

# EFFICIENT TEXT CLASSIFICATION WITH ECHO STATE NETWORKS

Jérémie Cabessa<sup>1,2,3</sup>, Hugo Hernault<sup>1</sup>, Heechang Kim<sup>1</sup>,  
Yves Lamonato<sup>1</sup> and Yariv Z. Levy<sup>1</sup>

<sup>1</sup>Playtika Research  
Lausanne, Switzerland  
&

<sup>2</sup>Department of Mathematical Economics  
University Paris II, France  
&

<sup>3</sup>Institute of Computer Science of the Czech Academy of Sciences  
Prague, Czech Republic

IJCNN 21, 18–22 July, 2021

# INTRODUCTION

- We consider Echo State Networks (ESNs) in the context of Natural Language Processing (NLP).
- More specifically, we apply ESNs with pre-trained word embeddings (GloVe) as inputs for text classification tasks.
- We show that ESNs are robust, efficient and fast candidates for text classification tasks.
- Some works about text classification with ESNs have already been done, but along different lines (different training paradigms).

# INTRODUCTION

- We consider Echo State Networks (ESNs) in the context of Natural Language Processing (NLP).
- More specifically, we apply ESNs with pre-trained word embeddings (GloVe) as inputs for text classification tasks.
- We show that ESNs are robust, efficient and fast candidates for text classification tasks.
- Some works about text classification with ESNs have already been done, but along different lines (different training paradigms).

# INTRODUCTION

- We consider Echo State Networks (ESNs) in the context of Natural Language Processing (NLP).
- More specifically, we apply ESNs with pre-trained word embeddings (GloVe) as inputs for text classification tasks.
- We show that ESNs are robust, efficient and fast candidates for text classification tasks.
- Some works about text classification with ESNs have already been done, but along different lines (different training paradigms).

# INTRODUCTION

- We consider Echo State Networks (ESNs) in the context of Natural Language Processing (NLP).
- More specifically, we apply ESNs with pre-trained word embeddings (GloVe) as inputs for text classification tasks.
- We show that ESNs are robust, efficient and fast candidates for text classification tasks.
- Some works about text classification with ESNs have already been done, but along different lines (different training paradigms).

# DATASETS

- **IMDb:** Sentiment Analysis. Large Movie Review Dataset (IMDb) for binary **sentiment classification**.
- **TREC:** Text REtrieval Conference dataset (TREC) for **question classification**.
  - TREC-6: questions classified into 6 classes (coarse-grained)
  - TREC-8: questions classified into 80 classes (fine-grained)

# DATASETS

- **IMDb:** Sentiment Analysis. Large Movie Review Dataset (IMDb) for binary **sentiment classification**.
- **TREC:** Text REtrieval Conference dataset (TREC) for **question classification**.
  - ▶ **TREC-6:** questions classified into 6 classes (coarse-grained).
  - ▶ **TREC-50:** questions classified into 50 classes (fine-grained).

# DATASETS

- **IMDb:** Sentiment Analysis. Large Movie Review Dataset (IMDb) for binary **sentiment classification**.
- **TREC:** Text REtrieval Conference dataset (TREC) for **question classification**.
  - ▶ **TREC-6:** questions classified into 6 classes (coarse-grained).
  - ▶ **TREC-50:** questions classified into 50 classes (fine-grained).



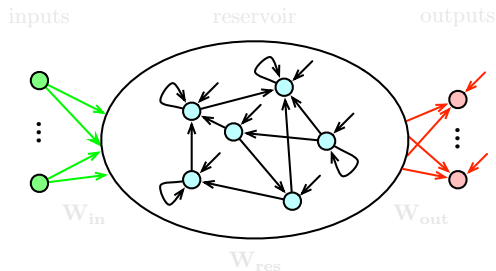
# DATASETS

- **IMDb:** Sentiment Analysis. Large Movie Review Dataset (IMDb) for binary **sentiment classification**.
- **TREC:** Text REtrieval Conference dataset (TREC) for **question classification**.
  - ▶ **TREC-6:** questions classified into 6 classes (coarse-grained).
  - ▶ **TREC-50:** questions classified into 50 classes (fine-grained).

# ECHO STATE NETWORKS

**Echo State Networks (ESNs)** are specific kinds of recurrent neural networks. An ESN consists of:

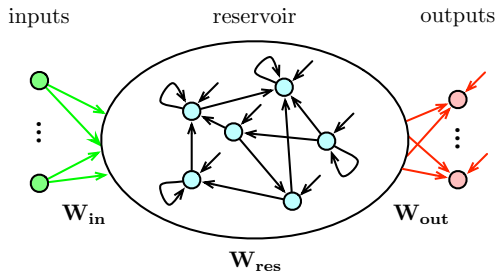
- An input layer
- A reservoir of neurons: random, recurrent and sparse
- An output layer
- A delay buffer
- A delay filter



# ECHO STATE NETWORKS

**Echo State Networks (ESNs)** are specific kinds of recurrent neural networks. An ESN consists of:

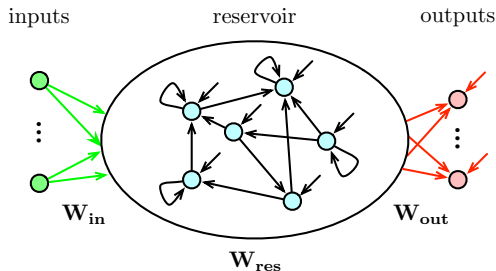
- An *input layer*
- A *reservoir* of neurons: random, recurrent and sparse
- An *output layer*
- Only output weights are trained!



# ECHO STATE NETWORKS

**Echo State Networks (ESNs)** are specific kinds of recurrent neural networks. An ESN consists of:

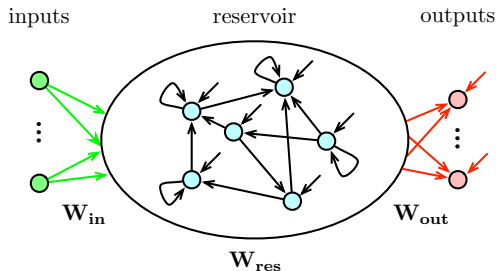
- An *input layer*
- A *reservoir* of neurons: random, recurrent and sparse
- An *output layer*
- Only output weights are trained!



# ECHO STATE NETWORKS

**Echo State Networks (ESNs)** are specific kinds of recurrent neural networks. An ESN consists of:

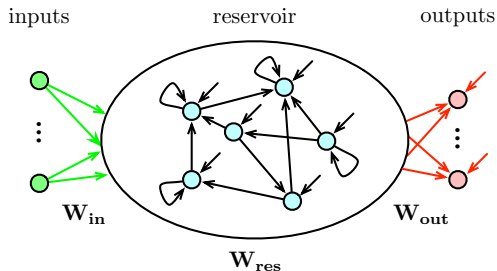
- An *input layer*
- A *reservoir* of neurons: random, recurrent and sparse
- An *output layer*
- Only output weights are trained!



# ECHO STATE NETWORKS

**Echo State Networks (ESNs)** are specific kinds of recurrent neural networks. An ESN consists of:

- An *input layer*
- A *reservoir* of neurons: random, recurrent and sparse
- An *output layer*
- Only output weights are trained!



# ESNs: INITIALIZATION

- Input weights  $\mathbf{W}_{\text{in}}$  (fixed):

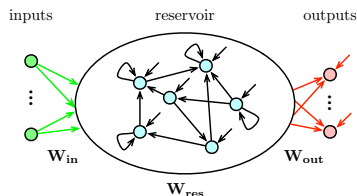
$\mathbf{W}_{\text{in}}$  sampled from  $\mathcal{U}(-a, a)$ , where  $a$  is the *input scaling*.

- Reservoir weights  $\mathbf{W}_{\text{res}}$  (fixed):

$\mathbf{W}_{\text{res}}$  sampled from  $\mathcal{U}(-1, 1)$ , set to 0 with *sparsity rate* 0.99, and rescaled to have a specific *spectral radius*  $\rho < 1$ .

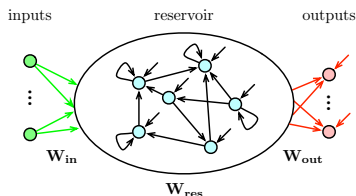
- Output weights  $\mathbf{W}_{\text{out}}$  (trainable!):

Here,  $\mathbf{W}_{\text{out}}$  closed-form solution of a simple Ridge Regression.



# ESNs: INITIALIZATION

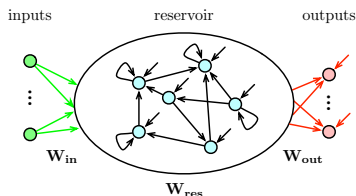
- Input weights  $\mathbf{W}_{\text{in}}$  (fixed):  
 $\mathbf{W}_{\text{in}}$  sampled from  $\mathcal{U}(-a, a)$ , where  $a$  is the *input scaling*.
- Reservoir weights  $\mathbf{W}_{\text{res}}$  (fixed):  
 $\mathbf{W}_{\text{res}}$  sampled from  $\mathcal{U}(-1, 1)$ , set to 0 with *sparsity rate* 0.99, and rescaled to have a specific *spectral radius*  $\rho < 1$ .
- Output weights  $\mathbf{W}_{\text{out}}$  (trainable!):  
Here,  $\mathbf{W}_{\text{out}}$  closed-form solution of a simple Ridge Regression.





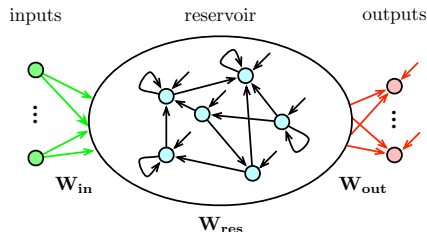
# ESNs: INITIALIZATION

- Input weights  $\mathbf{W}_{\text{in}}$  (fixed):  
 $\mathbf{W}_{\text{in}}$  sampled from  $\mathcal{U}(-a, a)$ , where  $a$  is the *input scaling*.
- Reservoir weights  $\mathbf{W}_{\text{res}}$  (fixed):  
 $\mathbf{W}_{\text{res}}$  sampled from  $\mathcal{U}(-1, 1)$ , set to 0 with *sparsity rate* 0.99, and rescaled to have a specific *spectral radius*  $\rho < 1$ .
- Output weights  $\mathbf{W}_{\text{out}}$  (trainable!):  
Here,  $\mathbf{W}_{\text{out}}$  closed-form solution of a simple Ridge Regression.



# ESNs: DYNAMICS

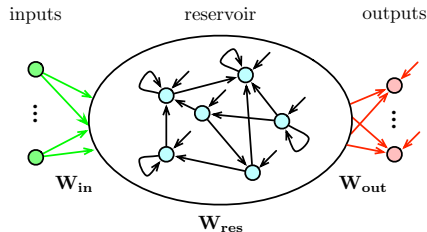
- We consider **leaky integrator ESNs (LI-ESNs)**:



$$\begin{aligned}\tilde{\mathbf{x}}(t+1) &= f_{res}\left(\mathbf{W}_{in}\mathbf{u}(t+1) + \mathbf{W}_{res}\mathbf{x}(t)\right) \\ \mathbf{x}(t+1) &= (1 - \alpha)\mathbf{x}(t) + \alpha\tilde{\mathbf{x}}(t+1) \quad \alpha \text{ is the } \textit{leaking rate} \\ \mathbf{y}(t+1) &= f_{out}\left(\mathbf{W}_{out}\mathbf{x}(t+1)\right)\end{aligned}$$

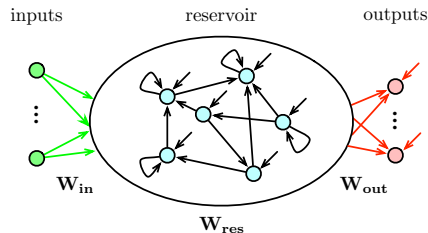
# ESNs: INPUT FEATURES

- We consider the pre-trained word embedding **GloVe-840B-300d** to obtain the input features.
- GloVe-840B-300d is a 300-dimensional vector representation for words (obtained via some unsupervised learning algorithm).
- Any input text is embedded into a sequence of 300-dimensional vectors.



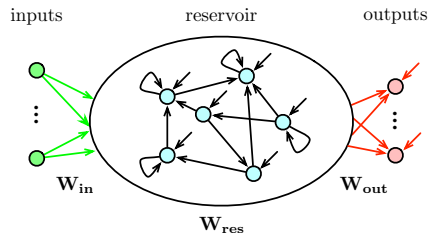
# ESNs: INPUT FEATURES

- We consider the pre-trained word embedding **GloVe-840B-300d** to obtain the input features.
- GloVe-840B-300d is a 300-dimensional vector representation for words (obtained via some unsupervised learning algorithm).
- Any input text is embedded into a sequence of 300-dimensional vectors.



# ESNs: INPUT FEATURES

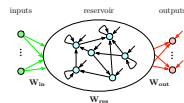
- We consider the pre-trained word embedding **GloVe-840B-300d** to obtain the input features.
- GloVe-840B-300d is a 300-dimensional vector representation for words (obtained via some unsupervised learning algorithm).
- Any input text is embedded into a sequence of 300-dimensional vectors.



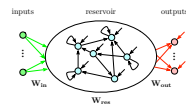
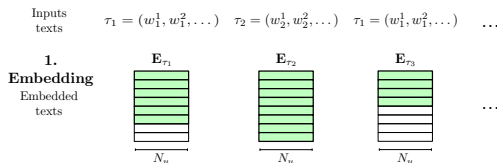
# ESNs: TRAINING

Inputs  
texts

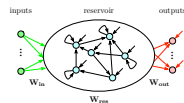
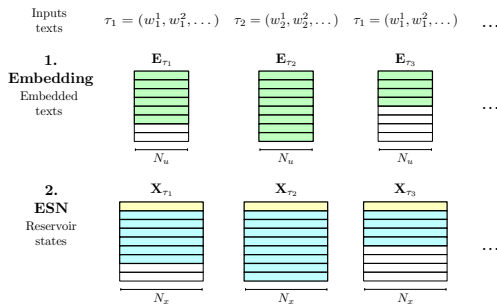
$$\tau_1 = (w_1^1, w_1^2, \dots) \quad \tau_2 = (w_2^1, w_2^2, \dots) \quad \tau_1 = (w_1^1, w_1^2, \dots) \quad \dots$$



# ESNs: TRAINING

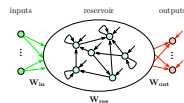
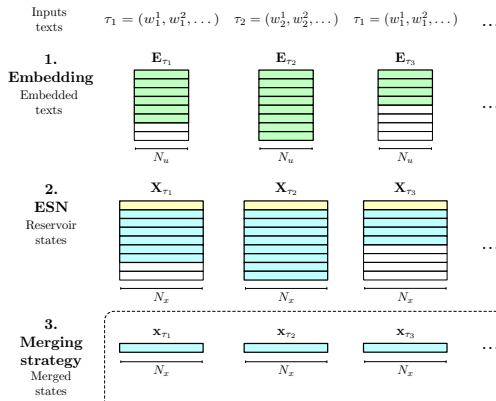


# ESNs: TRAINING

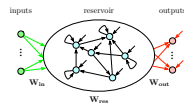
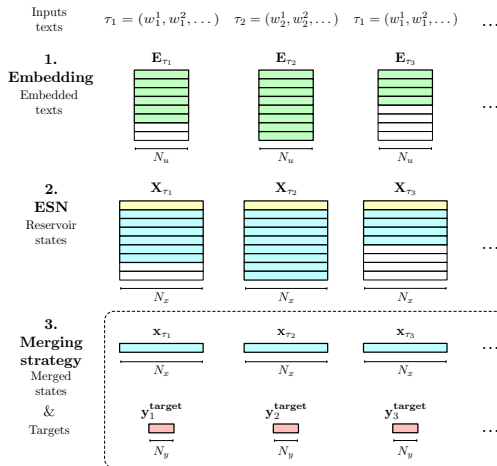




# ESNs: TRAINING



# ESNs: TRAINING



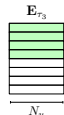
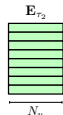
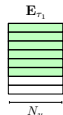
# ESNs: TRAINING

Inputs  
texts

$$\tau_1 = (w_1^1, w_1^2, \dots) \quad \tau_2 = (w_2^1, w_2^2, \dots) \quad \tau_3 = (w_3^1, w_3^2, \dots) \quad \dots$$

1.  
Embedding

Embedded  
texts



...

2.

ESN

Reservoir  
states



...

3.  
Merging  
strategy

Merged  
states

&

Targets



...



...

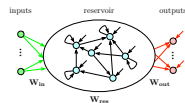
4.

Learning  
algorithm  
(Ridge regression)



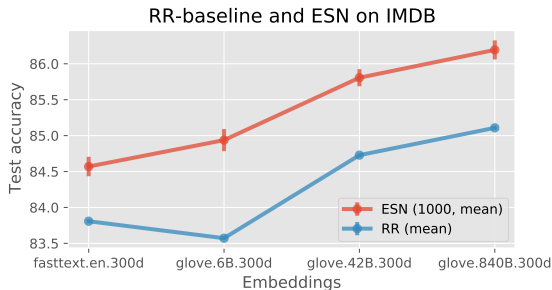
output weights  
(closed-form solution)

$$\mathbf{W}_{\text{out}}^T = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y}^{\text{target}}$$



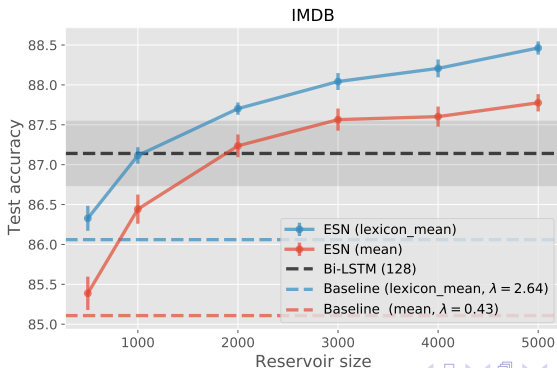
## RESULTS: EFFECT OF EMBEDDING

- The quality of the pre-trained embedding plays an important role (might be counter intuitive).



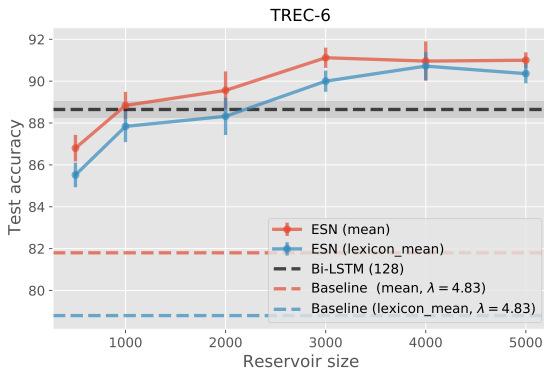
## RESULTS: EFFECT OF THE RESERVOIR

- ▶  $\text{ESN} = \text{EMB} + \text{RES} + \text{RR}$ ;  $\text{RR-baseline} = \text{EMB} + \text{RR}$
- ⇒ ESN vs RR-baseline allows to evaluate the contribution of the reservoir.
- **The temporal dynamics captured by the reservoir drastically improves the results.**



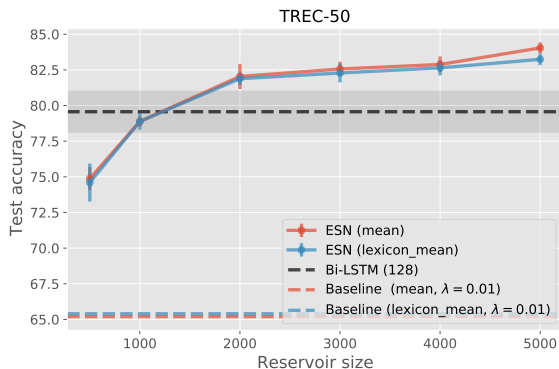
## RESULTS: EFFECT OF THE RESERVOIR

- ▶  $\text{ESN} = \text{EMB} + \text{RES} + \text{RR}$ ;  $\text{RR-baseline} = \text{EMB} + \text{RR}$
- ⇒ ESN vs RR-baseline allows to evaluate the contribution of the reservoir.
- **The temporal dynamics captured by the reservoir drastically improves the results.**



## RESULTS: EFFECT OF THE RESERVOIR

- ▶  $\text{ESN} = \text{EMB} + \text{RES} + \text{RR}$ ;  $\text{RR-baseline} = \text{EMB} + \text{RR}$
- ⇒ ESN vs RR-baseline allows to evaluate the contribution of the reservoir.
- **The temporal dynamics captured by the reservoir drastically improves the results.**



# RESULTS: COMPARISON WITH BI-LSTM NETWORKS

- **ESNs achieve competitive results with a significantly faster training time than Bi-LSTM.**<sup>1</sup>

	IMDb	
	Accuracy (%)	Training time (sec.)
RR (none)	76.25	16.97 (batch=64) <sup>2</sup>
RR (mean)	85.11	3.74 (batch=2048)
RR (lex. mean)	86.06	2.49 (batch=2048)
Bi-LSTM (128)	87.14 ± 0.40	899.32 ± 4.01
<b>ESN (5000, mean)</b>	87.78 ± 0.11	38.47 ± 0.14
<b>ESN (5000, lex. mean)</b>	<b>88.46 ± 0.08</b>	<b>39.10 ± 0.23</b>

**TABLE:** Test accuracy and training time of the RR-baselines, Bi-LSTM and ESNs over the IMDb dataset.

<sup>1</sup>1 GPU, 32 GB, NVIDIA V100

<sup>2</sup>Out of memory on GPU; hence trained on CPU



# RESULTS: COMPARISON WITH BI-LSTM NETWORKS

- **ESNs achieve competitive results with a significantly faster training time than Bi-LSTM.**<sup>3</sup>

	TREC-6	
	Accuracy (%)	Training time (sec.)
RR (none)	56.20	1.92 (batch=2048)
RR (mean)	81.80	0.42 (batch=2048)
RR (lex. mean)	78.80	0.13 (batch=2048)
Bi-LSTM (128)	$88.65 \pm 0.38$	$27.04 \pm 5.11$
<b>ESN (3000, mean)</b>	<b><math>91.12 \pm 0.48</math></b>	<b><math>2.64 \pm 0.12</math></b>
<b>ESN (4000, lex. mean)</b>	$90.72 \pm 0.66$	$4.35 \pm 0.09$

**TABLE:** Test accuracy and training time of the RR-baselines, Bi-LSTM and ESNs over the TREC-6 dataset.

<sup>3</sup>1 GPU, 32 GB, NVIDIA V100

# RESULTS: COMPARISON WITH BI-LSTM NETWORKS

- **ESNs achieve competitive results with a significantly faster training time than Bi-LSTM.**<sup>4</sup>

	TREC-50	
	Accuracy (%)	Training time (sec.)
RR (none)	27.40	1.81 (batch=2048)
RR (mean)	65.20	0.15 (batch=2048)
RR (lex. mean)	65.40	0.13 (batch=2048)
Bi-LSTM (128)	$79.56 \pm 1.43$	$27.12 \pm 0.93$
<b>ESN (5000, mean)</b>	<b><math>83.96 \pm 0.23</math></b>	<b><math>7.02 \pm 0.14</math></b>
<b>ESN (5000, lex. mean)</b>	$83.32 \pm 0.27$	$7.13 \pm 0.23$

**TABLE:** Test accuracy and training time of the RR-baselines, Bi-LSTM and ESNs over the TREC-50 dataset.

<sup>4</sup>1 GPU, 32 GB, NVIDIA V100

# CONCLUSIONS

- ESNs can be considered as robust, efficient and fast candidates for text classification tasks.
- State-of-the-art NLP are generally children of the “transformer revolution” (GPT, GPT-2, GPT-3, BERT, ...). Achieve impressive performance but can be very resource consuming.
- By contrast, our work fits within the context of light and fast-to-train models for NLP.

# CONCLUSIONS

- ESNs can be considered as robust, efficient and fast candidates for text classification tasks.
- State-of-the-art NLP are generally children of the “transformer revolution” (GPT, GPT-2, GPT-3, BERT, ...). Achieve impressive performance but can be very resource consuming.
- By contrast, our work fits within the context of light and fast-to-train models for NLP.

# CONCLUSIONS

- ESNs can be considered as robust, efficient and fast candidates for text classification tasks.
- State-of-the-art NLP are generally children of the “transformer revolution” (GPT, GPT-2, GPT-3, BERT, ...). Achieve impressive performance but can be very resource consuming.
- By contrast, our work fits within the context of light and fast-to-train models for NLP.

# CONCLUSIONS

**Thank you!**