Text-Based Ontology Enrichment Using Hierarchical Self-organizing Maps

Emil Şt. Chifu and Ioan Alfred Leţia

Technical University of Cluj-Napoca, Department of Computer Science, Bariţiu 28,
RO-400027 Cluj-Napoca, Romania
{Emil.Chifu, letia}@cs.utcluj.ro

Abstract. The success of the Semantic Web research is dependent upon the construction of complete and reliable domain ontologies. In this paper we describe an unsupervised framework for domain ontology enrichment based on mining domain text corpora. Specifically, we enrich the hierarchical backbone of an existing ontology, i.e. its taxonomy, with new domain-specific concepts. The framework is based on an extended model of hierarchical self-organizing maps. As being founded on an unsupervised neural network architecture, the framework can be applied to different languages and domains. Terms extracted by mining a text corpus encode contextual content information, in a distributional vector space. The enrichment behaves like a classification of the extracted terms into the existing taxonomy by attaching them as hyponyms for the nodes of the taxonomy. The experiments reported are in the "Lonely Planet" tourism domain. The taxonomy and the corpus are the ones proposed in the PASCAL ontology learning and population challenge. The experimental results prove that the quality of the enrichment is considerably improved by using semantics based vector representations for the classified (newly added) terms, like the document category histograms (DCH) and the document frequency times inverse term frequency (DF-ITF) weighting scheme.

Keywords: taxonomy enrichment, unsupervised neural network, extended growing hierarchical self-organizing maps (Enrich-GHSOM), document category histograms (DCH), document frequency times inverse term frequency (DF-ITF) weighting scheme, centroid vector.

1 Introduction

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 [14].

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 [8], and it allows the growing to start from an initial tree. 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.

In the rest of the paper, after a review of related work, section 3 presents the neural network learning solution chosen and adapted in our framework. Then section 4 details the architecture and implementation of the taxonomy enrichment framework and section 5 describes the experimental results. Conclusions and future directions are presented in section 6.

2 Related Work

There are two main categories of approaches for taxonomy enrichment [3]: 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 [10], 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 [14, 6, 1, 16, 15]. In a top-down variant of this classification [14, 1, 16], 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 behavior in our framework is modeled by a growing hierarchical self-organizing map (GHSOM) architecture [8] 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 4.2.

A similar, although non top-down approach is [15]. 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 neighbors of the candidate concept are most concentrated. He supposes that at least some of the semantic neighbors are already in the taxonomy, and he defines a function to compute the class label for the set of neighbors – a hypernym for all the neighbors. This class label becomes the concept under which to attach the new term as hyponym. The similarity measure to find neighbors is based on a latent semantic analysis vector space [13].

3 Neural Network Learning Method

Our extended model of hierarchical self-organizing maps – Enrich-GHSOM – represents the unsupervised neural network based learning solution adopted by our taxonomy enrichment framework. This choice is suitable to the knowledge structure to be enriched – a taxonomy, i.e. an *is-a* hierarchy of concepts.

3.1 Self-organizing Maps

GHSOM is an extension of the Self-Organizing Map (SOM, also known as Kohonen map) learning architecture [12, 5], which is one of the most popular unsupervised neural network models. SOM can be seen as a projection method which maps a high dimensional data space into a lower dimensional one. The resulting lower dimensional output space is a rectangular SOM map, represented as a two-dimensional grid of neurons. Each input data item is mapped into one of the neurons in the map. SOM is also a clustering method, so that similar data items – represented as vectors of numerical values – tend to be mapped into nearby neurons.

The SOM map learns by a self-organization process. There is no initial knowledge about the membership of any input data item in a particular class or about the number of classes. The training proceeds with unlabeled input data like any unsupervised learning. Clusters (classes) are discovered and described by gradually detected characteristics during the training process. These gradually adjusted characteristics play the role of weights in the weight vector associated to each neuron. The role of a completely trained map is to represent all the available observations – the whole input

data space – with optimal accuracy by using a restricted set of weight vectors associated to the map neurons.

The initial values for the weight vectors of the neurons can either be chosen depending on the problem domain or they can be taken randomly. Every iteration of the learning algorithm processes one input (training) vector as follows. Like usually for unsupervised neural networks, some form of competitive learning takes place: the winner neuron index c, which best matches the current input vector, is identified as the neuron whose weight vector is most similar to the current input vector in some metric. Then all the weight vectors or a subset of them that correspond to neurons centered around the winner neuron c – i.e. neurons in the neighborhood area of c –, including the winner itself, are updated in the direction of the input vector. This adaptation renders a globally ordered map in the process of learning. A neuron has four immediate neighbors in a rectangular map topology, which is our chosen map topology. This is merely a rectangular lattice type of the two-dimensional grid of neurons, and the SOM map is kept as a planar rectangle.

3.2 Growing Hierarchical Self-organizing Maps

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 [8]. 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. 1). 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.

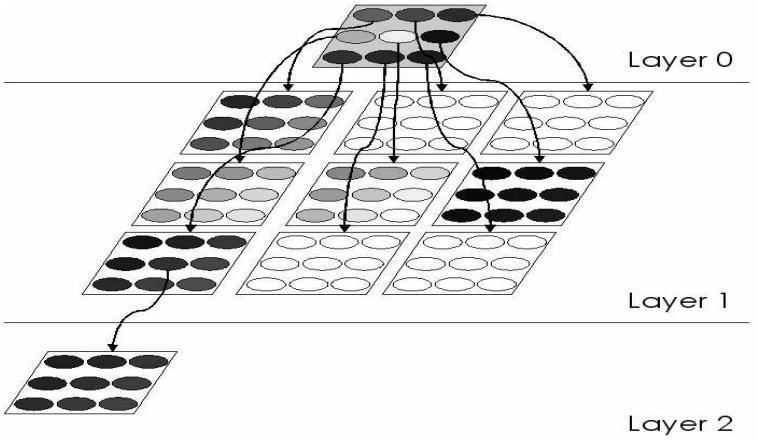

Fig. 1. The GHSOM neural network model.

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.

3.3 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 taxonomic 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 labeled 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 labeled *mammal*, and there are two predefined successor nodes labeled *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. 2, where the neuron initializers are marked with bold). 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 labeled *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.

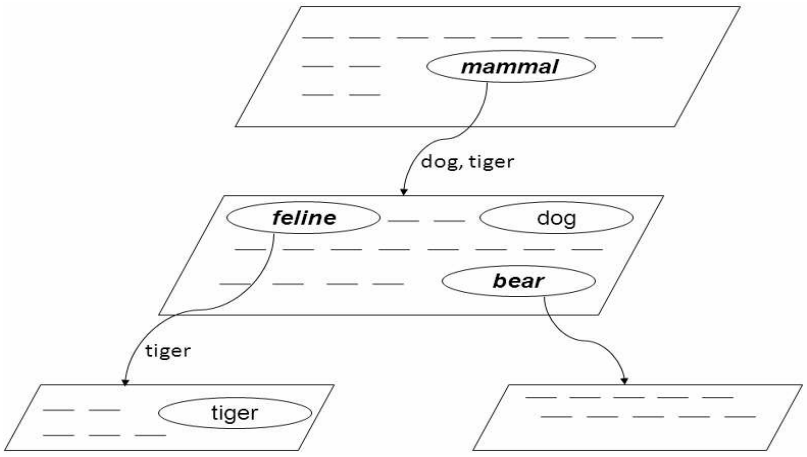

Fig. 2. The Enrich-GHSOM neural network model.

4 A Neural Model for Unsupervised Taxonomy Enrichment

The architecture of our framework 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.

4.1 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 [7]: morphological analyzer (stemmer), tokenizer, sentence splitter, the Hepple part-of-speech tagger, and a JAPE [7] transducer. The transducer has the role to identify noun phrase constructs, based on regular expressions over different parts of speech of the component words.

4.2 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 node.

In order to use our Enrich-GHSOM neural network to induce such a taxonomy enrichment behavior, 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 3.3). It is the vector representation for the noun phrase concept label associated to the node, computed as will be described in section 4.3. The acquisition of this vector takes place in the same way as the acquisition of the vector representation of the classified terms (section 4.3).

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.

4.3 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 [13]. 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 [6, 3]: 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 [2] 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 5.

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 [12] 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 5 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 180 and 179 (in two different experimental runs described in section 5).

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 [4]. 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 3.2). 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* [14, 6]. We have used the idea of centroid 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 [4], 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 by using the centroid vector representation for concepts is reported in [14, 6].

5 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") [9]. The associated corpus consists of 1801 text descriptions of tourist destinations from different countries around the world.

5.1 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 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 73 concepts, which is proposed in the PASCAL ontology learning and population challenge [9].

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* [14, 16, 15]. In every experimental run, exactly one node in the given initial taxonomy of 73 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 72 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.)

5.2 Evaluation Measures

The most appropriate measure for evaluating the taxonomy enrichment task is the *learning accuracy*, defined and evaluated in [9, 6, 14, 1, 16]. 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 hypernym-hyponym 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% classification 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% classification accuracy respectively.

For a given classified term *i*, if *pi* is the target concept assigned (predicted) by the system, and *ci* the correct target concept according to the given initial taxonomy, 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

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

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*). $\delta(a, b)$ is the taxonomic distance between the concepts *a* and *b*, i.e. the number of taxonomy edges to be traversed when going from the taxonomy node labeled *a* towards node *b*. 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 [11]:

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

According to formulae (1) and (2) 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 according to the given initial taxonomy. Consequently, as opposed to the first two learning accuracy measures (formulae (1) and (2)), the edge measure means a better classification for a lower edge measure value.

5.3 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 4.3. The improvements gained by using DCH and centroid are already confirmed qualitatively by our experiments reported in [4]. 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 dimension to 179.

[6] and [14] used the centroid vector to reperesent the concept nodes. [14] 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 [6] consider 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 [6]. We didn't also try the three-level version of [14], 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.

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 [9]. The results are illustrated in Table 1.

Table 1. Learning accuracy of the taxonomy enrichment when using DCH, DF-ITF, and different variants of centroid.

Vector Representation	DCH	DF-ITF + DCH	DF-ITF + DCH
Concept Label Centroid	whole subtree centroid	whole subtree centroid	first-level centroid
Learning Accuracy	33.565%	39.654%	37.679%
Symmetric Learning Accuracy	33.742%	40.437%	38.016%
Edge Measure	3.023	2.651	2.907

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 [16] also confirm that using whole sub-trees to represent the centroid of the concepts improve the performance of the taxonomy enrichment.

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 5.1).

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 [9]. 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[1]. 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 623 named entities extracted from the "Lonely Planet" corpus are classified against an ontology consisting of 74 concepts, which is proposed in the PASCAL challenge [9]. Actually there are much more named entities extracted by our framework, but only 623 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 2 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

[1] http://olc.ijs.si/eval.html.

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.

Table 2. Learning accuracy of the named entity classification.

Experiment	*maximal experiment*	*minimal experiment*
Vector Representation	flat counts	DCH
Concept Label Centroid	whole subtree centroid	whole subtree centroid
Learning Accuracy	22.3%	31.2%
Symmetric Learning Accuracy	21.2%	28.5%
Edge Measure	3.78	4.767

6 Conclusions and Further Work

We have presented an unsupervised top-down neural network based approach and framework for taxonomy enrichment. 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. [6] 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, *forest* 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 the reduced power of attraction 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 [6].

References

1. Alfonseca, E., Manandhar, S.: 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 (2002)
2. Buitelaar, P., Cimiano, P., Grobelnik, M., Sintek, M.: Ontology learning from text. Tutorial at ECML/PKDD workshop on Knowledge Discovery and Ontologies (2005)
3. Buitelaar, P., Cimiano, P., Magnini B.: 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 (2005)
4. Chifu, E.Şt., Leţia, I.A.: Unsupervised ontology enrichment with hierarchical self-organizing maps, In: IEEE 2nd International Conference on Intelligent Computer Communication and Processing, pp. 3-9, IEEE Press, Cluj-Napoca (2006)
5. Chifu, E.Şt., Leţia, I.A.: Web mining with self-organizing maps, 8th IEEE International Conference on Intelligent Engineering Systems, pp. 93-98 (2004)
6. Cimiano, P., Völker, J.: 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 (2005)
7. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: a framework and graphical development environment for robust NLP tools and applications. In 40th Anniversary Meeting of the ACL (2002)
8. Dittenbach, M., Merkl, D., Rauber, A.: Organizing and exploring high-dimensional data with the Growing Hierarchical Self-Organizing Map. In L. Wang, *et al*. (Eds.), 1st International Conference on Fuzzy Systems and Knowledge Discovery, vol. 2, pp. 626-630 (2002)
9. Grobelnik, M., Cimiano, P., Gaussier, E., Buitelaar, P., Novak, B., Brank, J., Sintek, M.: Task description for PASCAL challenge. Evaluating ontology learning and population from text (2006)
10. Hearst, M.A.: Automatic Acquisition of Hyponyms from Large Text Corpora. In: 14th International Conference on Computational Linguistics, pp. 539-545 (1992)
11. Hahn, U., Schnattinger, K.: 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 (1998)
12. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., Saarela, A.: Self-organization of a massive document collection. IEEE Transactions on Neural Networks 11, pp. 574-585 (2000)
13. Landauer, T., Dumais, S.: A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge. Psychological Review 104, 211–240 (1997)
14. Pekar, V., Staab, S.: Taxonomy learning – factoring the structure of a taxonomy into a semantic classification decision. In COLING'02, 19th International Conference on Computational Linguistic*s*, pp.786-792 (2002)
15. Widdows, D.: Unsupervised methods for developing taxonomies by combining syntactic and statistical information. In HLT-NAACL Conference, pp. 197-204 (2003)
16. Witschel, H.F.: 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 (2005)