Cognitive Computation
https://doi.org/10.1007/s12559-020-09758-w

A Reservoir Computing Approach to Word Sense Disambiguation

Kiril Simov[1] · Petia Koprinkova-Hristova[1] · Alexander Popov[1] · Petya Osenova[1]

Received: 11 February 2020 / Accepted: 23 July 2020

Abstract
Reservoir computing (RC) has emerged as an alternative approach for the development of fast trainable recurrent neural networks (RNNs). It is considered to be biologically plausible due to the similarity between randomly designed artificial reservoir structures and cortical structures in the brain. The paper continues our previous research on the application of a member of the family of RC approaches—the echo state network (ESN)—to the natural language processing (NLP) task of Word Sense Disambiguation (WSD). A novel deep bi-directional ESN (DBiESN) structure is proposed, as well as a novel approach for exploiting reservoirs' steady states. The models also make use of ESN-enhanced word embeddings. The paper demonstrates that our DBiESN approach offers a good alternative to previously tested BiESN models in the context of the word sense disambiguation task having smaller number of trainable parameters. Although our DBiESN-based model achieves similar accuracy to other popular RNN architectures, we could not outperform the state of the art. However, due to the smaller number of trainable parameters in the reservoir models, in contrast to fully trainable RNNs, it is to be expected that they would have better generalization properties as well as higher potential to increase their accuracy, which should justify further exploration of such architectures.

Keywords Reservoir computing · Echo state network · Word sense disambiguation · Word embeddings

Introduction

In the area of natural language processing (NLP), RNNs are considered a viable tool for linguistic modeling, due to their ability to keep memory traces of the context (preceding and/or succeeding text) of a given word at theoretically arbitrary distances from it. That is why BiLSTMs have been successfully applied to a number of sequence-to-sequence tasks in NLP, such as part-of-speech tagging, chunking, named entity recognition, dependency parsing [1–4]).

Word sense disambiguation (WSD) is an NLP task aimed at assigning proper categories of meaning to words that are ambiguous (i.e., they can assume several related or unrelated meanings, depending on the context). For instance, the word "chair" can refer to a person who is managing some activity ("The chair gave the word to the next participant in the meeting."), or to a piece of furniture ("The student sat down on his chair."). In order to do WSD, we need to account for all words (the context) not only preceding but also following the target word ("chair" in our example). Various LSTM-based architectures have been explored successfully with regard to the task of WSD [5].

In spite of the undoubted power of currently developed RNN architectures such as Long Short-Term Memory (LSTM) cells [6] and Gated Recurrent Units (GRUs) [7], their training via the gradient descent algorithm remains a computationally demanding task, especially in the case of very deep network structures.

Aimed at the development of fast trainable RNNs, an alternative approach was independently proposed in 2002

This article belongs to the Topical Collection: *Trends in Reservoir Computing*
Guest Editors: Claudio Gallicchio, Alessio Micheli, Simone Scardapane, Miguel C. Soriano

✉ Petia Koprinkova-Hristova
pkoprinkova@yahoo.com; pkoprinkova@bas.bg

Kiril Simov
kivs@bultreebank.org

Alexander Popov
alex.popov@bultreebank.org

Petya Osenova
petya@bultreebank.org

[1] Institute of Information and Communication Technologies, Bulgarian Academy of Sciences, Sofia, Bulgaria

Published online: 07 September 2020

by [8] under the name Liquid State Machines (LSMs) and by [9] under the name Echo State Networks (ESNs). Currently these are collectively referred to as reservoir computing (RC) [10, 11] approaches. Furthermore, reservoir computing was considered to be a more biologically plausible approach—there are parallels between reservoirs and cortical structures in the brain [12, 13].

The main insight of this line of research lies in generating a random and sparsely connected recurrent reservoir of neurons whose mutual connection weights are not subject to training, and a linear readout layer that can be tuned in one shot (presenting each training sample only once and solving the least squares problem). Thus the main advantages of RC and particularly ESNs over fully trainable RNNs are:

- Smaller number of trainable parameters
- Single training epoch

Since their emergence, RC approaches have been widely used for the modeling of a variety of dynamical systems [14]. Recently Gallicchio et al. [15] have compared deep ESN architectures with popular deep gated RNNs (like LSTM and GRU architectures) for time series prediction, and have shown them to be significantly more efficient than others RNN approaches, and the best solution in terms of prediction accuracy on 3 out of 4 benchmark tasks considered in that work.

Applications of ESNs in NLP have started to appear only recently and for that reason there are only a few works in this area. The possibility of language modeling via ESNs is investigated in [16, 17]. In [18, 19], ESNs are applied to semantic role labeling in a multimodal robotic architecture. Other NLP applications are in the area of speech processing—[20, 21], language modeling—[22], and named entity recognition [23]. To the best of our knowledge, there are no other examples of ESNs being used for WSD.

Our preliminary attempt on the WSD task, using a single ESN reservoir [24], has shown that although the training and testing errors on predicted sense embedding vectors are quite small, the accuracy achieved on sense prediction is actually quite low. That is why in [25] we have adopted the bi-directional approach from [26, 27]—but with linear or ReLU+softmax reservoir readout layers. Comparison with results reported in [28] demonstrated that BiESN models having ReLU+softmax reservoir readout layers were unable to outperform accuracy achieved by similar BiLSTM model architectures while in the case of linear readout BiESNs achieved higher WSD accuracy in comparison with BiLSTM models with an analogous architecture. Besides, BiESN models had a much smaller number of trainable parameters in comparison with BiLSTMs. Another advantage of bi-directional reservoir architectures is that training via the recursive least squares (RLS) algorithm needs one epoch, in contrast with iterative backpropagation training of BiLSTMs that need at least several epochs.

Provoked by the observation in [24] that in spite of the high predictive accuracy at identifying similar word embeddings at the ESN output, the final word sense disambiguation is not as successful due to closeness among the sense embedding vectors, in [29] we investigate the influence of different word embedding models on WSD accuracy and, based on ideas for secondary feature extraction from [30], we propose an approach for the improvement of pre-trained word embeddings which aims at increasing the distance between individual embedding vectors. Nevertheless, the highest results on the WSD task achieved with BiESN models remained slightly below the state of the art.

In the present paper, in an attempt to increase the BiESN model accuracy, we advance the BiESN architecture with linear readout from [25] to a hierarchical (deep) ESN structure called DBiESN and evaluate it on the same WSD setup. The basic idea to exploit reservoir steady state from [30] rather than its current state was also adopted. The improved word embedding vectors from [29] were used to train our DBiESN model. The achieved WSD accuracy reported here is slightly above that of the accuracy reported in our previous work [25]—regarding BiESN with linear readout. However, the number of trainable parameters of the DBiESN model which achieves similar WSD accuracy was smaller in comparison with our best BiESN model from [25].

The structure of the paper is as follows: the next section introduces ESN basics and briefly describes our previous research on BiLSTMs and BiESNs for WSD, as well as the process of improving word embeddings via processing through ESNs; then, "Deep BiESN Model" introduces our DBiESN model; the next section presents results from the application of our DBiESN model to the WSD task; the last section is dedicated to conclusions and future work.

Basics and Initial Results

In this section we briefly introduce ESN basics as well as some results from our previous work [25] and [29].

The ESN reservoir structure presented on Fig. 1 is described by the following equations:

$$\begin{aligned} r(k) &= (1-a)r(k-1) + a\tanh(W^{\text{in}} in(k) \\ &\quad + W^{\text{res}} r(k-1)) \qquad (1) \\ \text{ESN}_{\text{out}}(k) &= W_r^{\text{out}}[r(k), in(k)] \qquad (2) \end{aligned}$$

Here $in(k)$ is the input vector for the current time instant k; vector r represents the internal state of the ESN reservoir; a is a constant called *the leaking rate* (of ESN); $\text{ESN}_{\text{out}}(k)$

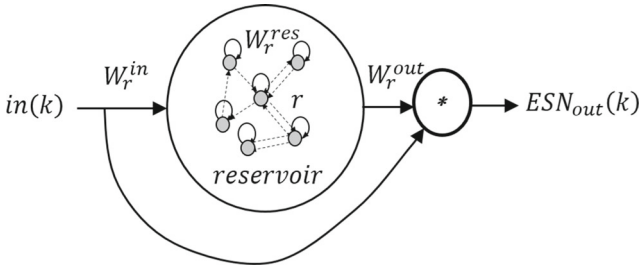

Fig. 1 ESN reservoir structure. Dashed lines represent recurrent connections, while solid lines—feedforward connections

denotes the output vector; W^{in} and W^{res} are input and recurrent connection weight matrices; the output weight matrix is denoted by W_r^{out}.

A comparison in [25] demonstrated that the number of trainable parameters of BiLSTM and BiESN models having the bi-directional structure adopted from [31] (two RNN cells—LSTM or ESN—accumulating the left and right contexts of the kth word) with the same number of internal units ($n_{\text{LSTM}} = n_r = n_{\text{units}}$), as well as the same input and output vectors' sizes (n_{in} and n_{out}, respectively), is:

$$n_{\text{tr.params}}^{\text{BiLSTM}} = 8n_{\text{units}}(n_{\text{in}} + n_{\text{units}} + 1) + n_{\text{out}}(n_{\text{in}} + 2n_{\text{units}}) \quad (3)$$

$$n_{\text{tr.params}}^{\text{BiESN}} = n_{\text{out}}(n_{\text{in}} + 2n_{\text{units}}) \quad (4)$$

Detailed information about both BiRNN architectures can be found in our previous work [25]. Here we draw attention to some of the results reported there (see Table 1) concerning pure BiRNN architectures (without additional readout layers such as softmax and ReLU, which have been investigated in [25]), since this is the starting point of our current work.

The results from the *IMS-s+emb* system are used as representative of the state of the art, as reported in [32] where the model is a trained SVM using tailor-made features, which include word embeddings. The most frequent sense (MFS) baseline, which is a very strong one and is also described in [32], is provided as well. Although the reported results are below the state of the art (*IMS-s+emb*), it is clear that the BiESN architecture significantly outperforms BiLSTMs in both aspects: with respect to the number of trainable parameters, and the achieved WSD accuracy.

Table 1 Best WSD accuracy scores for both BiRNN models from [25]

BiRNN model	n_{units}	Accuracy	$n_{\text{tr.params}}^{\text{BiRNN}}$
LSTM	100	62.6	470,800
ESN	100	61.712	150,000
LSTM	1000	61.9	11,098,000
ESN	1000	63.201	690,000
ESN	5000	65.049	3,090,000
MFS		64.8	
IMS-s+emb		69.6	

In our preliminary investigations [24], we observed that although the achieved root mean squares error (RMSE) on the training and testing data is quite small, the WSD accuracy obtained via cosine similarity was actually very poor. A reasonable explanation is that although the RMSE between predicted and target embedding vector elements is small, the distance between these vectors is much bigger than what is desirable, because the word sense embedding vectors are too close to one another. One potential option for remedying this situation is to use word embedding vectors generated in some alternative fashion, hoping that they would be clearly distinguishable. In [29] we have investigated two different embedding models—`EmbGraph` and `EmbGraphWiki`—for which embedding vector size is 300 elements and our observations were that although in all cases increasing the reservoir size lead to increased WSD accuracy, the highest accuracy scores for the two embedding models were almost the same (around 65%) and were achieved with BiESNs having two reservoirs of 5000 units each.

Using ESNs for embeddings improvement was inspired by the application of ESNs to the feature extraction task introduced in [30]. The idea behind this approach is to map original feature vectors to another space which has a different or identical dimensionality, thereby changing the relative positions of feature vectors and making them clearly distinguishable from one another. The newly extracted features are represented by the steady states of reservoir neurons—corresponding to the original input feature vectors. This approach has been successfully applied to various data clustering tasks [33]. In [29] we have used the same technique for further improving embedding vectors in order to achieve better discrimination between them. The "improved" embeddings were calculated as steady states of an ESN reservoir after repeating the presentation of each one of the original embeddings. The highest WSD accuracy (65.916%) was recorded with new input embeddings of size 600, new output embeddings of size 900 and a BiESN consisting of two reservoirs of 5000 neurons each thus having $n_{\text{tr.params}}^{\text{BiESN}} = 9{,}540{,}000$ trainable parameters that is almost three times more than reported in Table 1 number for the biggest BiESN model due to increased embedding sizes for both input and output of the model.

The conclusions from the evaluations in [25] and [29] were as follows:

– Bigger reservoir sizes help for improving the WSD accuracy in all cases;
– The new embeddings generated via iteration until settling into a steady state of a single ESN reservoir improved WSD accuracy for identical reservoir sizes.

Deep BiESN Model

The preliminary results described in the previous section, together with those reported in the recent literature on deep ESN architectures [34–38], have motivated the present work, which has two aims:

– To make the BiESN model deeper
– To exploit the ESN reservoir equilibrium states

The first aim is naturally based on the idea that deep structures are known to be able to extract hidden features from data. In the case of the WSD task we will consider deep layers as additional memory units accumulating word context information.

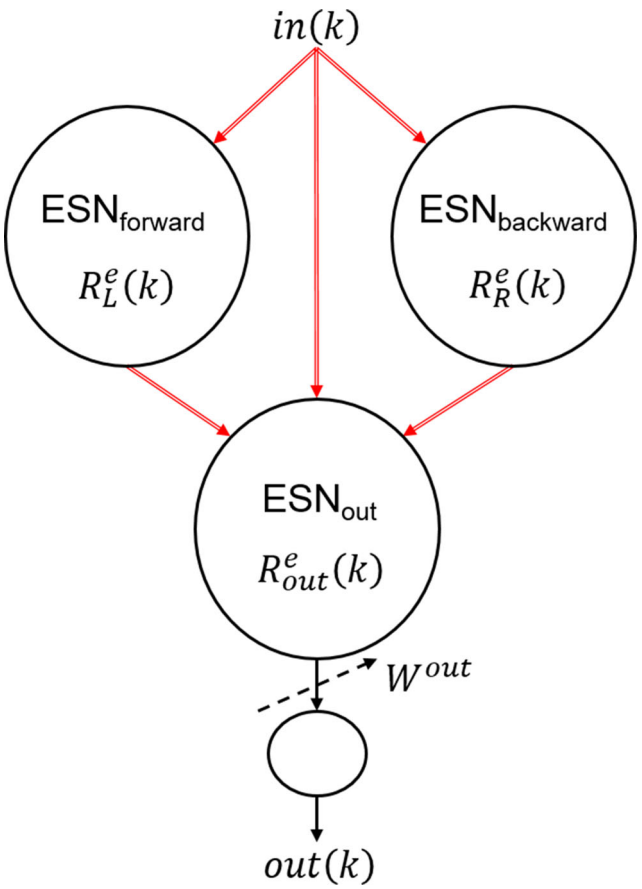

Fig. 2 Deep BiESN structure. Red double lines represent vectors which are fed multiple times until the reservoirs settle down into a new steady state

The second aim is based on the idea for feature extraction proposed first in [30], that is: mapping the original multidimensional feature space to another (ESN reservoir steady state for each original feature), which has a different dimensionality, could result in easier separability of data. Thus, we exploit the steady states of each ESN reservoir in our deep model, instead of its current state.

The proposed deep BiESN (DBiESN) structure is presented in Fig. 2. Our deep BiESN model has two hidden layers, which is typically considered a deep structure in the literature [39]. Both forward ($ESN_{forward}$) and backward ($ESN_{backward}$) reservoirs in the first hidden layer are considered as RNNs accumulating the left and right context, respectively. They are initialized from a zero state ($R_L(0) = 0$ and $R_R(0) = 0$) and then are "wormed" (led to another initial state, different from zero) by consecutive feeding of the first ($First\_word$) and last ($Last\_word$) word embedding from each of the training texts nt, thus achieving new initial state according to (5) (see Fig. 3).

$$R_{L/R}^{\text{ini}} = R_{L/R}^e(First\_word(nt)/Last\_word(nt))|_{nt=1}^{NT} \quad (5)$$

After worming (initialization), both reservoirs are fed with the target word embedding $in(k)$ as many times as necessary in order to achieve the new steady states $R_L^e(k)$ and $R_R^e(k)$, respectively (see Fig. 2). In both Figs. 2 and 3, red double arrows denote inputs that have been fed multiple times into the reservoirs until they settle into a new steady

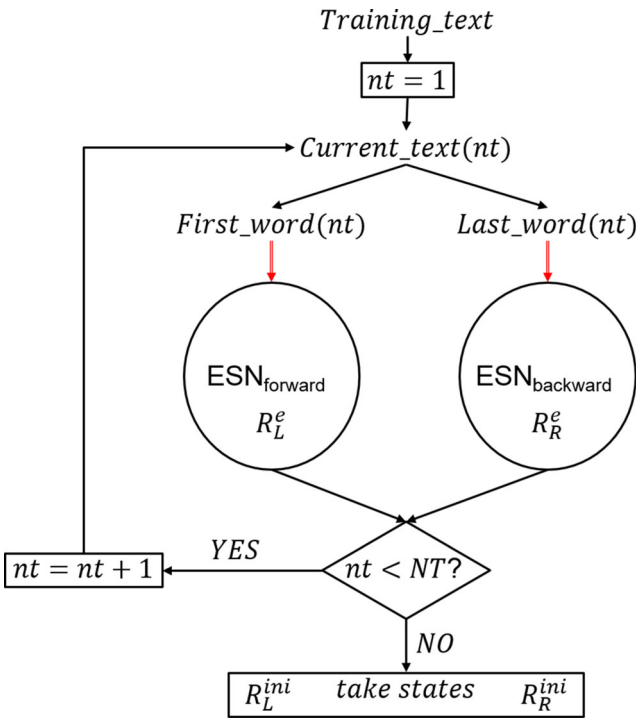

Fig. 3 Worming of the first layer of our deep ESN model. Red double lines represent vectors which are fed NT times into both reservoirs in order to achieve new nonzero initial state

state. The second hidden layer of the deep ESN model consists of another reservoir (ESN_{out}) whose input is a concatenation of the previous layers' output and of the target word embedding $[R^e_L(k), R^e_R(k), in(k)]$. The initial state of that reservoir is set to zero and it is fed with that input vector as many times as needed to achieve its steady state $R^e_{\text{out}}(k)$. The final model output is a linear combination of the steady states $R^e_{\text{out}}(k)$ of the second reservoir layer. The overall model equations are as follows:

$$R_{L/R}(0) = R^{\text{ini}}_{L/R} \quad (6)$$

$$R^e_{L/R}(k) = \text{ESN}^{\text{steady state}}_{L/R}(in(k)) \quad (7)$$

$$R_{\text{out}}(k) = 0 \quad (8)$$

$$R^e_{\text{out}}(k) = \text{ESN}_{\text{out}}{}^{\text{steady state}}([R^{e_k}_L(k), R^{f_k}_R(k), in(k)]) \quad (9)$$

$$out(k) = W^{\text{out}} R^e_{out}(k) \quad (10)$$

where the only trainable parameters are the weights W^{out} connecting the second layer (ESN_{out}) to the output. Thus, the overall number of DBiESN trainable parameters became:

$$n^{\text{DBiESN}}_{tr.params} = n_{\text{ESN}_{\text{out}}} n_{\text{out}} \quad (11)$$

Here $n_{\text{ESN}_{\text{out}}}$ is the number of units in the second (output ESN reservoir). Thus, the proposed DBiESN architecture has decreased its number of trainable parameters in comparison with the those of BiESN model with the same number of units (see (4)).

Training is done via RLS by comparing against the embedding vector for the gold sense.

The steady states of all three reservoirs ($R^e_{L,R,\text{out}}$) are achieved through the consecutive presentation of the constant input vector, until the corresponding reservoir states converge. The convergence criteria in all cases are:

$$\sum \sqrt{(R(it) - R(it-1))^2}/n_R < \theta \quad (12)$$

The threshold parameter θ is set to $1.0E07$. According to (12), each reservoir needs a different number it of consecutive iterations over constant input in order to settle into a new steady state.

Results and Discussion

Model Training and Accuracy

The sense inventory used for all experiments we report in the paper is the WordNet lexical database—[40]. This choice of a lexical database restricts the WSD task that we solve to word senses for nouns, verbs, adjectives and adverbs (i.e., excluding function, or closed class, words such as prepositions, conjunctions, determiners, etc.). Therefore, in the context of the WSD task, words in the input text that are subject to disambiguation are drawn only from those tokens in the input text that can be assigned sense tags from the WordNet lexical resource.

As in our previously reported research, the texts are from the SemCor corpus—[41] provides our training data set. SemCor consists of 352 texts with about 2000 words each. The open class words (nouns, verbs, adjectives, adverbs) in the text have been manually annotated with senses from the Princeton WordNet lexical database—[40]. A total of 166 documents contain only annotations of verbs. After some initial experimentation, we have decided to use the fully annotated 186 documents, which contain 185∼269 words annotated with senses—our training data samples. Each of the texts has been analyzed in its entirety—this provides the contextual input used by the ESN network to disambiguate the sense-bearing words in the same text.

The unified evaluation framework proposed by [32] is based on several evaluation datasets from different evaluation campaigns: Senseval-2 [42], Senseval-3 task 1 [43], SemEval-07 task 17 [44], SemEval-13 task 12 [45], and SemEval-15 task 13 [46]. In our experiments, we use the union of all these datasets for evaluation. This joint dataset contains 7∼253 annotated words—which is the number of tags in our testing data samples.

Following our preliminary results from [29], we have used ESN-improved embedding vectors of size 600 for the input and of size 900 for the output of the deep BiESN model, respectively. Both context reservoirs of the first (BiESN) layer have 100 neurons, while for the second (output) reservoir we have tested varying sizes. All reservoirs have leakage rates set to 0.5 and 50% sparsity, like in previous investigations.

First we have performed an empirical investigation on the number of iterations (it) per input needed to achieve steady states for all three reservoirs in our deep model structure, for about 10.000 target words. For this aim, we set the output reservoir size to 4000 neurons.

It turned out that while for the forward and backward ESNs from the first layer settling time has an almost Gaussian distribution, varying from 10 up to 25 iterations, with a mean number of iterations around 15–16 (see Figs. 4 and 5), the output reservoir (Fig. 6) in most cases needs 13 to 14 iterations to settle into its equilibrium state.

Our explanation for that fact is that for each target word the starting state (before iterative feeding of target word embedding) of the context reservoirs is initialized with the corresponding context sequences, which strongly depends on the given word and its position in the text. Thus, the first layer may need a varying number of iterations in order to settle into a steady state, depending on the initialization context.

In the case of the output reservoir, for each word the initialization of the reservoir is a zero value vector, which means that the starting point for processing each word is

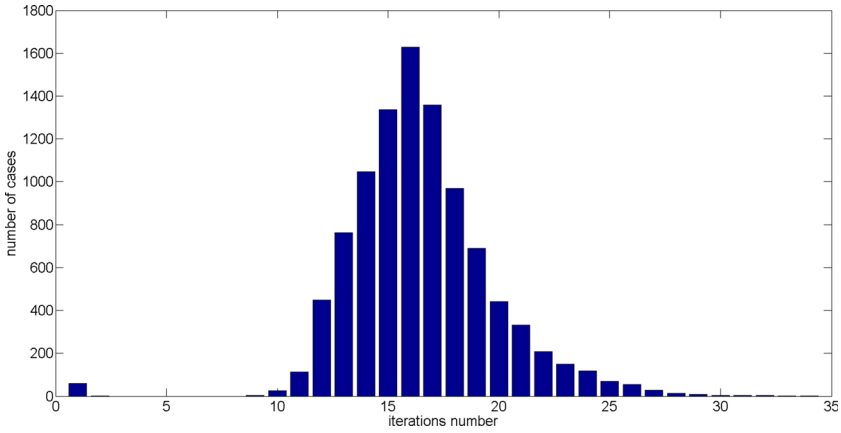

Fig. 4 Forward reservoir settling iterations

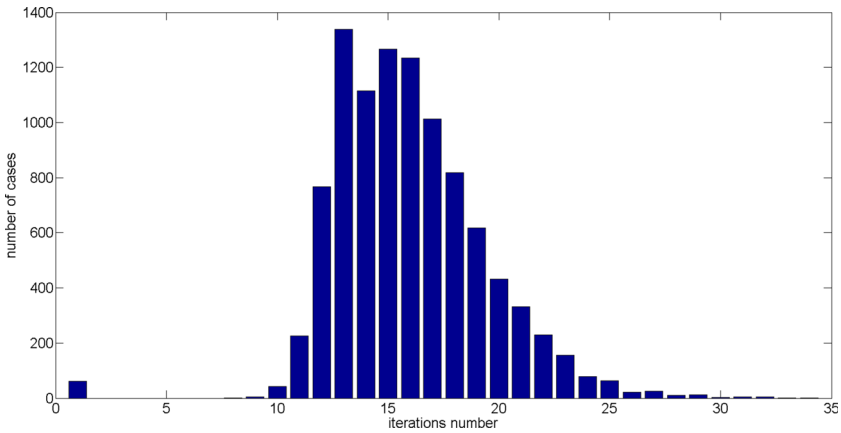

Fig. 5 Backward reservoir settling iterations

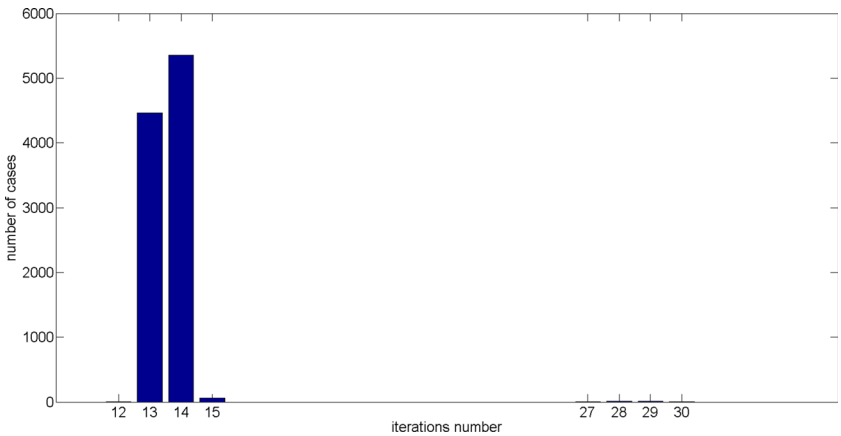

Fig. 6 Output reservoir settling iterations

Table 2 WSD accuracy with different sizes of the output reservoir, for spectral radius = 1.17

Reservoir size	3400	3500	3600	3700	3800	4000	4050	4500
Min. accuracy	64.75	64.73	64.69	64.92	64.93	64.99	65.02	64.80
Max. accuracy	65.44	65.40	65.88	65.55	65.64	65.88	65.84	65.87
Average accuracy	65.10	65.08	65.24	65.35	65.21	65.42	65.38	65.27

the same. This explains why ESN_{out} needs approximately the same number of iterations for almost all words to settle into its new steady state. We have also tried starting from a previous steady state (the result of the processing of a previous target word), but in such cases the model performs quite poorly.

Next we have performed several series of experiments in order to tune the reservoirs' spectral radius, as well as the size of the output reservoir.

Table 2 presents results on WSD accuracy with regards to the reservoir size of the second layer. In this experiment we set the spectral radius of the reservoir matrix to 1.17. Based on these results, the optimal output reservoir size appears to be 4000.

These experiments have shown that DBiESN architecture achieved almost the same accuracy like results reported in our previous works (see Table 1), with much smaller numbers of neurons in all three reservoirs together as well as less trainable parameters.

Table 3 summarizes minimal, maximal and average accuracy scores achieved out of 10 runs, starting with different initial model parameters and varying spectral radius. It shows that an increase of the spectral radius to a value slightly above 1.0 leads to increased model accuracy, but with values beyond 1.1 accuracy drops down. Hence we chose a spectral radius of 1.1 as optimal.

Further investigations aimed at the refinement of the results by varying the spectral radius between 1.1 and 1.17, with output reservoir sizes of 4000 and 4500, have led us to the following conclusions concerning optimal hyperparameter values: spectral radius = 1.1; $n_{R_{\text{out}}}$ = 4000; reservoir sparsity = 50%; θ = 1.0E−7. These parameters yield the following WSD accuracy scores: Min. accuracy = 65.03%; Max. accuracy = 65.72%; average accuracy = 65.34%.

These experiments show that our deep ESN architecture achieves accuracy similar to our previous results with a total number of 4200 neurons in all three reservoirs (100 + 100 + 4000) and total number of trainable parameters $n_{\text{tr.params}}^{\text{DBiESN}}$ = 3,600,000. The highest result for the BiESN model reported in our previous work was achieved with a total reservoir size of 10,000 (2×5000) neurons—more than twice the size of the best model reported here; it also has almost three times more trainable parameters (9,540,000).

Another advantage of the proposed approach is that training the model is much faster than that of fully recurrent architectures like LSTMs. As it was reported [47, 48], the stochastic gradient descend (SGD) algorithm applied in fully trainable RNNs, although less computationally demanding than classical gradient descend algorithms, is slower and much more unstable in comparison with the recursive least squares (RLS) we apply in reservoir model training. Training the best DBiESN model reported here—having 4200 units—took about 15 h. Having in mind that we are using our own Python implementation, which has not yet been significantly optimized at this stage, we could expect even faster training time in principle. In comparison, a BiLSTM with 100 units reported in our previous work [25] and written in the highly optimized Tensorflow software environment needed about 20 h of training via stochastic gradient descent. Training larger BiLSTM models (e.g., 1000 units per LSTM) takes close to a week.

Our investigations also provide evidence that the internal states of reservoirs could be used in different ways at different points of the deep architecture. The exploitation of their steady states instead of the current ones happens to be a powerful approach for the redistribution of the features encoded in the input vectors. In our future work we plan to implement this approach in conjunction with other RNN structures—like LSTM cells trained via backpropagation.

Table 3 WSD accuracy with different spectral radia of the reservoir connection matrix, for an output reservoir of size 4000

Spectral radius	0.5	0.85	0.92	1.0	1.1	1.25	1.5
Min. accuracy	64.91	64.70	64.74	64.98	64.82	64.85	64.62
Max. accuracy	65.61	65.75	65.65	65.75	65.83	65.82	65.64
Average accuracy	65.21	65.23	65.22	65.33	65.36	65.24	65.19

Linguistic Error Analysis

In this section we examine the main error types in the resulting automatic analyses. To this purpose, the predicted values given by the best-performing model are compared against the available reference data. The erroneous cases have been extracted and ordered for analysis in decreasing order of similarity between the predicted tag and the gold tag.

We have deliberately focused on cases where the similarity is high, i.e., the differences between the senses seem to be negligible. Two main error types can be distinguished in this respect. In the first type, the predicted meaning is very similar but not identical to the reference sense. This situation occurs mostly for lemmas with multiple meanings. This is understandable because the more fine-grained the senses, the more context-dependent they are. For example, instead of the gold meaning of 'pull' (*apply force so as to cause motion towards the source of the motion*), the system selects a very close one (*cause to move by pulling*).

Deviation might be caused by other factors as well, e.g., when the model cannot distinguish between the inchoative and causative meanings. Consider, for instance, the case where the gold meaning for "attach" is the inchoative—"be attached; be in contact with," while the one selected by the system is the causative—"cause to be attached."

For these reasons, the system tends to select the word sense that is listed first. This means that the available knowledge might be insufficient for making a correct selection.

In the second case type, errors are due to the usage of the lemma in more than one domain. Here, again, context plays a crucial role. For example, the lemma 'cell' is a term in biology with the following meaning: "(biology) the basic structural and functional unit of all organisms," which is given as the gold sense. However, the system chooses the more general meaning of "any small compartment." Another example is the legal term "trial"—"(law) the determination of a person's innocence or guilt by due process of law," given as the gold sense. The system again prefers the more general meaning—"the act of testing something." In this group, we put also the regular polysemy cases where a lemma refers to different but related entities. For instance, the lemma "glass" might refer to the container property (a container for holding liquids while drinking) or to the material (a brittle transparent solid with irregular atomic structure).

Additionally, the lexical resource holds overlapping senses that confuse the supervised system due to their sparse availability in the data. For example, "shock" is annotated with the gold sense of "an unpleasant or disappointing surprise," while the automatically selected sense says roughly the same but in other words—"the feeling of distress and disbelief that you have when something bad happens accidentally." One well-known solution to this problem would be to merge the similar as well as the fine-grained word senses into more general classes.

Conclusions

The research reported here, as well our previous comparative investigations on BiESN vs. BiLSTM architectures, demonstrates that reservoir approaches are a good alternative to the commonly accepted BiLSTM models, as used in the context of the word sense disambiguation task. ESN-based models do not reach the peak performance levels (in terms of accuracy) of LSTM-based ones when compared in a pure classification scenario (having as readout layers ReLU and softmax classifiers), but they come reasonably close to them. When used as a mechanism for finding maximally close contextual embeddings to the gold senses, ESN-based models significantly outperform LSTM-based ones, which indicates that they might offer a better way of analyzing meaning according to this particular approach. At the same time, ESN networks are shown as much lighter and easier to train—in terms of memory, and processing requirements. The paper further shows that more sophisticated DBiESN models can be designed to reduce the computational requirements even further. This makes ESNs a very attractive alternative to LSTMs, which remain difficult and expensive to train. Since the number of trainable parameters in reservoir models is much smaller than that in fully trainable RNNs, these approaches have the potential to achieve better generalization results anyway.

Having in mind the recently discovered "deep double descent" effect in [49], we could expect that further investigations would reveal the units number threshold above which the reservoirs would improve their accuracy even more.

Being a new and fast developing trend in RNN modeling, ESN modeling presents a good alternative to widely established RNN models, which has potential to outperform the latter in the near future.

The presented research was inspired by the work on neural network approaches in NLP and the particular application is focused on word sense disambiguation. In the future it could be applied to various dynamic data classification tasks. Since bi-directional architectures are suitable for time series for which both forward and preceding context are important, language processing is probably the most representative area with regards to such tasks. Our intention is to test the proposed approach on various other data sets in the future.

Funding This research has been partially supported by the National Scientific Program "Information and Communication Technologies for a Single Digital Market in Science, Education and Security (ICTinSES)," financed by the Ministry of Education and Science. Alexander Popov's contribution has been supported by the Bulgarian Ministry of Education and Science under the National Research Programme "Young scientists and postdoctoral students" approved by DCM # 577 / 17.08.2018.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Wang P, Qian Y, Soong FK, He L, Zhao H. Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network. 2015. arXiv:1510.06168.
2. Wang P, Qian Y, Soong FK, He L, Zhao H. A Unified Tagging Solution: Bidirectional LSTM Recurrent Neural Network with Word Embedding. 2015. arXiv:1511.00215.
3. Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF Models for Sequence Tagging. 2015. arXiv:1508.01991.
4. Wang W, Chang B. Graph-based Dependency Parsing with Bidirectional LSTM. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers Association for Computational Linguistics, Berlin Germany; 2016. p. 2306–2315. https://doi.org/10.18653/v1/P16-1218.
5. Popov A. Neural network models for word sense disambiguation: an overview. Cybernetics and Information Technologies. 2018;18:139–151.
6. Hochreiter S, Schmidhuber J. Long short-Term memory. Neural Comput. 1997;9(8):1735. https://doi.org/10.1162/neco.1997.9.8.1735.
7. Cho K, van Merriënboer B., Bahdanau D, Bengio Y. On the Properties of Neural Machine translation: encoder–Decoder approaches. In: proceedings of SSST-8, Eighth Workshop on Syntax Semantics and Structure in Statistical Translation Association for Computational Linguistics Doha Qatar; 2014. p. 103–111. https://doi.org/10.3115/v1/W14-4012.
8. Maass W, Natschläger T, Markram H. Real-time Computing Without Stable states: A New Framework for Neural Computation Based on Perturbations. Neural Comput. 2002;14(11):2531. https://doi.org/10.1162/089976602760407955.
9. Jaeger H. Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the Echo State Network Approach, GMD Report 159 German National Research Center for Information Technology. 2002.
10. Lukosevicius M, Jaeger H. Reservoir computing approaches to recurrent neural network training. Computer Science Review. 2009;3:127. https://doi.org/10.1016/j.cosrev.2009.03.005.
11. Scardapone S, Butcher J, Bianchi S, Malik Z. Advances in Biologically Inspired Reservoir Computing. 2017, Vol. 9. https://doi.org/10.1007/s12559-017-9469-1.
12. Enel P, Procyk E, Quilodran R, Dominey PF. Reservoir Computing Properties of Neural Dynamics in Prefrontal Cortex. PLOS Computational Biology. 2016;12(6):1. https://doi.org/10.1371/journal.pcbi.1004967.
13. Heinrich S, Wermter S. Interactive natural language acquisition in a multi-modal recurrent neural architecture. Connection Science. 2018;30(1):99. https://doi.org/10.1080/09540091.2017.1318357.
14. Butcher JB, Verstraeten D, Schrauwen B, Day CR, Haycock PW. Reservoir Computing and Extreme Learning Machines for Non-linear time-Series Data Analysis. Neural Netw. 2013;38:76. https://doi.org/10.1016/j.neunet.2012.11.011.
15. Gallicchio C, Micheli A, Pedrelli L. Comparison between Deep ESNs and Gated RNNs on Multivariate Time-Series Prediction. 2018. arXiv:1812.11527.
16. Frank SL, Čerňanský M. P. Generalization and Systematicity in Echo State Networks, in the Annual Meeting of the Cognitive Science Society; 2008. pp. 733–738.
17. Hinaut X, Dominey PF. Real-time Parallel Processing of Grammatical Structure in the fronto-Striatal system: A Recurrent Network Simulation Study Using Reservoir Computing. PLOS ONE. 2013;8(2):1. https://doi.org/10.1371/journal.pone.0052946.
18. Twiefel J, Hinaut X, Wermter S. Semantic Role Labelling for Robot Instructions using Echo State Networks. In: 24th European Symposium on Artificial Neural Networks, ESANN 2016, Bruges, Belgium, April 27-29, 2016; 2016. http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2016-168.pdf.
19. Twiefel J, Hinaut X, Soares MB, Strahl E, Wermter S. Using Natural Language Feedback in a Neuro-Inspired Integrated Multimodal Robotic Architecture. In: 25th IEEE International Symposium on Robot and Human Interactive Communication, RO-MAN New york, NY, USA, August 26-31, 2016; 2016. p. 52–57. https://doi.org/10.1109/ROMAN.2016.7745090.
20. Skowronski M, Harris J. Minimum mean squared error time series classification using an echo state network prediction model in 2006 IEEE international symposium on circuits and systems IEEE. 2006. https://doi.org/10.1109/ISCAS.2006.1693294.
21. Squartini S, Cecchi S, Rossini M, Piazza F. Echo State Networks for Real-Time Audio Applications. In: Advances in Neural Networks – ISNN 2007, ed. by D. Liu, S. Fei, Z. Hou, H. Zhang, C. Sun Springer Berlin Heidelberg, Berlin, Heidelberg; 2007. p. 731–740. https://doi.org/10.1007/978-3-540-72395-0_90.
22. Tong MH, Bickett AD, Christiansen EM, Cottrell GW. Learning grammatical structure with echo state networks. Neural Netw. 2007;20(3):424. https://doi.org/10.1016/j.neunet.2007.04.013.
23. Ramamurthy R, Stenzel R, Sifa R, Ladi A, Bauckhage C. Echo State Networks for Named Entity Recognition. In: Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions ed. by I.V. Tetko, V. Kůrková, P. Karpov, F. Theis Springer International Publishing, Cham; 2019. p. 110–120.
24. Koprinkova-Hristova P, Popov A, Simov K, Osenova P. Echo State Network for Word Sense Disambiguation. In: Artificial intelligence: Methodology, Systems, and Applications - 18th International Conference, AIMSA 2018, Varna, Bulgaria, September 12-14, 2018 Proceedings; 2018. p. 73–82. https://doi.org/10.1007/978-3-319-99344-7_7.
25. Popov A, Koprinkova-Hristova P, Simov K, Osenova P. Echo State vs. LSTM Networks for Word Sense Disambiguation. In: Artificial Neural Networks and Machine Learning – ICANN 2019: Workshop and Special Sessions ed. by I.V. Tetko, V. Kůrková, P. Karpov, F. Theis Springer International Publishing, Cham; 2019. p. 94–109.
26. Gallicchio C, Micheli A. A Reservoir Computing Approach for Human Gesture Recognition from Kinect Data. In: Proceedings of the Artificial Intelligence for Ambient Assisted Living 2016 co-located with 15th International Conference of the Italian Association for Artificial Intelligence AIxIA 2016 Genova, Italy, November 28th, 2016; 2016. p. 33–42. http://ceur-ws.org/Vol-1803/paper3.pdf.

27. Rodan A, Sheta AF, Faris H. Bidirectional Reservoir Networks Trained Using SVM+ Privileged Information for Manufacturing Process Modeling. Soft Comput. 2017;21(22): 6811. https://doi.org/10.1007/s00500-016-2232-9.
28. Popov A. Networks Word Sense Disambiguation with Recurrent Neural. In: Proceedings of the student research workshop associated with RANLP 2017 INCOMA ltd. Varna; 2017. p. 25–34. https://doi.org/10.26615/issn.1314-9156.2017_004.
29. Simov KI, Koprinkova-Hristova PD, Popov A, Osenova P. Word Embeddings Improvement via Echo State Networks. In: IEEE International Symposium on INnovations in Intelligent SysTems and Applications, INISTA 2019, Sofia, Bulgaria, July 3-5, 2019; 2019. p. 1–6. https://doi.org/10.1109/INISTA.2019.8778297.
30. Koprinkova-Hristova PD, Tontchev N. Echo State Networks for Multi-dimensional Data Clustering. In: Artificial Neural Networks and Machine Learning - ICANN 2012 - 22nd International Conference on Artificial Neural Networks, Lausanne, Switzerland, September 11-14, 2012, Proceedings, Part I; 2012. p. 571–578. https://doi.org/10.1007/978-3-642-33269-2_72.
31. Gallicchio C, Micheli A. A reservoir computing approach for human gesture recognition from kinect data inproceedings of the AI for ambient assisted living. 2016.
32. Raganato A, Camacho-Collados J, Navigli R. Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers Association for Computational Linguistics, Valencia, Spain; 2017. p. 99–110. https://www.aclweb.org/anthology/E17-1010.
33. Koprinkova-Hristova P. Multi-dimensional Data Clustering and Visualization via Echo State Networks Springer International Publishing, Cham. 2016. https://doi.org/10.1007/978-3-319-32192-9_3.
34. Jaeger H. Discovering multiscale dynamical features with hierarchical echo state networks. Tech. rep., Jacobs University Bremen. 2007.
35. Fernández S, Graves A, Schmidhuber J. Sequence Labelling in Structured Domains with Hierarchical Recurrent Neural Networks. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'07; 2007. p. 774–779.
36. Triefenbach F, Jalalvand A, Schrauwen B, Pierre Martens J. Phoneme Recognition with Large Hierarchical Reservoirs. In: Advances in Neural Information Processing Systems 23, ed. by J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, A. Culotta (Curran Associates, Inc.; 2010. p. 2307–2315. http://papers.nips.cc/paper/4056-phoneme-recognition-with-large-hierarchical-reservoirs.pdf.
37. Triefenbach F, Jalalvand A, Demuynck K, Martens J. Acoustic modeling with hierarchical reservoirs, IEEE transactions on audio. Speech, and Language Processing. 2013;21(11):2439. https://doi.org/10.1109/TASL.2013.2280209.
38. Triefenbach F, Demuynck K, Martens J. Large vocabulary continuous speech recognition with reservoir-Based acoustic models. IEEE Signal Processing Letters. 2014;21(3):311. https://doi.org/10.1109/LSP.2014.2302080.
39. Bengio Y, Lee DH, Bornschein J, Lin Z. Towards biologically plausible deep learning. 2015. arXiv:1502.04156.
40. Fellbaum C. Wordnet Springer Netherlands. 2010. https://doi.org/10.1007/978-90-481-8847-5_10.
41. G.A. Miller, Leacock C, Tengi R, Bunker RT. A Semantic Concordance. In: Proceedings of the Workshop on Human Language Technology Association for Computational Linguistics, Stroudsburg, PA, USA,HLT '93, pp. 303–308; 1993. https://doi.org/10.3115/1075671.1075742.
42. Edmonds P, Cotton S. SENSEVAL-2: Overview. In: Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems Association for Computational Linguistics, Toulouse, France; 2001. p. 1–5. https://www.aclweb.org/anthology/S01-1001.
43. Snyder B, Palmer M. The English all-words task. In: Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text Association for Computational Linguistics, Barcelona, Spain; 2004. p. 41–43. https://www.aclweb.org/anthology/W04-0811.
44. Pradhan S, Loper E, Dligach D, Palmer M. SemEval-2007 Task-17: English Lexical Sample, SRL and All Words. In: Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007) Association for Computational Linguistics, Prague, Czech Republic; 2007. p. 87–92. https://www.aclweb.org/anthology/S07-1016.
45. Navigli R, Jurgens D, Vannella D. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In: Second Joint Conference on Lexical and Computational Semantics *SEM, Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation SemEval 2013 Association for Computational Linguistics, Atlanta, Georgia, USA; 2013. p. 222–231. https://www.aclweb.org/anthology/S13-2040.
46. Moro A, Navigli R. SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015) (Association for Computational Linguistics, Denver, Colorado); 2015. p. 288–297. https://doi.org/10.18653/v1/S15-2049.
47. Yuxin C, Le-Ngoc T, Champagne B, Changjiang X. Recursive least squares constant modulus algorithm for blind adaptive array. IEEE Transactions on Signal Processing. 2004;52(5):1452.
48. Slobodyan S, Bogomolova A, Kolyuzhnov D. Stochastic gradient versus recursive least squares learning. SRNN CERGE-EI Working Paper. 2006;309:1. https://doi.org/10.2139/ssrn.1129821.
49. Belkin M, Hsu D, Ma S, Mandal S. Reconciling modern machine learning practice and the bias-variance trade-off arxiv: Machine Learning. 2018.