2018 SYMPOSIUM SERIES ON COMPUTATIONAL INTELLIGENCE
18 - 21 NOVEMBER, 2018, BENGALURU, INDIA

Retrieving Impressions from Semantic Memory Modeled with Associative Pulsing Neural Networks

Patryk Orzechowski
patryk.orzechowski@gmail.com

Adrian Horzyk
horzyk@agh.edu.pl

Janusz A. Starzyk
starzykj@ohio.edu

Jason H. Moore
jhmoore@upenn.edu

Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104, USA

AGH University of Science and Technology
Krakow, Poland

Ohio University, Athens, Ohio, U.S.A.
School of Electrical Engineering and Computer Science

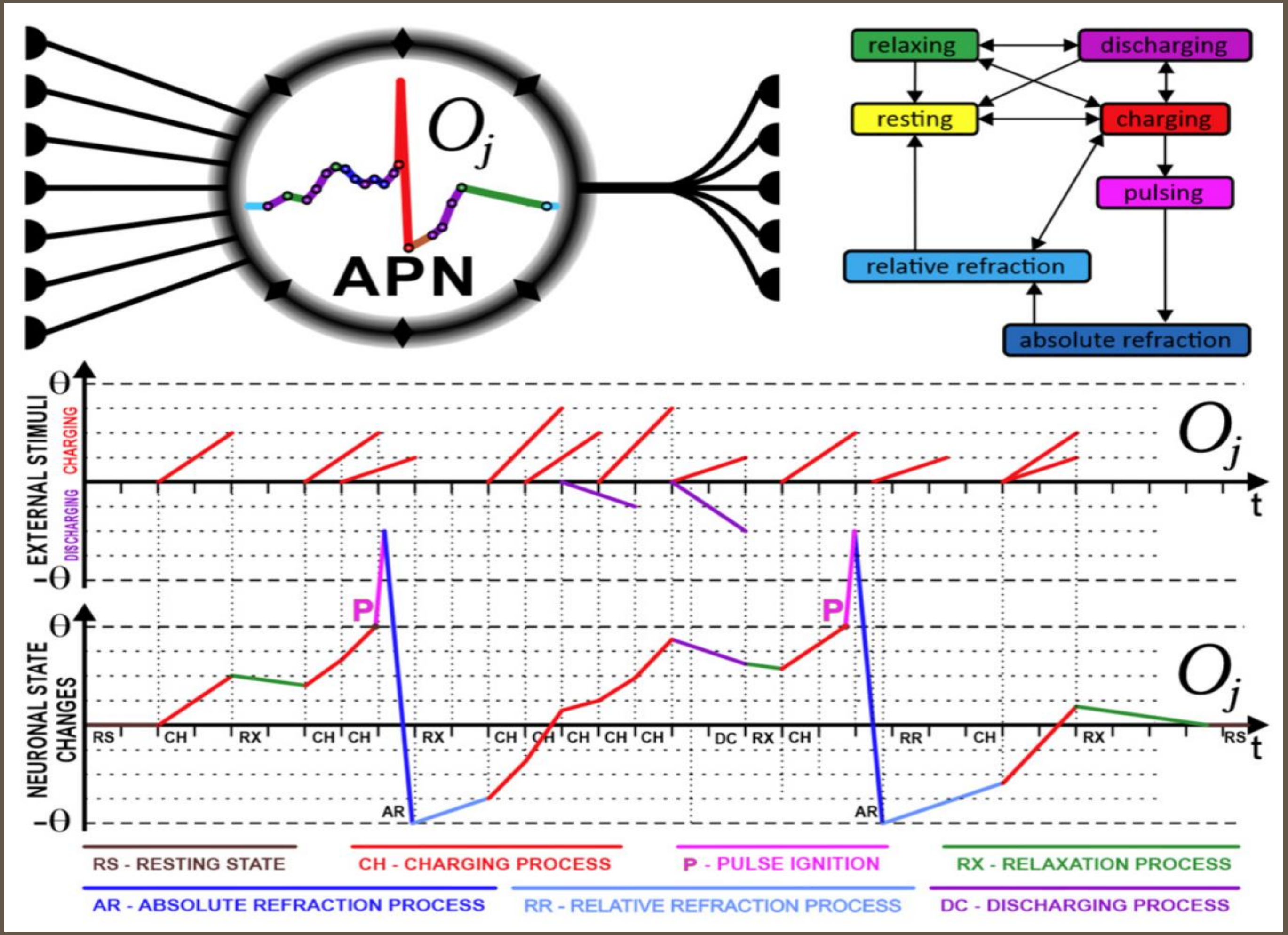
University of Information Technology and Management
Rzeszow, Poland

Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104, USA

Research inspired by brains and biological neurons

- ✓ Work asynchronously and in parallel
- ✓ Associate stimuli context-sensitively
- ✓ Self-organize neurons developing very complex structures
- ✓ Use time approach for computations
- ✓ Aggregate representation of similar data
- ✓ Represent various data and their relations
- ✓ Integrate memory and the procedures
- ✓ Provide plasticity to develop a structure to represent data and object relations

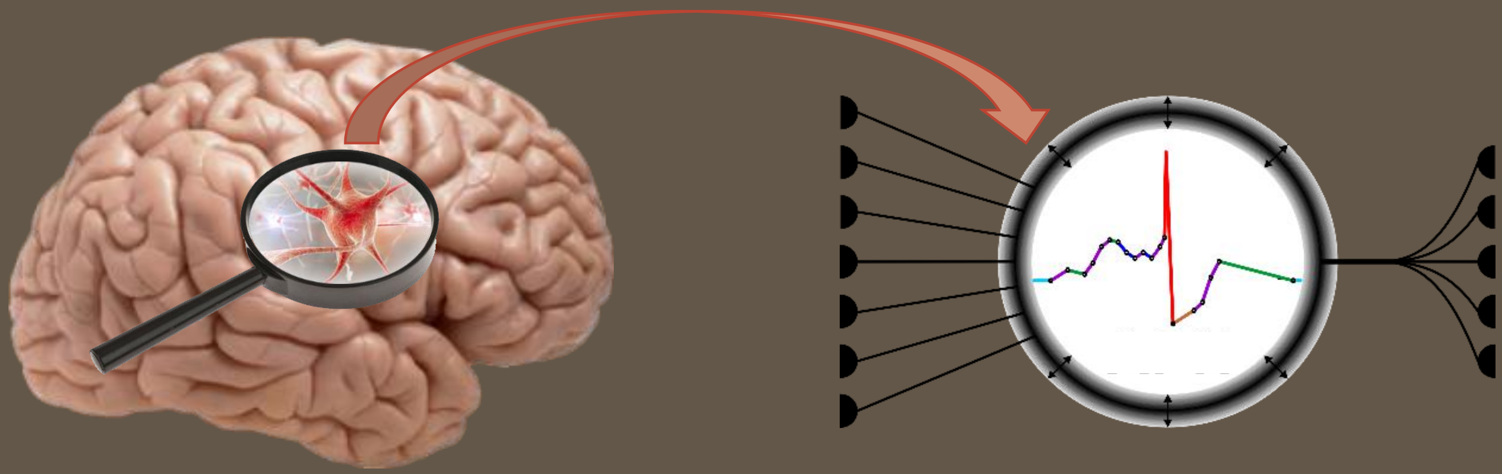
ASSOCIATIVE PULSING NEURONS



- ✓ Associative Pulsing Neurons can be used for retrieving Impressions from semantic memory representing a bag of words.

Associative Pulsing Neurons APN

- ✓ Were developed to reproduce plastic and associative functionalities of real neurons that work in time.



- ✓ They implement internal neuronal processes (IP) efficiently managed through internal process queues (IPQ) and a global event queue (GEQ).
- ✓ Connection weights are updated only for associated events resulting in associative graphs of APN neurons.
- ✓ APN neurons are updated only at the end of the internal processes to be efficient in data processing!

Objectives and Contribution

- ✓ **Construction of Associative Pulsing Neural Networks (APNN) to self-organize network structure for a bag of words (BOW).**
- ✓ **Use of these networks to provide easy interpretable and intuitive results because the results are represented by the number of pulses of the most associated neurons.**

APN Neurons

- ✓ Connected to emphasize a defining relation between words and sequences in the APNN neural network.
- ✓ Aggregate representations of the same words of the training sentences - no duplicates!
- ✓ Work asynchronously parallel because time is a computational factor which influences the results of the APNN neural networks.
- ✓ Integrate memory and associative processes
- ✓ Construction and training of APNN is very fast.

Bag of Words

Bag of words associates each word with the number of times it appears in a document.

Source: https://i0.wp.com/thecaffeinedev.com/wp-content/uploads/2017/12/bag.jpg

Retrieving Impressions

- ✓ This research uses **a bag of words approach** to find associations.
- ✓ Bag of words associates a given word w_i with the number of times it appears in a document d_j

$$b(d_j) = \{(w_i, c(w_i)) : c(w_i) = \sum_{k=1}^{n_i} \delta(w_i, w_k)\}$$

- where $b(d_j)$ *is* a set of pairs, associating a given word w_i with the number of times it appears in a document

$$d_j = \left(w_{1j}, w_{2j}, \ldots w_{n_i j}\right)$$

- $c(w_i)$ is the number of occurrences of a given word w_i in document d_i:
- n_i is the number of words in the document d_i:

$$\delta(x, y) = \begin{cases} 1 \; if \; x = y \\ 0 \; if \; x \neq y \end{cases}$$

Making Associations

We studied three techniques of ranking documents according to their relevance to the specific terms called:

- ✓ term frequency *(tf)*

$$tf\left(w_i, d_j\right) = \frac{\#w_i}{length(d_j)}$$

- ✓ inverse document frequency *(idf).*

$$idf\left(w_i, d_j,\right) = log\,\frac{N}{\#\left\{d_j\colon w_i \in d_j\right\}} + 1$$

- ✓ and their combination *(tfidf)*

$$tfidf\left(w_i, d_j\right) = tf\left(w_i, d_j\right) * idf\left(w_i, d_j,\right)$$

Method and Model Description

- APNN network was spanned on top of the bag of words created for the input text (a set of sequences).
- Each unique word was represented as a separate APN neuron. Repeated words were represented by the same APN neurons.
- Activation of a neuron sent a signal to the connected neurons increasing their potential.
- Original APN model was modified to include:
  - Neuron attributes were stored in dictionaries instead of Attribute-Value B-Trees (AVB-trees).
  - Internal neuron processes queue stores only current external stimuli events.
  - The logic of the neuron activity has been shifted towards neuron controller and global coordinator.

Method Description

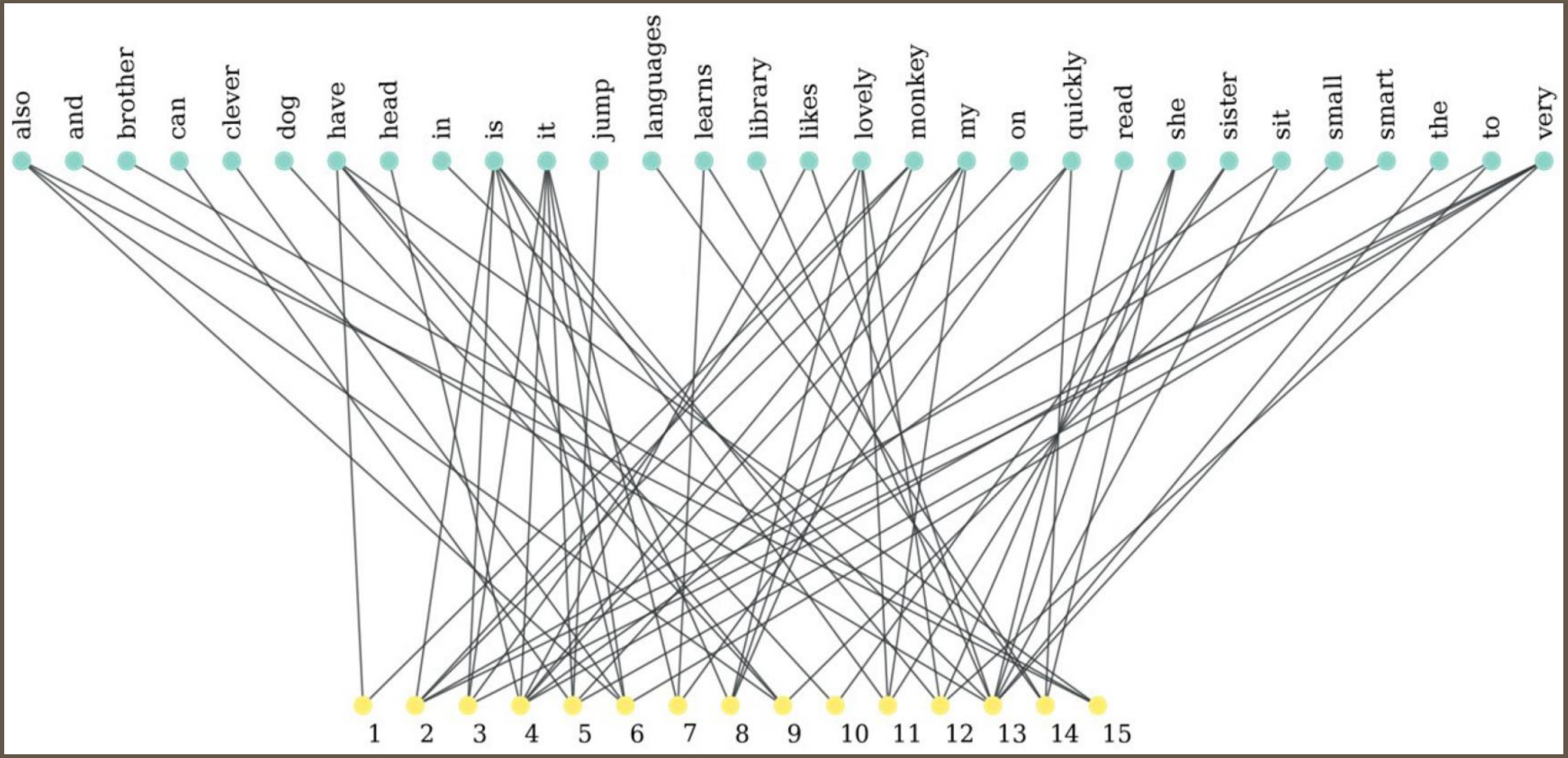
Two strategies of setting weights in the network were compared:

- CountVectorizer sets the weights from documents to words according to term frequency.
- TfIdfVectorizer sets the weights according to the product of term frequency and inverse document frequency.

Parameters of APNN used in simulation were:

Simulation parameter	Value
chargingPeriod	1
dischargingPeriod	1
relaxationPeriod	20
absrefractionPeriod	2
relrefractionPeriod	10
simulationTime	100

Example APNN Network for the Bag of Words Approach



Training data used for the creation of the APNN network:

I have a monkey. My monkey is very smart. It is very lovely. It likes to sit on my head. It can jump very quickly. It is also very clever. It learns quickly. My monkey is lovely. I also have a small dog. I have a sister. My sister is lovely. She is very lovely. She likes to sit in the library and to read. She quickly learns languages. I also have a brother.

Experimental Results

- Tests observed the network response to different words or phrases, e.g. ‘monkey’, ‘monkey is’, ‘she is’ etc.
- The neurons that spiked for the first scenario using term frequency weights will be presented in the next slide.
- The achieved pulse frequency will tell us how much the represented words are associated with the calling context constructed from different words.

Experimental Results

- The neurons that spiked using term frequency weights are presented in the table below.
  - The values in brackets correspond to the number of spikes (pulse frequency) observed during simulation.

Stimuli	Impressions
monkey(35)	have(4), is(3), my(3), lovely(3), very(1)
is(35) monkey(35)	lovely(9), very(8), my(7), it(5), have(4), sister(2), also(2), smart(1)
is(35) she(35)	lovely(8), very(8), my(5), it(5), quickly(4), monkey(3), learns(2)
she(35) sister(35)	is(4), lovely(4), have(4), very(2), my(2)
lovely(35)	is(9), very(6), my(5), it(3), monkey(2)
also(35) brother(35)	have(6)
sit(35) library(35)	to(4)
jump(35)	-

  - CountVectorizer is a 2D table which sets APNN weights based on the number of words (stored in columns) in each of the documents (stored in rows).

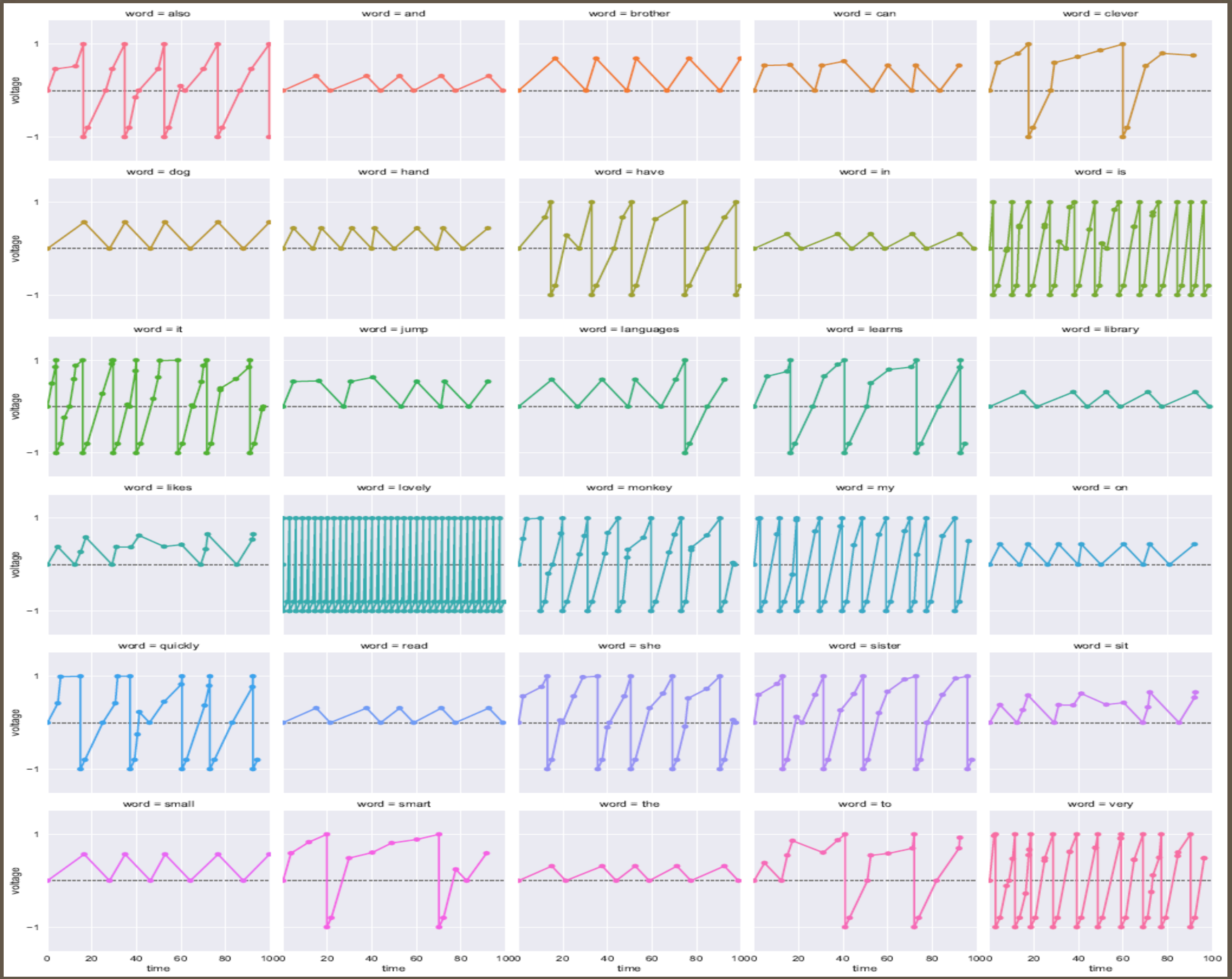
Experimental Results

- The neurons that spiked using TfIdfVectorizer are

Stimuli	Impressions
monkey(35)	my(7), is(7), lovely(7), very(7), it(7), quickly(7), learns(7), smart(5), have(4), also(4), sister(4), she(3)
is(35)	lovely(15), very(15), my(14), smart(9), it(8), have(5), sister(5),
monkey(35)	she(5), also(5), quickly(5), learns(5), clever(4), to(2)
she(35)	lovely(15), very(15), my(14), monkey(8), it(8), quickly(7), learns(7),
is(35)	to(6), languages(5), clever(5), likes(4), sit(4), sister(4), smart(4), have(3), also(3)
she(35)	lovely(9), is(8), very(8), my(8), it(8), quickly(8), learns(8), monkey(6),
sister(35)	have(6), also(6), to(6), languages(4), likes(3), sit(3)
lovely(35)	is(12), very(10), my(10), it(7), monkey(7), sister(5),she(5), have(5), quickly(5), also(5), learns(4), clever(2), smart(2), to(2)
also(35)	have(9), dog(4), small(4), clever(4), very(2), it(2),
brother(35)	is(2), monkey(2), sister(2), lovely(2), my(2), quickly(2), learns(2)
sit(35) library(35)	to(9), likes(6), and(4), in(4), read(4), the(4), head(3), on(3), she(2)
jump(35)	can(4), quickly(2), it(2), learns(2), very(2), is(2), lovely(2), my(2), monkey(2)

- TfIdfVectorizer sets the weights of APNN based on the frequency of words appearing across all documents.

APN Neurons Activations



Time response of APNN tested using 'lovely' input.
Most active impressions are: 'is', 'very', and 'my',
followed by 'it' and 'monkey'.

Experimental Results

- Setting tf-idf network weights allow retrieving deeper associations compared to using tf weights.
  - For example, activation of "monkey", using tf-idf network retrieves such impressions as monkey learns, quickly, smart, or association between monkey and it, which aren't retrieved using tf scenario.
  - Response to "she is" includes clever, likes, sit, smart, sister and languages not present in tf scenario.

- Disadvantage is that retrieving impressions using tf-idf network are associated with more noisy response,
  - e.g. monkey becomes wrongly associated with she and sister (as both appear in contexts of very lovely and I have).

Comparison to Other Methods from Literature

- Comparison to LSA/LSI Gensim software [7] .
  - For example, activation of “monkey” yields ‘sister’, ‘lovely’, ‘very’, ‘it’, ‘she’, ‘likes’, ‘sit’, ‘is’, and ‘learns’.
  - It misses important associations like ‘my’, ‘quickly’, and ‘smart’, while still providing wrong associations with ‘sister’ and ‘she’.

- Comparison to LDA Gensim software [7],
  - Activation of “monkey” yields ‘have’, ‘I’, ‘is’, ‘very’, ‘it’, ‘quickly’, ‘my’, sit’, and ‘lovely’.
  - It misses important associations like ‘learns’ and ‘smart’ while providing strong association to less important word ‘I’.

Comparison of Bag of Words to Topic Modeling

TABLE IV
EXEMPLARY TOPICS WITH THEIR CONTRIBUTIONS MODELED ON THE TOY DATASET USING LSA/LSI

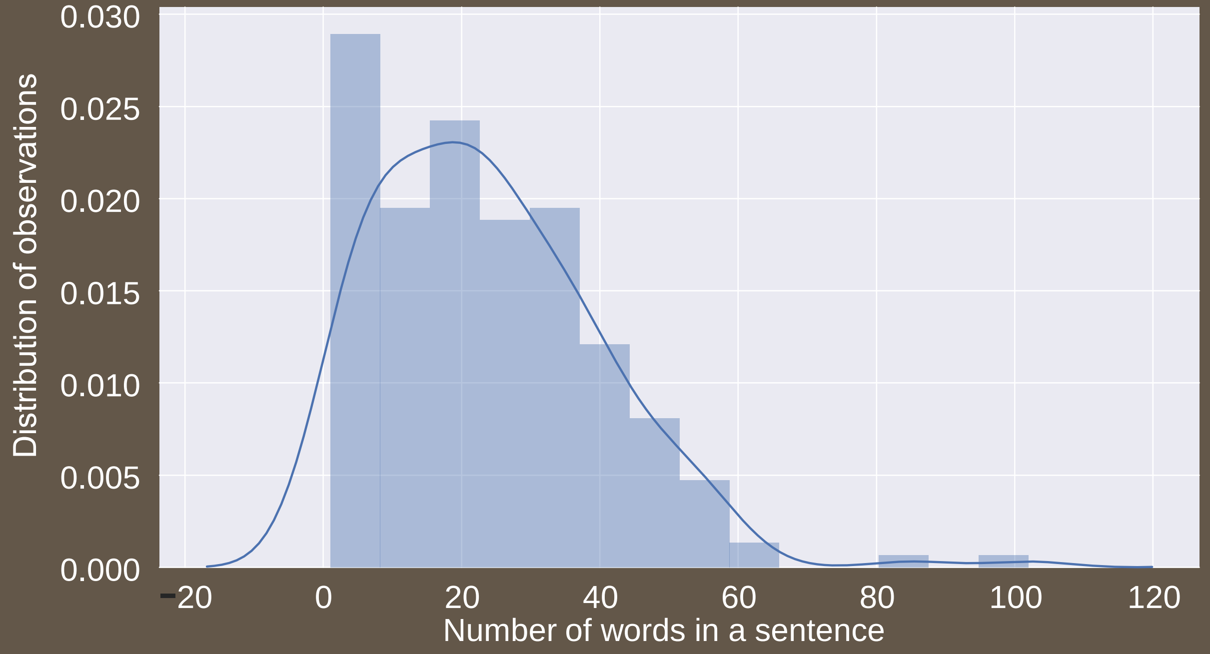
Topic ID	Word contributions
0	0.569*"is" + 0.446*"very" + 0.381*"lovely" + 0.333*"it" + 0.323*"my" + 0.211*"monkey" + 0.128*"she" + 0.125*"also" + 0.102*"quickly" + 0.100*"sister"
1	-0.633*"have" + -0.633*"i" + -0.348*"also" + -0.143*"sister" + -0.134*"monkey" + 0.090*"it" + 0.088*"very" + 0.077*"lovely" + 0.074*"is" + 0.062*"she"
2	0.530*"it" + 0.477*"quickly" + 0.314*"learns" + -0.266*"my" + -0.254*"lovely" + -0.248*"is" + -0.233*"monkey" + 0.175*"very" + 0.171*"she" + 0.159*"also"
3	-0.525*"sit" + -0.525*"likes" + -0.412*"my" + 0.356*"very" + -0.269*"she" + 0.163*"also" + 0.146*"is" + -0.109*"monkey" + -0.103*"sister" + -0.073*"learns"
4	0.497*"she" + 0.419*"learns" + -0.395*"it" + 0.384*"quickly" + 0.248*"lovely" + -0.227*"also" + -0.216*"likes" + -0.216*"sit" + 0.141*"sister" + -0.129*"very"
5	0.527*"she" + -0.391*"my" + -0.364*"monkey" + -0.337*"quickly" + -0.240*"learns" + -0.226*"it" + 0.221*"very" + 0.217*"also" + 0.200*"likes" + 0.200*"sit"
6	0.620*"monkey" + -0.575*"sister" + -0.357*"lovely" + 0.276*"very" + -0.211*"it" + 0.183*"she" + 0.032*"likes" + 0.032*"sit" + -0.022*"is" + -0.019*"learns"
7	0.717*"also" + -0.368*"very" + 0.297*"learns" + -0.257*"sister" + 0.250*"my" + 0.199*"is" + -0.166*"have" + -0.166*"i" + -0.124*"monkey" + -0.123*"it"
8	-0.610*"lovely" + 0.511*"sister" + 0.351*"very" + -0.288*"it" + 0.282*"my" + -0.154*"monkey" + 0.111*"also" + 0.111*"quickly" + 0.097*"she" + -0.085*"i"
9	0.441*"quickly" + -0.437*"learns" + -0.391*"is" + 0.389*"my" + 0.311*"lovely" + -0.251*"sister" + -0.227*"monkey" + -0.210*"it" + 0.193*"very" + 0.106*"also"

TABLE V
EXEMPLARY TOPICS WITH THEIR CONTRIBUTIONS MODELED ON THE TOY DATASET USING LDA

Topic ID	Word contributions
0	0.067*"is" + 0.067*"i" + 0.067*"have" + 0.067*"my" + 0.067*"very" + 0.067*"it" + 0.067*"quickly" + 0.067*"she" + 0.067*"lovely" + 0.067*"sit"
1	0.280*"she" + 0.147*"learns" + 0.147*"quickly" + 0.147*"likes" + 0.147*"sit" + 0.013*"i" + 0.013*"is" + 0.013*"it" + 0.013*"monkey" + 0.013*"my"
2	0.067*"i" + 0.067*"have" + 0.067*"is" + 0.067*"quickly" + 0.067*"it" + 0.067*"very" + 0.067*"she" + 0.067*"my" + 0.067*"also" + 0.067*"lovely"
3	0.067*"is" + 0.067*"very" + 0.067*"have" + 0.067*"it" + 0.067*"quickly" + 0.067*"i" + 0.067*"also" + 0.067*"lovely" + 0.067*"monkey" + 0.067*"she"
4	0.234*"is" + 0.177*"very" + 0.120*"lovely" + 0.120*"my" + 0.120*"monkey" + 0.063*"she" + 0.063*"also" + 0.063*"it" + 0.006*"i" + 0.006*"have"
5	0.067*"i" + 0.067*"it" + 0.067*"have" + 0.067*"very" + 0.067*"is" + 0.067*"quickly" + 0.067*"she" + 0.067*"my" + 0.067*"lovely" + 0.067*"sit"
6	0.168*"is" + 0.168*"lovely" + 0.168*"sister" + 0.088*"very" + 0.088*"it" + 0.088*"i" + 0.088*"have" + 0.088*"my" + 0.008*"quickly" + 0.008*"she"
7	0.244*"monkey" + 0.244*"have" + 0.244*"i" + 0.022*"is" + 0.022*"very" + 0.022*"it" + 0.022*"quickly" + 0.022*"my" + 0.022*"sit" + 0.022*"lovely"
8	0.200*"also" + 0.200*"have" + 0.200*"i" + 0.105*"it" + 0.105*"very" + 0.105*"quickly" + 0.010*"is" + 0.010*"my" + 0.010*"she" + 0.010*"sit"
9	0.247*"it" + 0.129*"my" + 0.129*"sit" + 0.129*"likes" + 0.129*"quickly" + 0.129*"learns" + 0.012*"is" + 0.012*"i" + 0.012*"very" + 0.012*"have"

Example Fairytale
'The golden bird'

- The bipartite graph created by the text of the fairy tale included 206 sentences and 884 words.
- The histogram of the length of the sentences in the text is presented in Fig.



- Manual inspection verified that meaningful impressions were extracted.

Conclusions

- ✓ A bag of words approach to modeling a semantic memory with APNN.
- ✓ APNN builds fast connections between APN neurons.
- ✓ It was simulated using a novel bipartite graph topology with modified APN neurons.
- ✓ It successfully extracts impressions associated with a given word.
- ✓ Two scenarios were developed and analyzed: tf and tf-idf:
    - The first one presents more direct associations.
    - The second one extracts more meaningful associations, but at the cost of returning more noise.
- ✓ The method was tested on documents with over 4500 sequences.
- ✓ The approach could be easily adapted for highlighting concepts highly associated with a given context.
- ✓ Future research will assess APNN in extracting patterns from continuous data and compare it to biclustering approaches.
- ✓ We plan to provide a toolbox with flexible connections of the neurons that would allow more complex analyses using APNN networks.
- ✓ It needs to be verified if the addition of a hidden layer to APNN related to n-grams could improve the efficiency of information retrieval.

Questions or Remarks?

Patryk Orzechowski
patryk.orzechowski@gmail.com

Adrian Horzyk
horzyk@agh.edu.pl

Janusz A. Starzyk
starzykj@ohio.edu

Jason H. Moore
jhmoore@upenn.edu

1. **A. Horzyk**, **J. A. Starzyk, J. Graham,** ***Integration of Semantic and Episodic Memories***, IEEE Transactions on Neural Networks and Learning Systems, Vol. 28, Issue 12, Dec. 2017, pp. 3084 - 3095, 2017, DOI: 10.1109/TNNLS.2017.2728203.
2. **A. Horzyk, J.A. Starzyk**, ***Fast Neural Network Adaptation with Associative Pulsing Neurons***, IEEE Xplore, In: 2017 IEEE Symposium Series on Computational Intelligence, pp. 339 -346, 2017, DOI: 10.1109/SSCI.2017.8285369.
3. **A. Horzyk**, ***Deep Associative Semantic Neural Graphs for Knowledge Representation and Fast Data Exploration***, Proc. of KEOD 2017, SCITEPRESS Digital Library, pp. 67 - 79, 2017, DOI: 10.13140/RG.2.2.30881.92005.
4. **A. Horzyk**, ***Neurons Can Sort Data Efficiently***, Proc. of ICAISC 2017, Springer-Verlag, LNAI, 2017, pp. 64 - 74, ICAISC BEST PAPER AWARD 2017 sponsored by Springer.
5. **Horzyk, A.**, ***How Does Generalization and Creativity Come into Being in Neural Associative Systems and How Does It Form Human-Like Knowledge?***, Elsevier, Neurocomputing, Vol. 144, 2014, pp. 238 - 257, DOI: 10.1016/j.neucom.2014.04.046.
6. **A. Horzyk**, ***Innovative Types and Abilities of Neural Networks Based on Associative Mechanisms and a New Associative Model of Neurons*** - Invited talk at ICAISC 2015, Springer-Verlag, LNAI 9119, 2015, pp. 26 - 38, DOI 10.1007/978-3-319-19324-3_3.
7. **R. Řehůřek and P. Sojka,** ***Software Framework for Topic Modelling with Large Corpora***, in Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. Valletta, Malta: ELRA, May 2010, pp. 45–50.
8. **P. Orzechowski and K. Boryczko**, ***Hybrid biclustering algorithms for data mining,*** in Applications of Evolutionary Computation, G. Squillero and P. Burelli, Eds. Cham: Springer International Publishing, 2016, pp. 156–168.
9. **P. Orzechowski and K. Boryczko,** ***Propagation-based biclustering algorithm for extracting inclusion-maximal motifs,*** Computing & Informatics, vol. 35, no. 2, 2016.