A Neuro-Evolutionary Corpus-Based Method for Word Sense Disambiguation

Antonia Azzini, Célia Da Costa Pereira, Mauro Dragoni, Andrea G. B. Tettamanzi

To cite this version:

Antonia Azzini, Célia Da Costa Pereira, Mauro Dragoni, Andrea G. B. Tettamanzi. A Neuro-Evolutionary Corpus-Based Method for Word Sense Disambiguation. IEEE Intelligent Systems, Institute of Electrical and Electronics Engineers, 2012, 27 (6), pp.26-35. <10.1109/MIS.2011.108>. <hal-01328703>

HAL Id: hal-01328703

https://hal.archives-ouvertes.fr/hal-01328703

Submitted on 8 Jun 2016

A Neuro Evolutionary Corpus-based Method for Word Sense Disambiguation

Antonia Azzini*
Università degli Studi di Milano

Célia da Costa Pereira*
Université de Nice Sophia Antipolis/CNRS

Mauro Dragoni+
Fondazione Bruno Kessler (FBK)

Andrea G. B. Tettamanzi*
Università degli Studi di Milano

We propose a supervised approach to Word Sense Disambiguation based on Neural Networks combined with Evolutionary Algorithms.

An established method to automatically design the structure and learn the connection weights of Neural Networks by means of an Evolutionary Algorithm is used to evolve a neural-network disambiguator for each polysemous word, against a dataset extracted from an annotated corpus.

Two distributed encoding schemes, based on the orthography of words and characterized by different degrees of information compression, have been used to represent the context in which a word occurs. The performance of such encoding schemes has been compared.

The viability of the approach has been demonstrated through experiments carried out on a representative set of polysemous words. Comparison with the best entry of the Semeval-2007 competition has shown that the proposed approach is almost competitive with state-of-the-art WSD approaches.

1. Introduction

Word Sense Disambiguation (WSD) consists of assigning the most appropriate meaning to a polysemous word, that is a word with a number of meanings greater than one.

The appropriate meaning of a polysemous word depends on the context in which it appears. Approaches to this problem are based on the distributional hypothesis pointed out, for example in (Harris 1954), whereby words with similar meanings are often used in similar contexts and similar contexts for an ambiguous word also suggest the same or a similar meaning.

The automatic word sense disambiguation process, in general, consists of two steps:

(i) considering the possible senses of the given word;

(ii) assigning each occurrence of the word to its *appropriate* sense.

To single out the appropriate sense of a word, its context, i.e., information contained within the text where the word occurs, is considered. To represent such contexts is one

* Dipartimento di Tecnologie dell'Informazione, Via Bramante 65, I-26013 Crema (CR), Italy. E-mail: antonia.azzini@unimi.it,andrea.tettamanzi@unimi.it

of the most important steps in the automatic process of WSD. The more *effective* the representation of the context is, the more satisfactory the results of the WSD process are.

The existing WSD systems can be classified into two groups: *unsupervised systems* and *supervised systems*. Unsupervived systems learn directly from raw data by: (1) considering the hypothesis that different words have similar meanings if they are presented in similar contexts (Lesk 1986); or (2) simply assigning to all instances of an ambigous word its most frequent sense (McCarthy et al. 2004). Instead, supervised systems, which are, to date, the most accurate existing WSD systems, essentially amount to a classification task and use annotated training data (Escudero G. and Rigau 2006; Martínez, de Lacalle, and Agirre 2008; Navigli and Ponzetto 2010). The training is made on a disambiguated corpus, and aims at collecting available information about the ambiguous words. The testing phase consists of choosing the sense with the highest similarity to the target sense on the basis of the data in the training set.

Actually, the state of the art comprises supervised and knowledge-based systems (Chen et al. 2010), which both have their advantages and drawbacks. In this work, we propose a supervised approach to word sense disambiguation based on artificial neural networks (ANNs) combined with evolutionary algorithms (EAs). EAs are attractive to approach Natural Language Processing (NLP) problems due to their ability to effectively search huge model spaces. This is exactly what needs to be done for solving WSD problems. A clear weakness of EAs is that it is not clear how to make them exploit linguistic knowledge to improve their performance.

We take advantage of the corpus developed by the IXA group which is a large tagged dataset describing the contexts in which every sense of a polysemous word occurs, and use them to evolve an optimized ANN that correctly disambiguates the sense of a word given its context. We obtain a class of ANNs, each of them specialized in recognizing the correct sense of its corresponding word, one for each polysemous word in the dictionary.

The paper is organized as follows: Section 2 presents the WSD problem in general and the particular formulation used in the paper; Section 4 describes the proposed WSD approach; Section 5 presents the experiments to test and validate our approach; and, finally, Section 6 concludes.

2. The Word Sense Disambiguation Problem

WSD is a classification problem. Given an *instance* of a word and the *context* in which it occurs, the aim is to determine the sense of that occurrence of the word (Veronis and Ide 1990).

Let W be the set of all words for a given natural language (e.g., English), and S_w the set of all possible senses of word w. Let C be a set of *contexts* in which a word instance may occur. We formulate the WSD problem as follows: given a polysemous word $w \in W$, find the function

$$f_w^* : C \to S_w, \tag{1}$$

such that, for all context $c \in C$, $f_w^*(c) \subset S_w$ is the most plausible (roughly speaking: correct) sense of w in context c.

It is surely very hard, and probably impossible, to solve the WSD problem *exactly*. Instead, all existing approaches aim at finding the best approximation of f_w^*. Measuring how well a candidate function f_w approximates f^*_w, i.e., *evaluation*, is not a trivial task, though.

The evaluation of a candidate function requires a validation corpus annotated with the correct senses (Navigli 2009). In our formulation, a validation corpus would be a set of n pairs $\{\langle c_i, s_i\rangle\}_{i=1}^n$, where $c_i \in C$, and $s_i = f^*_w(c_i) \in S_{w_i}$.

Usually, the evaluation criterion is *accuracy*, i.e, the fraction of correctly classified occurrences.

3. The Neuro-Evolutionary Approach

Artificial Neural Networks (ANNs) and Evolutionary Algorithms (EAs) are two well-defined computational models inspired by nature, belonging to so-called computational intelligence. The key difference between ANNs and EAs is that ANNs are based on the biological features of a natural neural network, while an EA deals with the adaptation of a population to a changing environment.

An ANN (Camargo 1990; Kröse and Van der Smagt 1996; Gurney 1997) is composed of simple computing units (the *neurons*) which are connected to form a neural network. Whether a neuron a influences another neuron b or not depends on the ANN structure. The extent of such an influence, when there is one, depends on the weight assigned to each connection between the neurons. It is very difficult to find a suitable network (structure and influence weights) for a given problem. In most cases, an appropriate structure is created by intuition.

EAs (De Jong 2002; Eiben and Smith 2003) are a broad class of stochastic optimization algorithms, inspired by biology and in particular by those biological processes that allow populations of organisms to adapt to their surrounding environment: genetic inheritance and survival of the fittest. Each individual of the population represents a point in the space of the potential solutions for the considered problem. The evolution is obtained by iteratively applying a (usually quite small) set of stochastic operators, known as *mutation*, *recombination*, and *selection*. Mutation randomly perturbs a candidate solution; recombination decomposes two distinct solutions and then randomly mixes their parts to form novel solutions; and selection replicates the most successful solutions found in a population at a rate proportional to their relative quality. The initial population may be either a random sample of the solution space or may be seeded with solutions found by simple local search procedures, if these are available. The resulting process tends to find, given enough time, globally optimal solutions to the problem much in the same way as in nature populations of organisms tend to adapt to their surrounding environment.

A large number of successful applications presented in the literature demonstrate that ANN design can be improved by synergicatelly combining it with evolutionary algorithms, obtaining what can be called Evolutionary Artificial Neural Networks (EANNs). As indicated by Yao (1999), an evolutionary algorithm represents an interesting and more integrated way of designing ANNs since it allows all aspects of ANN design to be taken into account at once. The evolutionary process handles the design optimization of a population of ANNs with respect to a particular problem, in which all the available information is given as input to the neural networks. In this approach, no expert knowledge of the problem is required, since the evolutionary algorithm is able to automatically find the best solution for that particular problem.

The overall algorithm is based on the joint optimization of structure and weights, here briefly summarized, and the error backpropagation algorithm (BP) is used in the network learning phase; a more complete and detailed description can be found in the literature (Azzini and Tettamanzi 2008b). The *indirect encoding* is used to 'decode' a *genotype* into a *phenotype* ANN. Accordingly, it is the genotype which undergoes the genetic operators and which reproduces itself, whereas the phenotype is used *only* for calculating the genotype's fitness. The rationale for this choice is that the alternative of using BP, applied to the genotype, as a kind of 'intelligent' mutation operator, would boost exploitation while impairing exploration, thus making the algorithm too prone to being trapped in local optima. An example of an individual genotype is represented in Figure 1.

Thanks to this encoding, individual ANNs are not constrained to a pre-established topology. Differently from NEAT (Stanley and Miikkulainen 2002), that starts with minimal network topologies and then applies evolutionary mechanisms to augment them, our approach randomly initializes the network's population with different hidden layer sizes and different numbers of neurons for each individual according to two exponential distributions, in order not to constrain search and provide a balanced mix of topologies. Such dimensions are not bounded in advance, even though the fitness function may penalize large networks. The number of neurons in each hidden layer is constrained to be greater than or equal to the number of network outputs, in order to avoid hourglass structures, whose performance tends to be poor.

3.1 ANN Encoding

Each individual in the population represents a multilayer perceptron (MLP). MLPs are feedforward neural networks with a layer of input neurons, a layer of one or more output neurons and zero or more 'hidden' (i.e., internal) layers of neurons in between; neurons in a layer can take inputs from the previous layer only.

The genotype encodes the number of hidden layers and the number of neurons for each hidden layer. We call this the *topology vector*. The input and output layers are identical for all the neural networks for a given task, but the size of the output layer depends on the number of the output classes for each benchmark dataset.

The number of hidden nodes in the ith hidden layer corresponds to the number specified in the ith element in the topology vector (see Figure 1, the vector above the graph); furthermore, the chromosome of an individual encodes the connection weights of the corresponding MLP (see Figure 1, the reported graph). All nodes in each layer are connected to all nodes in the next layer as specified by the corresponding weight matrix, $\mathbf{W}$, defined for each pair of layers. Also, for each layer, the bias vector b is defined, which contains, in each element, the bias value of the corresponding node in the neural network.

Initial weights and biases are extracted from a normal distribution. Like in evolution strategies (Bäck, Hoffmeister, and Schwefel 1991), for each connection weight and neuron bias encoded in the genotype, an associated variance is also encoded, which determines the probability distribution of mutations and is used to self-adapt the mutation step.

The network topology is affected by the genetic operators during evolution, in order to perform both incremental (adding hidden neurons or hidden layers) and decremental (pruning hidden neurons or hidden layers) learning.

3.2 Evolutionary Process

In the evolutionary process the genetic operators are applied to each network until the termination conditions are not satisfied. At each generation, the first half of the popula-

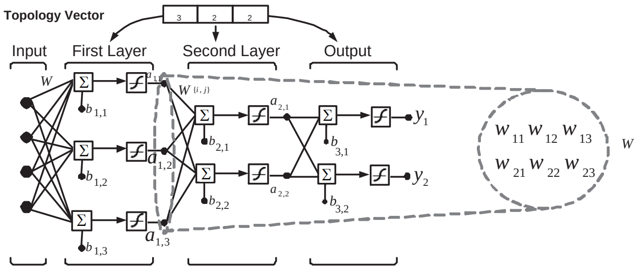

Figure 1
Representation of the ANN. The Topology vector reports the number of hidden nodes for each hidden layer and for the output layer. The graph explains all the connections among neural nodes and the node's biases. The matrix W shows the details of weights between two network's layers.

tion, i.e. the best $\lfloor n/2 \rfloor$ individuals, is selected by truncation from a population of size n, while the second half of the population is replaced by the offspring generated through the crossover operator. Crossover is then applied to two individuals selected from the best half of the population (parents), with a probability parameter p_{cross}, defined by the user together with all the other genetic parameters, and maintained unchanged during the entire evolutionary process.

Elitism allows the survival of the best individual unchanged into the next generation and the solutions to get better over time. Then, for all individuals of the population the algorithm mutates the weights and the topology of the offspring, trains the resulting network, calculates fitness, and saves the best individual and statistics about the entire evolutionary process.

The general framework of the evolutionary process can be described by the following pseudo-code. Individuals in a population compete and communicate with other individuals through genetic operators applied with independent probabilities, until termination conditions are not satisfied.

```
1.   Initialize the population by generating new random individuals.
2.   Create for each genotype the corresponding MLP, and calculate its cost
     and its fitness values.
3.   Save the best individual as the best-so-far individual.
4.   While not termination condition do
  (a)   Apply the genetic operators to each network.
  (b)   Decode each new genotype into the corresponding network.
  (c)   Compute the fitness value for each network.
  (d)   Save statistics.
```

The application of the genetic operators to each network is described through this pseudo-code, followed by a brief description of such genetic operators.

```
1.   Select from the population (of size $n$) $\lfloor n/2 \rfloor$ individuals by truncation and
     create a new population of size $n$ with copies of the selected individuals.
```

2. For all individuals in the population:
   (a) Randomly choose two individuals as possible parents.
   (b) Check their local similarity and apply crossover according to the crossover probability.
   (c) Mutate the weights and the topology of the offspring according to the mutation probabilities.
   (d) Train the resulting network using the training set.
   (e) Calculate the fitness f on the test set.
   (f) Save the individual with lowest f as the best-so-far individual if the f of the previously saved best-so-far individual is higher (worse).
3. Save statistics.

For each generation of the population, all the information of the best individual is saved.

Truncation selection, the selection method implemented in this work, is taken from the breeder genetic algorithm (Muhlenbein and Schlierkamp-Voosen 1993), and differs from natural probabilistic selection in that evolution only considers the individuals that best adapt to the environment. Truncation selection is not a novel solution and other previous works considered such a selection in order to prevent the population from remaining too static and perhaps not evolving at all (Goldberg 1989). It is a very simple technique which produces satisfactory solutions in conjunction with other strategies, like elitism, which allows the best individual to survive unchanged into the next generation and solutions to monotonically get better over time.

One of the most important aspects of the application of the crossover operator in neuro-evolutionary techniques is that, starting with a set of random solutions, it becomes helpful in finding optimal solution in the search space, by avoiding local monima entrapment. The recombination implemented in this approach is based on the `SimBa` crossover (Azzini, Dragoni, and Tettamanzi 2010), which works by looking for a 'local similarity' between two individuals selected from the population. If such a condition is satisfied the layers involved in the crossover operator are defined. The contribution of each neuron of the layer selected for the crossover is computed, and the neurons of each layer are reordered according to their contribution. Then, each neuron of the layer in the first selected individual is associated with the most 'similar' neuron of the layer in the other individual, and the neurons of the layer of the second individual are re-ranked by considering the associations with the neurons of the first one. Finally a cut-point is randomly selected and the neurons above the cut-point are swapped by generating the offspring of the selected individuals. Such an operator allows recombination of individuals that have different topologies, but with hidden nodest that are similarly performing in the cutting point, avoiding the well known disruptive effect over the considered ANNs.

After crossover, mutation is applied to the evolving population. The main function of this operator is to introduce new genetic material and to maintain diversity in the population. Generally, the purpose of mutation is to simulate the effect of transcription errors that can occur with a very low probability, the mutation rate, when a chromosome is duplicated. The evolutionary process applies two kinds of neural network perturbations: weights mutation and topology mutation.

Weights mutation perturbs the weights of the neurons before performing any structural mutation and applying the BP to train the network. All the weights and the corresponding biases are updated by using variance matrices and evolutionary strategies applied to the synapses of each NN, in order to allow a control parameter, like mutation variance, to self-adapt rather than changing their values by some deterministic algorithm. This solution is similar to the approach implemented by *evolution strategies* (Schwefel 1981), algorithms in which the strategy parameters are proposed for self-adapting the mutation concurrently with the evolutionary search. The main idea behind these strategies is to allow a control parameter, like mutation variance, to self-adapt rather than changing its value by some deterministic algorithm. Evolution strategies perform very well in numerical domains, and are well-suited to (real) function optimization.

Topology mutation is implemented with four types of mutation by considering neurons and layer addition and elimination by setting the corresponding parameters p_{+layer}, p_{-layer}, $p_{+neuron}$ and $p_{-neuron}$. It is implemented after weight mutation because a perturbation of weight values changes the behavior of the network with respect to the activation functions. The addition and the elimination of a layer and the insertion of a neuron are applied with three independent probabilities, while the elimination of a neuron is carried out only if the contribution of that neuron is negligible with respect to the overall network output (Azzini and Tettamanzi 2008a). All the topology mutation operators are aimed at minimizing their impact on the behavior of the network; in other words, they are designed to be as little disruptive, and as much neutral, as possible, preserving the behavioral link between the parent and the offspring better than by adding random nodes or layers.

Finally, although it is customary in EAs to assume that better individuals have higher fitness, the convention that a lower fitness means a better ANN is adopted in this work. This maps directly to the objective function of an error- and cost- minimization problem, which is the natural formulation of most problems ANNs can solve.

In the case of WSD, the fitness of an individual is defined as a function of the confusion matrix M obtained by that individual,

$$f(M) = N_{\text{outputs}} - \text{Trace}(M), \tag{2}$$

where N_{outputs} is the number of output neurons (i.e., the number of senses) and $\text{Trace}(M)$ is the sum of the diagonal elements of a row-wise normalized confusion matrix, which represent the conditional probabilities of the predicted outputs given the actual ones.

Following the commonly accepted practice of machine learning, the problem data are partitioned into three sets: training, validation, and test set, used respectively to train, to stop the training, thus avoiding overfitting, and to assess the generalization capabilities of a network. In this method, the fitness is calculated for each individual on the validation set after applying BP.

4. The Proposal

This section describes the approach to WSD proposed in this article. We start by providing an outline of the approach, then examine each constituent thereof in greater detail.

4.1 Outline

The approach we describe here is a supervised method, using annotated corpora to train ANNs by means of an evolutionary algorithm.

Each ANN is trained to disambiguate a specific word. Therefore, the method provides for a specialized ANN disambiguator for each individual polysemous word in the target language.

The method consists of two phases:

1. an off-line *training* phase, whereby an ANN disambiguator is evolved for each polysemous word by using a word-specific annotated corpus;
2. an on-line *application* phase, whereby the appropriate ANN disambiguators are used to disambiguate instances of polysemous words occurring in texts.

For each given polysemous word w, the relevant annotated corpus consists of a set of sentences (i.e., contexts) where an instance of w occurs. A training and a validation datasets are extracted from such corpus by converting each context into a vector of numerical features, which represents a compressed encoding of the context. Two connectionist encoding schemes have been tested for that purpose, both based on the orthography of the words in the context, not on their meanings or semantic connections with w, which lead to feature vectors of relatively low dimension. This, in turn, limits the number of input neurons the ANN disambiguators must have, and makes their training computationally tractable and their application efficient.

The proposed method is designed for a modular text disambiguation system which uses a database of word-specific ANN disambiguators. Each disambiguator, which takes only a few kilobytes of disk space, is retrieved and applied on demand whenever an instance of the word to disambiguate is encountered. According to this architecture, the disambiguators for all polysemous words in the target language must be trained offline before the system can be used. However, new disambiguators can be added while the system is operational and existing disambiguators can be retrained or tuned at any moment to improve the performance of the system.

4.2 Corpus

Since the system implemented, as explained in Section 4.1, uses supervised learning, it needs a corpus to train each neural network created by the algorithm. The corpus has to be composed, for each word that has to be disambiguated, by a set of sentences, each containing the target word and information about the correct sense of the target word in that sentence.

For the purpose of implementing a system that disambiguates all polysemous words, one difficulty is to find a corpus that allows us to have this information for each word. As a solution, we have decided to use IXA Group's web corpus (Agirre and de Lacalle 2004), which comprises all WordNet senses of a selection of nouns for which its construction method is applicable. The construction method is inspired by the "monosemous relatives" method (Leacock, Chodorow, and Miller 1998). This method usually relies on information in WordNet in order to retrieve examples, from large corpora or from the web, containing the monosemous synonyms, hyponyms, hypernyms and siblings of a target polysemous word. With reference to the structure of WordNet, synonyms are words that have the same synset as the target word; hypernyms are generic terms used to designate a whole class of specific instances: Y is a hypernym of X if X is a kind of Y; hyponyms are specific terms used to designate a member of

Table 1
A summary of the corpus structure for the word *memory*.

Sense	Synonyms	Direct Hyponyms	Hypernyms, Indirect Hyponyms (dist. 2)	Siblings, Indirect Hyponyms (dist. 3)
1. something that is remembered	-	engram, screen memory, memory trace	internal representation, mental representation	concrete representation, concretism, mental image, percept, perceptual experience, phantasmagoria, psychosexuality, unrealism
2. the cognitive processes whereby past experience is remembered	remembering	immediate memory, long-term memory, ltm, retrospection, short-term memory, stm, working memory	basic cognitive process	apperception, believing, inattention, representational process
3. the power of retaining and recalling past experience	-	-	mental faculty	sensory faculty, sentiency
4. an electronic memory device	computer memory, computer storage, memory board	fixed storage, random access memory, random memory, random-access memory, readwrite memory, read-only memory, read-only storage, real storage, rom, scratchpad, virtual memory, virtual storage	memory device, storage device	acoustic storage, auxiliary storage, buffer storage, buffer store, external storage, mag tape, magnetic disc, magnetic disk, magnetic tape, push-down storage, push-down store, secondary storage
5. the area of cognitive psychology that studies memory processes	-	-	cognitive psychology	psycholinguistics

a class: X is a hyponym of Y if X is a kind of Y; finally, siblings are sister terms of the target word.

The advantage of using this corpus instead of other traditional corpora specifically designed for WSD is that, due to the way the corpus is constructed, we get an extensive coverage of word usage in all domains. Furthermore, since the corpus is not tagged by hand, it gives wider guarantees of accuracy and objectivity.

In Table 1, an example of the corpus structure is shown. For instance, the word *memory* has 5 senses, represented in the first column. For each sense, the corpus contains a set of snippets related to each word listed in the same row of the sense. For instance, Sense 2 has the monosemous synonym *remembering*. Therefore, it is possible to find in the corpus a set of sentences containing the word *remembering*. Contexts in which a determined sense occurs are then easily identified. This set is present for each word listed in the table.

From this web corpus, we extracted all data relative to the words listed in Table 5, which reports the number of senses. For each of these words, 50% of the records have been used for training, 25% for validation, and 25% for testing.

4.3 Context Encoding

A critical problem in supervised approaches to WSD is how to represent the context in which a word is used. In our case, such representation should be specifically targeted to its use with neural networks.

Two kinds of representations commonly used in connectionism are distributed (Hinton, McClelland, and Rumelhart 1986) and localist schemes (Cottrell 1989).

The major drawback of the latter is the high number of inputs needed to disambiguate a single sentence, because every input node has to be associated to every possible sense; indeed, this number increases staggeringly if one wants to disambiguate an entire text.

Starting from this consideration, the hypothesis of using an association between nodes and all the different concepts or words in a text was discarded.

Examples of distributed schemes are microfeatures (Waltz and Pollack 1985) and the word-space model (Schütze 1993). Distributed schemes are very attractive in that they represent a context as an activation pattern over the input neurons. Therefore, unlike with localist schemes, the number of input neurons required does not have to equal the size of the vocabulary. On the contrary, there is an intrinsic idea of *compression*, whereas each input neuron encodes a given *feature* of a word or sentence and thus can be reused to represent many different words or senses. Of course, the more the input patterns are compressed into a low-dimensional space, the more information is lost. However, despite such information loss, enough information may still be there to allow meaningful processing by the ANN.

We used three distributed representation schemes, all based on the orthography of words, corresponding to three different degrees of compression:

1. a *positional* scheme (Section 4.3.1), whereby 156 input neurons, divided in six groups of 26 input neurons each;
2. a *lexicographic* scheme (Section 4.3.2), whereby 45 input neurons, one for each lexicographic categories based on syntactic category and logical groupings that are defined in WordNet;
3. a *POS-tagged lexicographic* scheme (Section 4.3.3), it is derived from the previous one, with the difference that each word is with the part-of-speech element that represent the role of the word in the sentence, for example, as a noun, verb, adjective, and so on.

In all cases, the activations of the input neurons are obtained by summation of the activation patterns representing the words occurring in a given context, excluding the target word, after removing stop words and stemming the remaining words.

Stemming is required in order to generalize beyond surface words by extracting lemmas, which abstract away from morphological variation, as is customary in WSD (Yarowsky and Florian 2002). The stemming algorithm we use may be regarded as an effective heuristics to get to world lemmas without performing a full-fledged morphological analysis, which is computationally more expensive. This extreme simplification is the essence of our proposal, which makes it alternative, and complementary, to other approaches.

In the current version of our work, we remove stop words, such as articles, pronouns, and other particles, since this type of words occurs very frequently and does not always carry useful information about the context in which the target word is used. The implemented algorithm takes also in account the compound names and verbs in order to increase the stemming accuracy.

Additional fields of the training set (one for each output neuron) correspond to the n senses of the target word. They are all set to zero except the one corresponding to the correct sense.

For example, if we consider the target word *tunnel*, which has two senses, namely (1) "a passageway through or under something" and (2) "a hole made by an animal", starting from a hypotetical training sentence '*The tunnel was part of an aqueduct system.*', related to Sense 1, stopword elimination and stemming would yield the list `(PART AQUEDUCT SYSTEM)`.

In the following section we show how the encoding schemes are implemented and which is the representation of this example by applying them.

4.3.1 Positional Encoding Scheme. In this encoding scheme, each word is reduced to a six letters representation by deleting as many vowels, starting from the last one but except the first one, as required to reduce the word to six letters, because consonants carry a heavier distinctive load; if, even after deleting all the vowels but the first, the word is still more than six letter long, only the first six letters are actually represented (thus *representation* would be encoded as `REPRSN`, but *cotton* would be encoded as `COTTON`); if the word is shorter than six letters, the representation is padded with blanks;

By considering the previous example, after reducing all words to their 6-letter representatives '`PART  `', '`ACQDCT`', and '`SYSTEM`', the input given to the neural networks will be the following:

$$
\begin{matrix}
\overbrace{\begin{matrix}1&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\1&0&0&1&0\\0&0&0&0&0\\&&0&&\end{matrix}} &
\overbrace{\begin{matrix}1&0&1&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&1\\&&0&&\end{matrix}} &
\overbrace{\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&1&1&1&0\\0&0&0&0&0\\&&0&&\end{matrix}} \\
\overbrace{\begin{matrix}0&0&0&1&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&2\\0&0&0&0&0\\&&0&&\end{matrix}} &
\overbrace{\begin{matrix}0&0&1&0&1\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\&&0&&\end{matrix}} &
\overbrace{\begin{matrix}0&0&0&0&0\\0&0&0&0&0\\0&0&1&0&0\\0&0&0&0&1\\0&0&0&0&0\\&&0&&\end{matrix}} \\
& \rightarrow 1\ 0, &
\end{matrix}
$$

where the numbers on the left-hand side of the arrow represent the occurrences of the letters in the six positions, i.e., `A1`, ..., `Z6`, here displayed according to the template

$$
\overbrace{\begin{matrix}A1&B1&C1&D1&E1\\F1&G1&H1&I1&J1\\K1&L1&M1&N1&O1\\P1&Q1&R1&S1&T1\\U1&V1&W1&Z1&Y1\\&&Z1&&\end{matrix}} \quad
\overbrace{\begin{matrix}A2&B2&C2&D2&E2\\F2&G2&H2&I2&J2\\K2&L2&M2&N2&O2\\P2&Q2&R2&S2&T2\\U2&V2&W2&Z2&Y2\\&&Z2&&\end{matrix}} \quad
\overbrace{\begin{matrix}A3&B3&C3&D3&E3\\F3&G3&H3&I3&J3\\K3&L3&M3&N3&O3\\P3&Q3&R3&S3&T3\\U3&V3&W3&Z3&Y3\\&&Z3&&\end{matrix}}
$$

$$\overbrace{\begin{array}{ccccc} \texttt{A4} & \texttt{B4} & \texttt{C4} & \texttt{D4} & \texttt{E4} \\ \texttt{F4} & \texttt{G4} & \texttt{H4} & \texttt{I4} & \texttt{J4} \\ \texttt{K4} & \texttt{L4} & \texttt{M4} & \texttt{N4} & \texttt{O4} \\ \texttt{P4} & \texttt{Q4} & \texttt{R4} & \texttt{S4} & \texttt{T4} \\ \texttt{U4} & \texttt{V4} & \texttt{W4} & \texttt{Z4} & \texttt{Y4} \\ & & \texttt{Z4} & & \end{array}} \quad \overbrace{\begin{array}{ccccc} \texttt{A5} & \texttt{B5} & \texttt{C5} & \texttt{D5} & \texttt{E5} \\ \texttt{F5} & \texttt{G5} & \texttt{H5} & \texttt{I5} & \texttt{J5} \\ \texttt{K5} & \texttt{L5} & \texttt{M5} & \texttt{N5} & \texttt{O5} \\ \texttt{P5} & \texttt{Q5} & \texttt{R5} & \texttt{S5} & \texttt{T5} \\ \texttt{U5} & \texttt{V5} & \texttt{W5} & \texttt{Z5} & \texttt{Y5} \\ & & \texttt{Z5} & & \end{array}} \quad \overbrace{\begin{array}{ccccc} \texttt{A6} & \texttt{B6} & \texttt{C6} & \texttt{D6} & \texttt{E6} \\ \texttt{F6} & \texttt{G6} & \texttt{H6} & \texttt{I6} & \texttt{J6} \\ \texttt{K6} & \texttt{L6} & \texttt{M6} & \texttt{N6} & \texttt{O6} \\ \texttt{P6} & \texttt{Q6} & \texttt{R6} & \texttt{S6} & \texttt{T6} \\ \texttt{U6} & \texttt{V6} & \texttt{W6} & \texttt{Z6} & \texttt{Y6} \\ & & \texttt{Z6} & & \end{array}} .$$

In the first position, the 6-letter representatives have one `A`, one `P`, and one `S`. Therefore, $\texttt{A1} = \texttt{P1} = \texttt{S1} = 1$. In the fourth position, one `D` and two `T`s are found. Therefore, $\texttt{D4} = 1$ and $\texttt{T4} = 2$, and similarly for the other positions.

4.3.2 Lexicographic Encoding. In this encoding scheme we define the context in which a particular word occurs. To create a lexicographic representation of a word we use the lexicographic annotation that WordNet assigns to each word sense — each synset is classified into one of forty-five lexicographic categories based on syntactic category and logical groupings. An example of lexicographic categories is shown in Table 2.

Table 2
Example of Lexicographic Categories

Lexicographic Category	Category Description
noun.artifact	nouns denoting man-made objects
noun.location	nouns denoting spatial position
noun.process	nouns denoting natural processes

For each word that occurs in a sentence, every associated synset has been extracted and, for each synset, the related lexicographic information is considered. The context of a word w is then represented as a vector of the forty-five lexicographic categories, and the elements of such a vector correspond to the contribution of the instances of the other words in the sentence to the corresponding category. The so defined context is then given in input to the neural network.

Formally, the contribution $C_k(w)$, of an instance of word w to the k-th component of the context vector C is calculated as follows:

$$C_k(w) = \frac{N_k(w)}{N(w)} \tag{3}$$

where $N_k(w)$ is the number of synsets of w whose category is k, and $N(w)$ is the number of synsets of word w. As we can see, the contribution of a monosemous word is maximal, i.e., it is 1.0.

Let S be the sentence in which the word to be disambiguated w occurs. The k-th element of the vector context C of S, C_k, is given by:

$$C_k = \sum_{w \in S} C_k(w). \tag{4}$$

For example, starting from the example sentence '*part aqueduct system*', the contribution to each input neuron $(C_1, \ldots, C_{45})$ is calculated as shown in Table 3, where the number in parenthesis is the number of instances of the lexicographic category in the sense list of each word.

Table 3
Input for the sentence "part aqueduct system"

Word	Lexicographic Category	Contribution	Word	Lexicographic Category	Contribution
part		(18 senses)	aqueduct		(1 sense)
	<noun.act>	0.110 (2)		<noun.artifact>	1.000
	<noun.artifact>	0.055 (1)			
	<noun.body>	0.055 (1)	system		(9 senses)
	<noun.cognition>	0.175 (3)		<noun.artifact>	0.111 (1)
	<noun.communication>	0.055 (1)		<noun.attribute>	0.111 (1)
	<noun.location>	0.055 (1)		<noun.body>	0.222 (2)
	<noun.object>	0.055 (1)		<noun.cognition>	0.334 (3)
	<noun.possession>	0.055 (1)		<noun.group>	0.111 (1)
	<noun.relation>	0.055 (1)		<noun.substance>	0.111 (1)
	<verb.motion>	0.110 (2)			
	<verb.social>	0.055 (1)			
	<verb.contact>	0.110 (2)			
	<adv.all>	0.055 (1)			

4.3.3 POS-Tagged Lexicographic Encoding. The POS-Tagged encoding aims to increase the accuracy with which the lexicographic information associated to each word is used to represent the context.

The POS tagging consists in the assignment of speech tags to each word (token) of a sentence in order to describe its corresponding role. Each word may be tagged, for example, as a noun, verb, adjective, and so on, and by applying a tagger, the sentence "Part of an aqueduct system" is then tagged into: "Part/NN of/IN an/DT aqueduct/NN system/NN". The tagging process has been performed by using the Stanford POS Tagger (Toutanova et al. 2003), and the tags used are chosen according to the Penn Treebank POS tagset (Marcus, Santorini, and Marcinkiewicz 1993).

For example, the word "part", according to the WordNet dictionary, has not only different meanings, but also different possible parts of speech in a sentence since it can be used as noun, verb, or adverb. A word representation in the sentence, without using a tagging procedure, will contain an error due to possible improper parts of speech.

To create the POS-tagged lexicographic representation of a word we have used the same lexicographic annotation shown in Table 2, therefore the POS tags are used to consider only the senses that are compatible with the tag assigned to each word. According to the previous example, the word "part" has 18 senses, however, only 12 refer to the noun part of speech of the word "part", therefore, only those 12 senses will be considered and introduced in the word representation.

5. Experiments and Results

A computational framework for evolving neural disambiguators according to the proposed approach has been developed by the authors and experiments have been carried out to assess the viability of the method described in the previous section.

5.1 Protocol

Experiments have been performed in two phases, each with a distinct purpose:

Table 4
Input for the sentence “part aqueduct system”

Word	Lexicographic Category	Contribution	Word	Lexicographic Category	Contribution
part		(18 senses)	aqueduct		(1 sense)
	<noun.act>	0.167 (2)		<noun.artifact>	1.000
	<noun.artifact>	0.083 (1)			
	<noun.body>	0.083 (1)	system		(9 senses)
	<noun.cognition>	0.250 (3)		<noun.artifact>	0.111 (1)
	<noun.communication>	0.083 (1)		<noun.attribute>	0.111 (1)
	<noun.location>	0.083 (1)		<noun.body>	0.222 (2)
	<noun.object>	0.083 (1)		<noun.cognition>	0.334 (3)
	<noun.possession>	0.083 (1)		<noun.group>	0.111 (1)
	<noun.relation>	0.083 (1)		<noun.substance>	0.111 (1)
	<verb.motion>	0.000 (2)			
	<verb.social>	0.000 (1)			
	<verb.contact>	0.000 (2)			
	<adv.all>	0.000 (1)			

1. a *comparative assessment* phase, aimed at comparing the three connectionist encoding schemes described in Section 4.3, as well as the neuro-evolutionary algorithm described in Section 3 with different baselines;
2. a *performance test* phase, whose purpose was to compare the overall performance of the proposed method to the state of the art of WSD algorithms.

In the first phase the IXA Group’s web corpus (see Section 4.2) has been used for training the ANN disambiguators; while in the second phase we performed two different tests in order to test the effectiveness of our disambiguators by training them with the IXA web corpus for the first test and with the Semeval[1] 2007 benchmark for the second test.

In this last phase, to test the effectiveness of the proposed approach, we have compared its performance to the best state-of-the-art WSD algorithm presented at Semeval 2007, an evaluation exercise organized by ACL-SIGLEX every three years. The purpose of Semeval is to evaluate the strengths and weaknesses of WSD programs with respect to different words, languages and their varieties. The most recent edition, at the time of this writing, was Semeval 2010, and included 18 different tasks targeting the evaluation of systems for the semantic analysis of text. However, in the Semeval 2010 edition was not defined a task that is suitable to apply our corpus; therefore, we switched to the second-last edition of Semeval (Semeval 2007) in which the Task 17 (Subtask 1), consisting of coarse disambiguation of nouns and verbs, was selected as the one best fitting the aim and scope of the approach described in this article.

For our experiments we selected the set of 30 nouns used in this Task showed in Table 5.

[1] Further information on the Semeval initiative is available on the WWW at the URL “`http://www.senseval.org/`”.

Table 5
A summary of the information and the accuracies used as baseline for the words used for the system testing. The last row contains the overall results, each word accuracy has been weighted over the test set size.

Word	# of Senses	Avg Dist.	Test Set Size	Word	# of Senses	Avg Dist.	Test Set Size
area	5	10	8252	management	2	10	841
authority	6	12	8431	network	4	7	4050
base	12	13	6822	order	9	9	5746
bill	8	13	3338	part	7	9	-
carrier	10	11	3459	people	3	4	3608
chance	4	10	517	point	13	10	12814
condition	4	8	10135	policy	2	14	1287
defense	8	10	5072	position	6	8	13257
development	3	9	3910	power	4	9	8256
drug	2	-	-	president	3	11	1653
effect	4	9	3152	rate	2	8	3696
exchange	6	10	4189	source	6	10	4557
future	3	9	915	space	6	8	3324
hour	4	9	4619	state	4	11	20315
job	10	11	4225	system	7	9	8366

5.2 Neuro-Evolutionary Algorithm Set-Up

Previous experiences in applying the neuro-evolutionary algorithm to a variety of problems in several domains (see Section 3) have clearly shown that the algorithm is quite robust with respect to the parameter setting. In particular, a setting of the three parameters governing the mutation operator that has proved to be a good starting point in all previous applications is $p^+_{\text{layer}} = p^-_{\text{layer}} = p^+_{\text{neuron}} = 0.1$. Nevertheless, in the first phase, several experiments have been carried out in order to find out the optimal settings for these three parameters. For each of them, all combinations of the values in the set $\{0.05, 0.1, 0.2\}$ have been tried. In this work, only a limited set of values has been used, due to the lack of available time/resources, since each run of the neuro-evolutionary algorithm is computationally very expensive.

For each setting of these parameters, 10 runs of the EA on the word *memory* have been carried out. For each run, up to 250,000 network evaluations (i.e., applications of the network to the whole training set) have been allowed, including those performed by the BP algorithm. The results of this phase are not reported in detail here, because their interest is merely technical in deciding which parameter setting would likely give the best performance. The best experimental solutions have been found for p^+_{layer}, p^-_{layer} and p^+_{neuron} all equal to 0.05. However, they did not differ dramatically from other settings, and the best settings found for *memory* have been used for evolving ANN disambiguators for a representative set of polysemous words. It should be noted that this is an important aspect, since it indicates the robustness of the neuro-evolutionary algorithm with respect to variations of its parameters.

5.3 Comparative Assessment

In this phase, we compare the three connectionist encoding schemes of Section 4.3. At the same time, the performance of the neuro-evolutionary algorithm has been compared with different baselines:

- the most frequent sense (MFS), that is known to be one of the hardest baselines to beat for WSD algorithms;
- the k-nearest neighbor (k-NN) clustering algorithm, in which the training set is used to compute the centroid associated with each sense. Then, the distance between each instance of the test set with each centroid is computed in order to show the accuracy of the generated clusters;
- the accuracy of a simple (i.e., not evolved) neural network, in order to measure the improvement the use of an EA brings about;
- the performance of NEAT (Stanley and Miikkulainen 2002), in order to measure the performance of the evolutionary algorithm with respect to the well known tate-of-the-art of evolutionary approaches applied to neural networks.

For these experiments, the training, validation, and test sets have been extracted from the IXA Group's web corpus. In Table 6 we present the overall results computed as the average of the accuracies obtained on each word, while in Tables 8, 9, and 10 we show the detailed results obtained on each word by each encoding scheme. Moreover, in Table 7 we show the results of the T-test in order to verify the statistical significance of the obtained accuracies.

Table 6
A summary of overall results computed as the average of the accuracies obtained on each word. Training and test are calculated over the IXA Corpus.

Encoding Scheme	Accuracies				
	MFS	K-NN	Simple ANN	NEAT	NeuroGEN
Positional	55.16	24.69	56.22	59.16	**60.32**
Lexicographic	55.16	24.59	56.69	66.36	**68.57**
POS-Tagged Lexicographic	55.16	24.03	57.47	69.65	**73.10**

Table 7
The results of the significance T-test applied to the accuracies showed in Table 6. Each element represents, for each encoding scheme, the statistical significance of the difference between the accuracy obtained from our neuro-genetic approach and that obtained from the baselines.

Encoding Scheme	T-test Results			
	MFS	K-NN	Simple ANN	NEAT
Positional	70.04%	100.00%	59.10%	18.62%
Lexicographic	99.30%	100.00%	98.36%	36.12%
POS-Tagged Lexicographic	99.97%	100.00%	99.84%	54.64%

In Table 6 the columns show the averaged values of the accuracies obtained by applying, respectively, the MFS, the K-NN algorithm, the ANN without evolution, the NEAT tool and, finally, the proposed neuro-genetic approach. On each row we report the adopted encoding scheme.

In general, the superiority of the pos-tagged encoding scheme over the other two schemes is evident for all the approaches considered as baseline (obviously excepted for the MFS). We may observe that the improvement of the neuro-genetic approach with respect to such baselines is directly proportional to the increment of the quantity of information, used to represent the context of each sentence. Indeed, the pos-tagged encoding scheme considers more information than the positional one, and this difference is reflected on the obtained accuracies.

The superiority of the neuro-genetic algorithm over the baselines is confirmed for all the encoding scheme. To demostrate the statistical significance of such improvements we performed the T-test with a confidence value of 95% over the differences between the accuracies obtained by the neuro-genetic approach and those obtained by the other approaches.

The positional encoding scheme obtains significant results only with respect to K-NN algorithm; instead, the encoding schemes based on lexicographic information obtain significant results with respect to all baselines except for the NEAT algorithm. However, for all words, the neuro-genetic approach obtains better results with respect to the NEAT algorithm, and, even if the test returns a 55% of significance, we may claim that the neuro-genetic is more effective than the NEAT algorithm.

By observing the detailed results obtained on each word (Tables 8, 9, and 10), we can notice how the neuro-genetic approach always obtains better accuracies with respect to the baselines, even if, in the positional encoding scheme, the differences between the baselines (except for K-NN) and the neuro-genetic approach are not particularly evident. Such an aspect is also confirmed by the low statistic significance of the results for the positional encoding scheme showed in Table 7.

However, it is possible to notice how the use of POS-tagging to refine the lexicographic encoding scheme considerably improves the performance of the neuro-genetic approach, by obtaining an accuracy higher than 80% for 10 words.

5.4 Performance Test

In order to verify the effectiveness of the presented approach we performed a further test by using the Semeval benchmarks introduced above. For each of the 30 words listed in Table 5, detailed results are provided by the organizers of Semeval 2007, indicating the performance of the best eight systems (Pradhan et al. 2007). These systems are representative of state-of-the-art WSD techniques, in that they are based on several distinct methods. More details about it, as well as its competitors, may be found in (Pradhan et al. 2007).

In this test we compare the performance of our approach with the system that obtained the best average accuracy at the Semeval 2007. Our aim is not to outperform that system, but to show that the performances of the proposed approach are comparable with the state-of-the-art even if our system is trained by using different data and without tuning it with ad-hoc parameters for the Senseval 2007 dataset.

Since the rules for Task 17 allowed the use of any available resource for training, we used training and validation datasets extracted from the IXA Group's web corpus for all words except for *drug* and *part*: the word *drug* is monosemous in WordNet, while in Task 17 it has two distinct senses; the word *part* is not included in the IXA Group's web corpus.

For all words, the evolved ANNs have been scored by using two different training sets:

1. the training set extracted from the IXA Group's corpus;
2. the training set extracted from the Semeval 2007 Task 17 dataset.

Table 8
A summary of the results of applying the positional encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words, with a comparison to the baselines.

Word	MFS	K-NN	Simple ANN	NEAT	EANN POS
area	76.16	29.14	76.65	78.32	78.78
authority	57.18	21.52	58.53	61.08	62.16
base	29.17	14.13	30.71	35.50	37.50
bill	30.35	15.85	32.06	36.59	38.87
carrier	19.50	12.19	21.47	26.67	28.49
chance	42.36	21.09	43.63	47.44	48.95
condition	93.43	41.35	93.63	94.01	94.19
defense	35.21	18.47	36.79	40.88	42.70
development	81.51	36.44	81.95	83.17	83.61
effect	78.20	34.94	78.71	80.12	80.88
exchange	33.92	14.50	35.53	39.85	41.85
future	69.84	32.06	70.45	72.51	73.22
hour	59.02	25.28	60.16	62.69	63.75
job	17.79	10.12	19.61	25.15	27.00
management	92.51	39.80	92.61	93.15	93.32
network	90.94	32.83	91.18	91.73	91.97
order	33.19	17.79	34.95	38.99	40.81
people	64.41	28.74	65.25	67.67	68.40
point	35.38	18.40	36.51	41.22	42.98
policy	64.65	25.96	65.48	67.75	68.73
position	48.46	22.36	49.67	53.00	54.19
power	73.52	34.04	74.17	75.94	76.48
president	81.31	35.61	81.72	82.99	83.30
rate	57.58	27.36	58.95	61.47	62.59
source	28.33	13.70	29.99	34.73	36.51
space	37.33	17.38	38.87	42.96	44.83
state	83.49	35.55	83.92	85.01	85.31
system	29.79	14.82	31.13	36.00	37.64

For both the evaluations, we have used the test set provided by the Semeval 2007 organizers.

The overall results of this experiment are shown in Table 11. We have performed all the experiments also by testing the NEAT system; the rational behind is that the approach presented at Senseval 2007 is tuned on the dataset used in such a competition. Therefore, a comparison with a more general-purpose (like NEAT) algorithm is necessary in order to prove the effectiveness of the proposed approach.

Generally our approach outperforms the NEAT algorithm by using all proposed encoding schemes, however, the obtained accuracies are lower than the ones obtained by the compared Semeval 2007 system.

However, by observing the detailed results obtained on each word (Tables 12, 13, and 14), we can notice that in some cases (words) the neuro-genetic approach performs better with respect to the Senseval 2007 system. The number of cases changes with respect to the different encoding schemes adopted, for instance, by using the pos-tagged encoding scheme, in 9 cases out of 30 the approach obtained a better performance with respect to the Senseval 2007 system, while by using the positional and the lexicographic encoding only in 2 cases.

The use of the two different datasets to train our approach brings up the question to what extent the performance of our evolved disambiguators is due to the specific choice of one training dataset with respect to another one. It is important to notice that the results obtained by using different training set are very similar, indeed, the accuracy

Table 9
A summary of the results of applying the lexicographic encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words, with a comparison to the baselines.

Word	MFS	K-NN	Simple ANN	NEAT	EANN LEX
area	76.16	28.25	77.05	82.11	83.43
authority	57.18	26.84	58.76	67.88	69.86
base	29.17	14.42	31.53	46.88	50.86
bill	30.35	15.43	32.42	47.75	51.43
carrier	19.50	11.83	22.22	39.64	43.34
chance	42.36	19.32	44.32	56.67	58.97
condition	93.43	37.46	93.63	95.07	95.31
defense	35.21	16.88	37.68	51.40	54.50
development	81.51	34.82	82.23	86.11	86.92
effect	78.20	30.40	78.92	83.63	84.75
exchange	33.92	17.63	35.74	50.44	53.71
future	69.84	29.43	71.05	77.38	78.99
hour	59.02	24.70	60.11	69.26	71.57
job	17.79	12.43	20.33	38.34	41.93
management	92.51	37.91	92.70	94.29	94.64
network	90.94	40.06	91.21	93.19	93.62
order	33.19	16.99	35.85	49.88	53.74
people	64.41	30.92	65.74	73.31	74.54
point	35.38	17.74	37.81	51.54	54.67
policy	64.65	27.80	65.96	73.43	75.47
position	48.46	22.60	50.21	61.34	63.61
power	73.52	31.93	74.47	80.14	81.38
president	81.31	30.43	81.89	85.96	86.82
rate	57.58	27.62	59.18	68.18	70.05
source	28.33	13.05	30.47	46.24	50.43
space	37.33	18.21	39.41	52.98	56.06
state	83.49	38.22	84.10	87.62	88.42
system	29.79	15.07	32.17	47.33	51.05

difference is always less than 0.2%. This means that the algorithm is stable with respect to the dataset used, the same happens by considering the NEAT algorithm.

6. Conclusions

A neuro-evolutionary approach to WSD has been presented which is based on three distributed encoding schemes that are very different from the usual means to represent the context in which a target word occurs. A comparison with the best entry in the coarse disambiguation of noun and verbs task of the 2007 Senseval evaluation of WSD systems suggests that the proposed approach can compete with state-of-the-art WSD systems.

At first sight, creating a single ANN for every ambiguous word might seem hardly practical or even infeasible. However, there are just 15,935 polysemous words out of the 117,798 WordNet entries. Evolving an ANN for disambiguating a polysemous word takes, on an ordinary desktop PC, two hours on average. Assuming some 30 PCs are available day and night, 45 days would be enough to evolve a ANN for each polysemous word. We estimate that the entire set of almost 16,000 ANNs would occupy 30 Mbytes of disk space. When disambiguating a document, a stored ANN would be recalled from the database and executed every time a polysemous word were encountered. Recalling a network can take a few milliseconds, whereas executing it is

Table 10
A summary of the results of applying the POS-tagged lexicographic encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words, with a comparison to the baselines.

Word	MFS	K-NN	Simple ANN	NEAT	EANN POS-TAGGED LEX
area	76.16	29.84	77.11	83.82	85.93
authority	57.18	23.18	59.39	71.07	74.07
base	29.17	15.67	31.96	52.24	55.45
bill	30.35	14.34	33.68	52.81	58.56
carrier	19.50	10.40	23.75	45.74	51.71
chance	42.36	17.28	45.42	60.86	65.52
condition	93.43	35.69	93.73	95.54	96.01
defense	35.21	18.22	38.12	55.86	61.01
development	81.51	35.94	82.44	87.51	88.77
effect	78.20	29.96	79.11	85.24	86.94
exchange	33.92	16.55	36.99	55.46	61.58
future	69.84	30.24	71.37	79.41	81.89
hour	59.02	23.97	61.59	72.20	75.29
job	17.79	11.27	22.48	44.44	51.16
management	92.51	38.50	92.82	94.89	95.49
network	90.94	34.31	91.46	93.87	94.51
order	33.19	13.74	36.72	54.82	59.93
people	64.41	28.56	66.17	75.95	79.13
point	35.38	18.86	38.93	56.34	61.79
policy	64.65	26.12	66.32	75.88	78.82
position	48.46	22.67	51.25	65.09	69.12
power	73.52	31.89	75.09	82.25	83.67
president	81.31	31.96	82.08	87.30	88.83
rate	57.58	27.19	60.21	71.09	74.41
source	28.33	16.27	32.42	51.45	56.17
space	37.33	18.74	40.54	57.81	62.69
state	83.49	35.33	84.19	88.77	90.44
system	29.79	16.26	33.76	52.55	57.89

Table 11
A summary of overall results computed as the average of the accuracies obtained on each word.

Encoding	Accuracies				
	Train IXA		Train SE 2007		
Scheme	NEAT	NeuroGEN	NEAT	NeuroGEN	Benchmark
Positional	68.51	69.25	68.48	69.23	89.83
Lexicographic	73.91	75.43	73.91	75.37	89.83
POS-Tagged Lexicographic	76.21	83.82	76.25	83.75	89.83

just a matter of microseconds. Therefore, the approach we propose can be considered realistic and feasible with state-of-the-art technology.

The advantage of using the IXA Group's corpus is that such a corpus, being constructed automatically, might constitute a possible remedy to the well known bottleneck problem in supervised approaches for WSD. As explained above, this choice does provide a slight advantage to the proposed approach, although comparison with using the Semeval training dataset revealed that advantage is not dramatic.

A point to emphasize is that the approach to WSD proposed in this article is intended to be *complementary* to linguistically or cognitively informed approaches. Its essence is the extreme simplification of the context encoding, which makes the evolved

Table 12
A summary of the results of applying the positional encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words and by evaluating the system on the Semeval 2007 test set, with a comparison to the best Semeval 2007 system.

Word	Train on IXA		Train on SE2007 Benchmark		
	NEAT	NeuroGEN	NEAT	NeuroGEN	Benchmark
area	72.03	72.03	71.35	72.43	89.00
authority	71.43	71.43	71.43	71.43	86.00
base	70.00	70.00	70.00	70.00	80.00
bill	77.45	77.89	77.45	77.79	99.00
carrier	71.43	71.43	71.43	71.43	71.00
chance	33.33	37.00	33.33	38.33	73.00
condition	76.47	77.35	76.47	77.94	91.00
defense	33.33	34.76	33.33	34.05	57.00
development	65.52	68.62	65.52	67.76	100.00
drug	-	-	86.96	86.96	96.00
effect	80.00	80.00	80.00	80.00	97.00
exchange	75.41	76.31	75.41	76.15	92.00
future	87.67	87.98	87.67	88.12	98.00
hour	89.58	89.58	89.58	89.58	92.00
job	82.05	82.44	82.05	82.05	90.00
management	75.44	75.56	75.22	75.44	98.00
network	56.36	57.09	56.36	57.27	98.00
order	91.23	91.23	91.23	91.23	95.00
part	-	-	68.94	69.44	97.00
people	23.74	25.39	23.65	25.57	96.00
point	33.03	34.67	33.00	34.67	92.00
policy	100.00	100.00	100.00	100.00	97.00
position	48.89	49.67	48.89	49.56	78.00
power	68.09	69.36	68.09	68.83	92.00
president	74.94	75.62	75.03	75.76	98.00
rate	87.59	88.03	87.59	87.97	92.00
source	41.14	43.43	41.71	42.86	86.00
space	78.57	78.57	78.57	78.57	100.00
state	80.56	80.69	80.56	80.97	86.00
system	54.21	55.21	53.5	54.86	79.00

disambiguators compact and fast to execute. To be sure, further improvements are to be expected from its combination with more conventional WSD techniques.

References

Agirre, E. and O. Lopez de Lacalle. 2004. Publicly available topic signatures for all WordNet nominal senses. In *Proceedings of the 4th International Conference on Languages Resources and Evaluations (LREC), Lisbon, Portugal*.

Azzini, A., M. Dragoni, and A.G.B. Tettamanzi. 2010. A novel similarity-based crossover for artificial neural network evolution. In *Proceedings of the XI International Conference on Parallel Problem Solving from Nature, PPSN'10 - Lecture Notes in Computer Science*, volume 6238, pages 344–353. Springer.

Azzini, A. and A.G.B. Tettamanzi. 2008a. Evolving neural networks for static single-position automated trading. *Journal of Artificial Evolution and Applications*, 2008(Article ID 184286):1–17.

Azzini, A. and A.G.B. Tettamanzi. 2008b. A new genetic approach for neural network design. In *Engineering Evolutionary Intelligent Systems. Studies in Computational Intelligence*, volume 82. Springer.

Bäck, T., F. Hoffmeister, and H.P. Schwefel. 1991. A survey of evolutionary strategies. In R. Belew and L. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 2–9, San Mateo, CA. Morgan Kaufmann.

Camargo, F.A. 1990. Learning algorithms in neural networks. Technical report, Computer Science Department, Columbia University.

Table 13
A summary of the results of applying the lexicographic encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words and by evaluating the system on the Semeval 2007 test set, with a comparison to the best Semeval 2007 system.

Word	Train on IXA		Train on SE2007 Benchmark		
	NEAT	NeuroGEN	NEAT	NeuroGEN	Benchmark
area	75.68	77.97	75.68	77.70	89.00
authority	76.19	77.38	76.19	77.14	86.00
base	75.00	76.00	75.00	75.75	80.00
bill	81.37	82.45	81.37	82.40	99.00
carrier	76.19	77.14	76.19	76.90	71.00
chance	46.67	49.00	46.67	49.67	73.00
condition	82.35	82.35	82.35	82.35	91.00
defense	42.86	48.81	42.86	47.14	57.00
development	72.41	73.97	72.41	74.14	100.00
drug	89.13	89.57	89.13	89.35	96.00
effect	83.33	83.67	83.33	83.67	97.00
exchange	80.33	80.82	80.33	81.07	92.00
future	89.73	90.79	89.73	90.51	98.00
hour	91.67	91.67	91.67	91.67	92.00
job	84.62	86.03	84.62	86.67	90.00
management	80.00	80.22	80.00	80.44	98.00
network	63.64	65.64	63.64	66.09	98.00
order	92.98	92.98	92.98	92.98	95.00
part	74.65	75.07	74.65	75.77	97.00
people	37.39	41.35	37.39	41.87	96.00
point	44.67	49.03	44.67	48.17	92.00
policy	100.00	100.00	100.00	100.00	97.00
position	57.78	60.44	57.78	59.56	78.00
power	74.47	75.43	74.47	75.21	92.00
president	79.66	80.88	79.66	80.71	98.00
rate	89.66	90.48	89.66	90.59	92.00
source	51.43	54.57	51.43	54.14	86.00
space	78.57	80.00	78.57	80.36	100.00
state	83.33	84.79	83.33	84.86	86.00
system	61.43	64.43	61.43	64.14	79.00

Chen, P., W. Ding, M. Choly, and C. Bowes. 2010. Word sense disambiguation with automatically acquired knowledge.

Cottrell, G.W. 1989. *A Connectionist Approach to Word Sense Disambiguation*. Pitman, London.

De Jong, K.A. 2002. *Evolutionary Computation: A unified approach*. MIT Press, Cambridge, MA.

Eiben, A.E. and J.E. Smith. 2003. *Introduction to Evolutionary Computing*. Springer, Berlin.

Escudero G., L. Màrquez, D. Martinèz and G. Rigau, 2006. *Supervised Corpus-Based Methods for WSD*, pages 167–207. Springer Netherlands.

Goldberg, D. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Kluwer Academic Publishers, Boston, MA.

Gurney, K. 1997. *An Introduction to Neural Networks*. Taylor & Francis, Inc., Bristol, PA, USA.

Harris, Z. 1954. Distributional hypothesis. *Word*, 10(23):146–162.

Hinton, G.E., J.L. McClelland, and D.E. Rumelhart. 1986. Distributed representations. In *Parellel Distributed Processing: explorations in the microstructure of cognition*. MIT Press, Cambridge, MA.

Kröse, B. and P. Van der Smagt. 1996. *An introduction to neural networks*. URL ftp://ftp.informatik.uni-freiburg.de/papers/neuro/ann_intro_smag.ps.gz, The University of Amsterdam.

Leacock, C., M. Chodorow, and G.A. Miller. 1998. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24(1):147–165.

Lesk, M. 1986. Automated sense disambiguation using machine-readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation, SIGDOC*, pages 24–26.

Marcus, M.P., B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

Table 14
A summary of the results of applying the POS-tagged lexicographic encoding scheme and the neuro-evolutionary approach to the disambiguation of the test words and by evaluating the system on the Semeval 2007 test set, with a comparison to the best Semeval 2007 system.

Word	Train on IXA		Train on SE2007 Benchmark		
	NEAT	NeuroGEN	NEAT	NeuroGEN	Benchmark
area	78.38	85.00	78.38	85.00	89.00
authority	77.86	84.05	77.62	85.00	86.00
base	76.75	84.00	76.50	84.00	80.00
bill	82.75	88.43	82.89	88.28	99.00
carrier	77.86	83.57	77.38	85.48	71.00
chance	53.33	65.00	53.33	67.67	73.00
condition	82.35	88.38	82.35	88.97	91.00
defense	49.05	66.90	49.29	66.43	57.00
development	75.86	81.72	75.86	83.28	100.00
drug	89.78	93.70	90.43	93.26	96.00
effect	83.83	89.00	84.33	89.67	97.00
exchange	81.48	87.21	81.31	86.72	92.00
future	90.86	94.11	90.89	93.39	98.00
hour	91.67	94.58	91.67	94.48	92.00
job	87.18	90.38	87.18	90.77	90.00
management	80.44	87.00	80.89	86.33	98.00
network	66.73	78.18	67.36	76.82	98.00
order	92.98	95.35	92.98	95.44	95.00
part	76.27	83.73	76.55	83.66	97.00
people	43.09	64.00	43.30	62.04	96.00
point	49.83	69.93	49.97	64.17	92.00
policy	100.00	100.00	100.00	100.00	97.00
position	61.11	72.33	61.33	74.00	78.00
power	76.17	83.19	75.74	84.15	92.00
president	81.50	88.14	81.44	88.14	98.00
rate	90.72	93.55	90.90	93.93	92.00
source	55.71	70.14	56.00	68.71	86.00
space	82.14	86.79	80.71	86.43	100.00
state	85.00	89.31	85.14	90.00	86.00
system	65.64	77.00	65.64	76.36	79.00

Martínez, D., O. Lopez de Lacalle, and E. Agirre. 2008. On the use of automatically acquired examples for all-nouns word sense disambiguation. *Journal of Artificial Intelligence Research*, 33:79–107.

McCarthy, D., R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 280–287.

Muhlenbein, H. and D. Schlierkamp-Voosen. 1993. The science of breeding and its application to the breeder genetic algorithm (bga). *Evolutionary Computation*, 1(4):335–360.

Navigli, R. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

Navigli, R. and S.P. Ponzetto. 2010. Knowledge-rich word sense disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1522–1531. Association for Computational Linguistics.

Pradhan, S.S., E. Loper, D. Dligach, and M. Palmer. 2007. Semeval-2007 task 17: English lexical sample, srl and all words. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval'2007*, pages 87–92, Prague. Association for Computational Linguistics.

Schütze, H. 1993. Word space. In *Proceedings of the 1993 Conference on Advances in Neural Information Processing Systems, NIPS '93*, pages 895–902, San Francisco, CA. Morgan Kaufmann.

Schwefel, H.P. 1981. *Numerical Optimization for Computer Models*. John Wiley, Chichester, UK.

Stanley, K.O. and R. Miikkulainen. 2002. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127.

Toutanova, K., D. Klein, C.D. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *HLT-NAACL*.

Veronis, J. and N.M. Ide. 1990. Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *Proceedings of the 13th conference on Computational linguistics - Volume 2*, COLING '90, pages 389–394. Association for Computational Linguistics.

Waltz, D.L. and J.B. Pollack. 1985. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, 9:51–74.

Yao, X. 1999. Evolving artificial neural networks. In *Proceedings of the IEEE*, pages 1423–1447.

Yarowsky, D. and R. Florian. 2002. Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering*, 8(4):293–310.