to classes (concepts) of the existing taxonomy. The taxonomy enrichment algorithm proceeds by "populating" the given taxonomy with the terms collected from the corpus. The *Enrich*-GHSOM neural network drives a top-down hierarchical classification of the terms along with the given taxonomy branches and inserts new nodes (concepts) corresponding to these classified terms. Every new concept is attached as successor of an intermediate or a leaf node of the given taxonomy and becomes a hyponym of that target node.
In order to use our Enrich-GHSOM neural network to induce such a taxonomy enrichment behaviour, a symbolic-neural translation is first done by parsing a textual representation of the initial taxonomy (*is_a*(*concept*, *superconcept*) assertions or OWL format). The result of this parsing is the initial internal tree-like state of the neural network. In order for the initialized network to be able to classify terms into this initial taxonomic structure, apart from the

vector representation of the classified terms, a representation as a numerical vector is also needed for each node in the initial taxonomy. This vector plays the role of initial weight vector for the neural network (see section 5). It is the vector representation for the noun phrase concept label associated to the node, computed as will be described in section 7.3. The acquisition of this vector takes place in the same way as the acquisition of the vector representation of the classified terms (section 7.3).
We assume that the concept labels of the initial taxonomy are terms – noun phrases – extractable from the domain text corpus from which the classified terms themselves have also been extracted. Their vectors are then computed in the same way as the vectors of all the corpus extracted terms which are classified during the enrichment. Using the same corpus from a specialized domain to acquire the feature vectors of the concepts in the initial taxonomy and the terms to be classified is a reasonable choice, since it will reduce the problems with ambiguous (multiple) senses of one and the same term.

7.3 Vector Representation for Terms

Since Enrich-GHSOM is a connectionist system, the terms classified by Enrich-GHSOM and the concepts of the given taxonomy have to be represented as vectors. In our framework, the attributes (features) of the vector representation of a term or concept encode contextual content information, in a *distributional vector space*. Specifically, the context features are the frequencies of the occurrence of the term – classified term or concept label term – in different documents of the corpus. The number of component attributes of such a term vector coincides with the number of documents of the text corpus out of which all the terms have been extracted. Every attribute in the vector of a term is essentially the number of occurrences of the term in one document. This representation is inspired from the latent semantic analysis (Landauer and Dumais, 1997). A similar semantics-based dimensionality reduction effect as the one obtained in the latent semantic analysis by singular value decomposition is achieved in our framework by the *document category histograms* (DCH), defined in what follows.
The vector representation in the current framework satisfies Harris' distributional hypothesis (Cimiano and Völker, 2005; Buitelaar, et al., 2005b): the meaning of each classified term (or concept label) is related to the meanings of the contexts in which the term (or the concept label) occurs. In such a setting, we use the *distributional similarity* which asserts that the meaning of semantically similar terms and concept labels is expressed by similar vectors in the distributional vector space. The Euclidean distance is used in the current framework to compute the dissimilarity among vectors.
The framework allows multiple ways to encode the frequencies of occurrence, starting from simple *flat counts of occurrences*. Another variant is the *DF-ITF weighting scheme*, which means "document frequency times inverse term frequency". We propose this weighting scheme, which is a transposed of TF-IDF (Buitelaar, et al., 2005a) relative to a term/document occurrence matrix. TF-IDF is used in document classification (text categorization) and information retrieval. Now we rather classify terms, by using DF-ITF. By using this weighting scheme, we consider that long documents, which talk about too many terms, should have a lower weight when classifying terms, since they have a reduced discrimination power among the meanings of different terms. This effect is achieved by our DF-ITF weighting scheme and is confirmed by the experimental results reported in section 7.4.

A third way to encode the vector representation is one in which we propose the vector to be a *document category histogram* (*DCH*). Specifically, first a SOM (Kohonen, et al., 2000) is trained having the corpus documents as input data space to arrive at approximately 200 semantic document categories. Documents similar in meaning are clustered together by the unsupervised SOM neural network. In this SOM training, the documents are represented as vectors of frequencies for the terms they talk about. Equally like the term vectors, the document vectors are collected from the same term/document matrix, but after transposing this matrix. As we want a number of approximately 200 semantic document categories, we impose the training of a rectangular SOM map of dimension 16x12. Then, by summing up the frequencies of a term in different documents of the same category, and merely keeping the summed frequencies in different document categories as vector components, we arrive at a reduced dimensionality for the vector representation. In our experiments reported in section 7.4 with the "Lonely Planet" tourism data set, the reduction induced by such a vector representation as a histogram on semantic document categories is from 1801 (which represents the number of documents in the "Lonely Planet" corpus) to a value between 175 and 180 (in the different experimental runs described in section 7.4).

7.3.1 Data Sparseness

The *dimensionality reduction* achieved by using the document category histogram (DCH) representation is important since it removes the semantic noise caused by minor differences in semantic content for different corpus documents. Such documents now belong together to the same semantic category. This intuition is already confirmed by our experiments reported in previous work (Chifu and Leţia, 2006). Moreover, the term/document occurrence matrix is sparse (with many zeros), and reducing the dimensionality by using histograms leads to less sparse vectors. A more natural behavior of the neural network model is expected by using reduced and less sparse vectors.

A source of data sparseness is represented by *terms with very few occurrences* in the text corpus. Among such terms are the most generic terms that label the roots of the main trees in a given initial taxonomy and usually the concepts which are very high in a taxonomy. When in the *Enrich-GHSOM* neural network such an overly generic term with a very sparse vector labels the concept of one of the roots, and also when using the flat count vector representation instead of the histogram representation, then the main tree rooted by that concept is unable to attract and classify a relevant quantity of training terms. Thus the top-down search during the classification is misled. It is the case of the root concepts *spatial_concept*, *intangible*, and *thing* in the ontology of the "Lonely Planet" tourism dataset used in the present experiments. Some of the branches of these main trees are populated by no training term, which leads to the *starvation* of the neural network. Starvation means that the neural network enters an infinite loop when trying to tune the quantization error on a neuron below the thresholds (see section 4). Many of our experiments which used a flat count vector representation failed by starvation. As opposed, all the experiments using the reduced, histogram vector representation (DCH) converged to a result.

A way of reducing the number of zeros in the vector representation of the generic terms that label the generic concepts in the initial taxonomy is the *centroid vector* (Pekar and Staab, 2002; Cimiano and Völker, 2005). We have used the idea of centroid of a concept in the following way: the average vector of the vector representations of all the concepts in the sub-tree rooted by the given concept, including the root itself. Using the centroid representation

method has led us to a significant improvement of the experimental results, partially reported in (Chifu and Leția, 2006), where we rather proposed a similar approach: one of the more specific concepts in a main tree becomes a substitute for the too generic concept in the root of the tree. So, the label of every main tree root was one representative and more specific concept in the tree, for instance *course* was a substitute for *activity*, and *staff* was a substitute for *person* (in the "4 universities" domain). The improvement obtained in related work by using the centroid vector representation for concepts is reported in (Pekar and Staab, 2002; Cimiano and Völker, 2005).

7.4 Experimental Results

The experiments carried out in what follows are in the tourism domain, consisting of a corpus and a given taxonomy (the "Lonely Planet dataset") (Grobelnik, et al., 2006). The associated corpus consists of 1801 text descriptions of tourist destinations from different countries around the world.

7.4.1 Experimental Setup

In order for the corpus extracted terms to actually become domain specific concepts, they have to be noun phrases with enough frequency of occurrence in the domain specific corpus. In the term extraction process, we have set a threshold for the extracted noun phrases to occur in at least 0.5% of the number of documents in the corpus. Having set this frequency threshold, we have extracted and acquired the corresponding numerical vector representations for 1241 noun phrases. These extracted terms are classified against the taxonomy of a tourism ontology consisting of 72 concepts, which is proposed in the PASCAL ontology learning and population challenge (Grobelnik, et al., 2006).

The evaluation of the enrichment means evaluating the quality of the mapping from corpus extracted terms into target concepts of the given initial taxonomy. An extracted term becomes a new concept added to the taxonomy, and it is attached as hyponym (successor) under its associated target node. In order to evaluate the taxonomy enrichment, we followed a *cross-validation strategy* (Pekar and Staab, 2002; Witschel, 2005; Widdows, 2003). In every experimental run, exactly one node in the given initial taxonomy of 72 concepts was removed from the taxonomy, together with the whole subtree rooted by that node. The classification process was run against the result taxonomy, and the position of the held out concept, as classified like any corpus extracted term is assessed. The correct (direct hit) classification of the concept corresponds to its initial position in the taxonomy before its removal. In other words, the concept should be mapped to a target concept which was its direct hypernym (parent node) before its experimental removal. The process should be repeated 71 times, for every concept in the taxonomy except its very root, named root. Actually we repeated this experimental run 43 times, since we only had corpus statistical data to build the distributional vector representation for 43 of the taxonomy concepts. (We need a statistical distributional vector for every term to be classified.)

7.4.2 Evaluation Measures

The most appropriate measure for evaluating the taxonomy enrichment task is the *learning accuracy*, defined and evaluated in (Grobelnik, et al., 2006; Cimiano and Völker, 2005; Pekar & Staab, 2002, Alfonseca and Manandhar, 2002; Witschel, 2005). By choosing this measure,

we consider correct classifications of the new concepts with different levels of detail. For instance, the new concept *cat* can be mapped to the target concept *feline*, *carnivore*, *mammal*, or *animal* with different levels of detail, as a consequence of different taxonomic distances between the target concept as chosen by the system and the direct hypernym of the classified concept before its removal. Before removal, *cat* was direct hyponym of the *feline* concept. Classifying *cat* as *feline*, by associating it to the *feline* target concept is a direct hit, since *cat* is correctly a *direct hyponym* of *feline*, i.e. 100% learning accuracy. Though, classifying *cat* as *carnivore*, *mammal*, or *animal* are near hits, since *cat* is correct only as an indirect hyponym of *carnivore*, *mammal*, or *animal*, corresponding say to 50%, 30%, 20% learning accuracy respectively.

For a given classified term *i*, if *pi* is the target concept assigned (predicted) by the system, and *ci* the ccorrect target concept, the learning accuracy is the average over all the classified terms i of the function $LA(pi, ci)$, where the function LA is defined as

$$LA(p, c) = \frac{\delta(top, a) + 1}{\delta(top, a) + \delta(a, c) + \delta(a, p) + 1} \tag{3}$$

top is the root of the taxonomy, and *a* is the least common subsumer of the concepts *p* and *c* (i.e. the most specific common hypernym of *p* and *c*). $\delta(x,y)$ is the taxonomic distance between the concepts *x* and *y*, i.e. the number of taxonomy edges to be traversed when going from the taxonomy node labelled *x* towards node *y*. This is the most used formula to compute the learning accuracy. In the context of the Pascal ontology learning and population challenge, it is actually called *symmetric learning accuracy*, and the term *learning accuracy* is used for a historically initial version of the learning accuracy measure, as introduced by (Hahn and Schnattinger, 1998):

$$LA'(p, c) = \begin{cases} \dfrac{\delta(top, a) + 1}{\delta(top, c) + 1} & \text{if } p \text{ is ancestor of } c \\ & \text{(then also } a = p\text{)} \\ \dfrac{\delta(top, a) + 1}{\delta(top, a) + 2 * \delta(a, p) + 1} & \text{otherwise} \end{cases} \tag{4}$$

According to formulae (3) and (4) to compute both variants of the learning accuracy, the same number of edges in the taxonomic distance between the predicted and the correct target concept means a better accuracy when the edges are lower in the taxonomy. This is due to the intuition that the same number of edges between two concrete (lower in the taxonomy) concepts means an increased similarity (a reduced semantic distance), as compared to the same number of edges between two abstract concepts (higher in the taxonomy).

Another quantitative evaluation measure similar in spirit to the learning accuracy is the *edge measure*. It actually counts the average deviation (in terms of taxonomic distance) between the system predicted target concept and the correct one. Consequently, as opposed to the

first two learning accuracy measures (formulae (3) and (4)), the edge measure means a better classification for a lower edge measure value.

7.4.3 Evaluation Results

A first set of experimental runs is based on a document category histogram (DCH) vector representation for the extracted terms and concept label terms. Also, the concept label terms of the given initial taxonomy are represented using the *centroid* method for the whole sub-tree of a given concept node, as described in section 7.3. The improvements gained by using DCH and centroid are already confirmed qualitatively by our experiments reported in (Chifu and Leția, 2006). Furthermore, not only the training of the Enrich-GHSOM neural network is less efficient on flat count vectors with 1801 attributes (corresponding to the 1801 corpus documents) compared to the 180 attributes (for the 180 semantic document categories) in DCH's, but also using flat count (unreduced) vectors often leads to the starvation of the neural network.

In a second set of experiments, we first applied the *DF-ITF* weighting scheme on the flat count term vectors of 1801 attributes. The result vectors were then converted into DCH histograms, thus reducing the term vector dimensionality to 179.

(Cimiano and Völker, 2005) and (Pekar and Staab, 2002) used the centroid vector to reperesent the concept nodes. (Pekar and Staab,2002) found out that their best results were achieved when taking into account only the first three levels of successors in the sub-tree of the concept in order to compute the centroid. The experiments in (Cimiano and Völker, 2005) considered only the direct successors of the concept to compute the centroid. Driven by these results, we ran a third set of experiments, in which we considered only the first level of successors to represent the centroid of any concept in the given taxonomy, like in (Cimiano and Völker, 2005). We didn't also try the three-level version of (Pekar and Staab, 2002), since the results would be similar with our results for whole sub-trees. This is because the average depth of the taxonomy to be enriched in our experiments is 4, and the majority of the nodes don't have sub-trees of depth greater than 3. In this third set of experiments we kept the DF-ITF and DCH settings like in the second experiment.

All these experiments involved the cross-validation experimental strategy described in section 7.4.1, in which every classified concept was previously removed together with its whole subtree. In a fourth set of experiments, we only removed the tested concept alone, and kept untouched its whole subtree, which rather became a subtree of its parent node.

We evaluated the three learning accuracy measures on placing the 43 concepts in their actual position in the given initial ontology from the Pascal challenge (Grobelnik, et al., 2006). The results are illustrated in Table 2.

All the three learning accuracy measures are considerably improved by using the DF-ITF weighting measure, and keeping the DCH histogram vector representation. These results prove that the quality of the enrichment is improved by using our contributed semantics based vector representations (DCH and DF-ITF) for the classified terms and the concept label terms in the initial taxonomy. Another finding is that limiting the depth of the sub-concepts for the computation of the centroid vector representation for taxonomy concepts leads to a slight degradation of the learning accuracy. The experiments in (Witschel, 2005) also confirm that using whole sub-trees to represent the centroid of the concepts improves the performance of the taxonomy enrichment. Removing only the tested taxonomic node alone keeps the enrichment quality roughly unchanged as compared to removing the whole

subtree. This confirms the expectation that a concept is semantically linked with all the nodes in its subtree. The whole subtree represents a class of terms which lexicalize the concept in the root of the subtree.

Vector Representation	DCH	DF-ITF+DCH	DF-ITF+DCH	DF-ITF+DCH
Concept Label Centroid	whole subtree centroid	whole subtree centroid	first-level centroid	whole subtree centroid
Removed	whole subtree	whole subtree	whole subtree	node only
Learning Accuracy	33.351%	39.654%	37.679%	38.76%
Symmetric Learning Accuracy	33.998%	40.272%	38.016%	40.402%
Edge Measure	3.023	2.651	2.861	2.698

Table 2. Learning accuracy of the taxonomy enrichment when using DCH, DF-ITF, and different variants of centroid.

7.4.3.1 Named Entity Classification

In a last set of experiments, instead of classifying terms represented by common noun phrases extracted from the "Lonely Planet" corpus, we rather classified noun phrases for proper names – i.e. named entities – extracted from the same corpus. The majority of the named entities occur few times in the corpus, and many of them only occur once, in a singe document. This is why, in the experiments reported in what follows, we have reduced the frequency threshold to zero. It was 0.5% in the preceding experiments (see section 7.4.1).

Having no more frequency threshold for the corpus extracted noun phrases, we found and extracted a total of 43006 noun phrases, compared to 1241 in the preceding three taxonomy enrichment experiments. Some of them are common nouns and the other are named entities. We will refer in what follows to this experiment as *the maximal experiment*. To reduce the dimensionality of the data, and consequently the inherent noise, one of our experiments was trying to keep only what is absolutely necessary for the classification. We kept a minimum of common noun phrases corresponding to the concept labels in the taxonomy, and a minimum of proper noun phrases representing the set of named entities asked to be classified in the PASCAL ontology learning and population challenge (Grobelnik, et al., 2006). The total number of common and proper noun phrases extracted is reduced to 631. We will call this experimental run *the minimal experiment*.

We evaluated these last experiments automatically by using the PASCAL challenge site online evaluation system. This evaluation system is based on a *gold standard*, i.e. an ontology populated with the set of named entities that are asked to be classified in the PASCAL challenge. In other words, the PASCAL competition target set of named entities are considered as correctly mapped to the different concepts in the gold standard ontology. In *the maximal experiment*, a number of 625 named entities extracted from the "Lonely Planet" corpus are classified against an ontology consisting of 72 concepts, which is proposed in the PASCAL challenge (Grobelnik, et al., 2006). Actually there are much more named entities extracted by our framework, but only 625 of them are also included in the set of named

entities asked to be classified in the PASCAL ontology learning and population challenge. In *the minimal experiment*, 417 named entities are classified into a taxonomy consisting of 96 concepts. Table 3 illustrates these last two experiments, as evaluated automatically with the PASCAL challenge online evaluation system.

There are two explanations for the lower classification quality values in the maximal experiment as compared to the minimal one. First, the minimal experiment uses the DCH histogram vector representation as compared to the flat counts of the maximal experiment, and second is the noise caused by the much bigger quantity of noun phrases classified in the maximal experiment – 43006 versus 631. Also, an explanation for an overall degraded quality of the *named entity classification* as compared to the *taxonomy enrichment* in the preceding experiments is that the classified named entities have very low frequency of occurrence as compared to the classified terms (common nouns) from the taxonomy enrichment, and consequently they have a very sparse vector representation. This misleads their classification.

Experiment	*maximal experiment*	*minimal experiment*
Vector Representation	flat counts	DCH
Concept Label Centroid	whole subtree centroid	whole subtree centroid
Learning Accuracy	22.4%	31.2%
Symmetric Learning Accuracy	21.2%	28.5%
Edge Measure	3.754	4.767

Table 3. Learning accuracy of the named entity classification.

8. Taxonomy Learning Using Self-Organizing Trees

SOTA (self-organizing tree algorithm) (Herrero, 2001) is another dynamical tree-like self-organizing map. It is an unsupervised neural network with a binary tree topology, which is available as SOTArray (Herrero, 2001). The clustering algorithm in SOTA is a top-down process: the tree grows starting from its root, and then develops into more detailed classifications on the lower hierarchical levels. This growing stops when a predefined level of classification detail is reached. The level of detail is set according to the distribution of probability obtained by randomization of the data set to be classified. The tree-like output space can freely grow until adapting as much as possible to the variability of the input data space. Alternatively, new nodes can grow until reaching a complete classification of the data items, i.e. until having a single data item in every leaf of the tree. This is the setting we used when applying SOTA for taxonomy learning.

We have used the SOTA model to learn a taxonomy starting from a text corpus. A learned SOTA hierarchy plays the role of a learned taxonomy. The hierarchical clustering of the terms is done in a top-down manner, the upper levels being generated before the lower levels, which are more detailed and will contain more specific terms. SOTArray classifies the initial data set only in the leaves of the binary tree that it develops, the inner nodes being empty. The taxonomy structure obtained with the SOTA algorithm is a binary tree of terms. In every leaf we have one term from the corpus. After the training of SOTArray, we labelled the inner nodes starting from the leaves and ascending towards the root of the tree

hierarchy, by searching the WordNet database (Fellbaum, 1998) for the most specific common hypernym of every two sibling nodes.

9. Conclusions and Further Work

This chapter presented several uses of the self-organizing maps in the context of web mining and the semantic web. The self-organizing maps constitute a powerful model for Web mining by defining a visual overview of a set of Web documents. A document SOM map is a semantically ordered spread of the documents in the set. Our SOM-based visualization system for document collections is a powerful information retrieval tool for browsing a set of Web documents. The system is especially useful when the user has rather limited knowledge about the domain or the contents of the text collection.

The unsupervised top-down neural network based approach and framework for taxonomy enrichment is also essentially based on self-organizing maps. The framework can be applied to different domains and languages. The experimental results obtained in the "Lonely Planet" tourism domain prove that our contributed semantics based vector representations, i.e. the *document category histograms* and the *DF-ITF weighting scheme*, are suitable for the task of taxonomy enrichment.

The comparison of taxonomy enrichment systems (and of named entity classifiers) is problematic. Different systems use different domains and, even for the same domain, they use different corpora of different sizes and different ontologies. (Grobelnik, et al., 2006) present such a comparison of existent systems, and the conclusion is that the classification quality degrades with the increase in the size of the ontology.

Another interesting point is that sometimes given taxonomic structures are not reflecting correctly some fine-grained meanings. For instance, in the initial taxonomy used in our experiments, f*orest* is hyponym of *area*. However the context in which the term *forest* occurs in the corpus are rather specific to plants (*plant* concept), which is far in the taxonomy from *area*. Our system "incorrectly" classified *forest* as *plant*.

The data sparseness remains a problem for the task of taxonomy enrichment. Terms (or named entities) represented by sparse vectors have an increased chance to be wrongly classified, because of their reduced power of being attracted towards the correct branches and nodes of the taxonomy. Thus the top-down search during the classification is misled, and this phenomenon is mostly encountered in the case of named entity classification, where named entities have very sparse vector representations. Consequently, as further work, we will try to change the statistical distributional vector representation of the terms to further reduce the dimensionality of the vectors. We will try using pseudo-syntactic dependencies as representation of the terms, in the spirit of (Cimiano and Völker, 2005).

10. References

Alfonseca, E., Manandhar, S. (2002). Extending a lexical ontology by a combination of distributional semantics signatures, In: A. Gómez-Pérez, V.R. Benjamins (eds.) *13th International Conference on Knowledge Engineering and Knowledge Management*, LNAI. Springer, pp. 1-7

Buitelaar, P., Cimiano, P., Grobelnik, M., Sintek, M. (2005a). Ontology learning from text, Tutorial at ECML/PKDD workshop on Knowledge Discovery and Ontologies

Buitelaar, P., Cimiano, P., Magnini B. (2005b). Ontology learning from text: an overview, In: P. Buitelaar, P. Cimiano, B. Magnini (eds.) *Ontology Learning from Text: Methods, Evaluation and Applications, Frontiers in Artificial Intelligence and Applications Series,* IOS Press, pp. 1-10

Chifu, E.Şt., Leţia, I.A. (2006). Unsupervised ontology enrichment with hierarchical self-organizing maps. In: Leţia, I.A. (ed.) *IEEE 2nd International Conference on Intelligent Computer Communication and Processing*, pp. 3-9

Cimiano, P., Völker, J. (2005). Towards large-scale, open-domain and ontology-based named entity classification, In: *RANLP'05, International Conference on Recent Advances in Natural Language Processing*, pp. 166-172

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V. (2002). GATE: a framework and graphical development environment for robust NLP tools and applications, In: *40th Anniversary Meeting of the ACL*

Dittenbach, M., Merkl, D., Rauber, A. (2002). Organizing and exploring high-dimensional data with the Growing Hierarchical Self-Organizing Map, In Wang, L., et al. (eds.) *1st International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 2, pp. 626-630

Fellbaum, Chr. (Ed.) (1998). *WordNet: An Electronic Lexical Database*, MIT Press. Cambridge, Mass.

Giles, J.T.,L. Wo, L., and Berry, M.W. (2003). GTP (General Text Parser) software for text mining, in H. Bozdogan, ed., *Statistical Data Mining and Knowledge Discovery*, CRC Press, Boca Raton, pp. 455-471.

Grobelnik, M., Cimiano, P., Gaussier, E., Buitelaar, P., Novak, B., Brank, J., Sintek, M. (2006). Task description for PASCAL challenge, Evaluating ontology learning and population from text

Hearst, M.A. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora, *In: 14th International Conference on Computational Linguistics*, pp. 539-545

Hahn, U., Schnattinger, K. (1998). Towards text knowledge engineering, In: *15th National Conference on Artificial Intelligence and the 10th Conference on Innovative Applications of Artificial Intelligence* (*AAAI/IAAI*), pp. 524-531

Hautaniemi, S., Yli-Harja, O., Astola, J., Kauraniemi, P., Kallioniemi, A., Wolf, M., Ruiz, J., Mousses, S., and Kallioniemi, O.-P. (2003). Analysis and visualization of gene expression microarray data in human cancer using self-organizing maps, *Machine Learning*, vol. 52, pp. 45-66.

Herrero, J., Valencia, A., and Dopazo, J. (2001). A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, 17, pp. 126–136.

Honkela, T. (1997). Self-organizing maps in natural language processing, *PhD thesis*, Neural Networks Research Center, Helsinki University of Technology, Finland.

Honkela, T., Kaski, S., Lagus, K., and Kohonen, T. (1996). Exploration of full-text databases with self-organizing maps, in *Proceedings of the International Conference on Neural Networks*, vol. I, pp. 56-61.

Kohonen, T., Hynninen, J., Kangas, J., and Laaksonen, J. (1996). SOM_PAK: The self-organizing map program package, *Technical Report A31*, Helsinki University of Technology, Laboratory of Computer and Information Science, 1996.

Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., Saarela, A. (2000). Self-organization of a massive document collection, *IEEE Transactions on Neural Networks* 11, pp. 574-585

Lang, K. (1995). NewsWeeder: Learning to filter news. In: *12th International Conference on Machine Learning*, pp. 331-339

Lagus, K., (2000). Text retrieval using self-organized document maps, *Technical Report A61*, Helsinki University of Technology, Laboratory of Computer and Information Science.

Lagus, K. and Kaski, S. (1999). Keyword selecton method for characterizing text document maps, *in Proceedings of the 9th International Conference on Artificial Neural Networks*, vol. 1, pp. 371-376.

Landauer, T.K., Foltz, P.W. and Laham, D (1998). Introduction to Latent Semantic Analysis, Discourse Processes, vol. 25, 1998, pp. 259-284.

Landauer, T., Dumais, S., (1997). A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge, *Psychological Review* 104, 211–240

Lesk M.E. and Schmidt, E.,(1995). Lex – a lexical analyzer generator, *Computing Science Technical Report 39*, AT&T Bell Laboratories, Murray Hill, 1975; UNIX Programmer's Manual, vol. 2B, Bell Laboratories.

Pekar, V., Staab, S. (2002). Taxonomy learning – factoring the structure of a taxonomy into a semantic classification decision, In: *COLING'02, 19th International Conference on Computational Linguistics*, pp.786-792

Ultsch, A., (1993). Self organized feature maps for monitoring and knowledge acquisition of a chemical process", in S. Gielen and B. Kappen, eds., *Proceedings of the International Conference on Artificial Neural Networks,* pp. 864-867.

Widdows, D. (2003). Unsupervised methods for developing taxonomies by combining syntactic and statistical information, In: *HLT-NAACL Conference*, pp. 197-204

Wilppu, E. (1997). The visualization capability of self-organizing maps to detect deviations in distribution control, *Technical Report* 153, Turku Centre for Computer Science.

Witschel, H.F. (2005). Using decision trees and text mining techniques for extending taxonomies, In: *Learning and Extending Lexical Ontologies by using Machine Learning Methods*, Workshop at ICML-05, pp. 61-68

Self-Organizing Maps

Edited by George K Matsopoulos

ISBN 978-953-307-074-2
Hard cover, 430 pages
Publisher InTech
Published online 01, April, 2010
Published in print edition April, 2010

The Self-Organizing Map (SOM) is a neural network algorithm, which uses a competitive learning technique to train itself in an unsupervised manner. SOMs are different from other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space and they have been used to create an ordered representation of multi-dimensional data which simplifies complexity and reveals meaningful relationships. Prof. T. Kohonen in the early 1980s first established the relevant theory and explored possible applications of SOMs. Since then, a number of theoretical and practical applications of SOMs have been reported including clustering, prediction, data representation, classification, visualization, etc. This book was prompted by the desire to bring together some of the more recent theoretical and practical developments on SOMs and to provide the background for future developments in promising directions. The book comprises of 25 Chapters which can be categorized into three broad areas: methodology, visualization and practical applications.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen