Retrieving Impressions from Semantic Memory Modeled with Associative Pulsing Neural Networks

Patryk Orzechowski*
Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104, USA
patryk.orzechowski@gmail.com

Adrian Horzyk
Department of Biocybernetics and Biomedical Engineering
AGH University of Science and Technology in Krakow
30-059 Krakow, Poland
horzyk@agh.edu.pl

Janusz A. Starzyk
School of EECS
University of Information Technology and Management in Rzeszow
and School of EECS, Ohio University
Rzeszow, Poland, and Athens, OH 45701.J., USA
starzykj@ohio.edu

Jason H. Moore
Institute for Biomedical Informatics
University of Pennsylvania
Philadelphia, PA 19104 USA
jhmoore@upenn.edu

***Abstract*—Associative Pulsing Neuron (APN) is a recently proposed model of neurons which incorporates multiple functional aspects of biologically-inspired Spiking Neuron (SN) while maintaining the simplicity of the simulation model. The APN model also enriches the functionality of SN models with conditional plasticity rules which let APNs to intelligently connect and represent associative relations that can be retrieved from raw training data or deduced from activities of presynaptic neurons. Associative Pulsing Neural Networks (APNNs) built with APN neurons use these plasticity rules to self-organize their network structures, are computationally efficient and provide easy interpretable and intuitive results. In this paper, we model the human-like activity of impression retrieval from a semantic memory using a self-organizing APNN structure spanned on a graph built from a bag of words (BOW), a popular data structure typically used for information retrieval. We illustrate that generation of associations in the network and present suitability of the proposed model to retrieve impressions.**

***Index Terms*—spiking neurons, bag of words, information retrieval, topic modeling, text mining, natural language processing**

I. Introduction

Computational neuroscience analyzes how human brain provides solutions to non-algorithmic problems. By mimicking the properties of biological neurons science tries to reveal the secrets of brain activities in order to use them in artificial neural, cognitive systems. Human perception reacts to the external environment by creating associations, which are fundamental elements of human cognition and thinking processes. Creating association is usually the result of a particular impression developed after stimulating receptors of senses. Human observes and learns how to distinguish colors, flavors, shapes, textures, sounds etc. The previous experience with similar objects are later projected and used as the base for creating particular impressions. For example, if human previously tasted apples, some of the first thoughts that may be recalled are that the apples are either green, yellow, or red, oval, sweet or sour, tasty, juicy etc. Those impressions are created because the human mind perceives multiple features in a similar context together with the 'apples'. Millions of similar associations are created during our lifetime by perceiving and confronting different concepts, for example: "Red apple is sweet and juicy.", "Lemons are yellow and sour.", "Green apples are sour." Many experiences form associations between the names of the object, what is sweet, juicy or sour in our minds, so after some time, we can also recognize objects on the basis of previously associated features, not necessarily by having the object in front of us.

Retrieving impressions associated with a given term usually involves a set of activities that take place in a human body and a brain. First, various receptors of the body senses react to external stimulus and stimulate connected neurons. This launches further neurotransmissions to other connected neurons through synapses which adapt to similar repeated time-spread combinations of input stimuli. As a result, we achieve stimulations in different subnetworks of neurons which pulse at various frequencies depending on the initial external stimulation. Neurons can be connected to several external devices (e.g. muscles in a human body) using effectors that transform neuronal pulses into the outputs interpretable by the external output devices. The reaction could be not only physical (e.g. body movement) but also mental (e.g. a thought). The activities of neurons can also be reinforced, especially when they repeat several times. Biological neurons have the ability to gradually develop their networks of connections as a reaction to various time-spread combinations of input stimuli which affect them. The conditional plastic adaptability of biological neural networks is a fundamental mechanism that allows them to adapt and learn various surroundings and satisfy the needs of human bodies and their living styles.

*Patryk Orzechowski is affiliated also with Dept. of Automatics, AGH University of Science and Technology, Krakow, Poland.

978-1-5386-9276-9/18/$31.00 ©2018 IEEE

Hence, plastic properties of neurons are a very important part of neuronal adaptation mechanisms. Therefore, the synaptic plasticity and neuronal body plasticity should be used when creating artificial neural networks. This is more effective than only the changes in the synaptic weight parameters provided by the majority of spiking and artificial neural networks training routines.

There are three major generations of neurons used in artificial neural networks [1]. The first one is based on McCulloch-Pitts neurons, which are basic binary switches with input and output signals equal either to zero or one. Although this model is fairly simple, it has been successfully adapted to create multilayer perceptrons, ADALINE and Hopfield networks [2]. As a result, the second generation of neurons uses continuous activation functions. Neither threshold nor step functions are used to determine the output of a neuron. Some of the most popular activation functions include sigmoid, tangent hyperbolic [3], rectified linear units (ReLU) [4], or various radial functions, e.g. a Gaussian function [2]. These type of artificial neurons are commonly used to construct various kinds of artificial neural networks. They became very popular with the development of deep learning networks and techniques, and have proven to be extremely successful in solving different tasks [2], [5]–[7]. The significant advantage of the second generation neurons over the predecessors is in the reduction of the number of required neurons, as well as the ability of these neurons to approximate analog signals.

Despite the great popularity of the second generation of neurons, it has been observed that the behavior of these neurons poorly corresponds to the mechanisms of biological neurons which primarily communicate through a series of asynchronous spikes instead of continuous synchronous signals [8], [9]. This fostered the development of a biologically-inspired third generation of neurons called spiking neurons (SN) [10]–[12]. Although spiking neural networks (SNNs) are believed to have greater potential than their predecessors, they have not yet been found to outperform the networks built from the second generation of neurons. In addition, we still do not have efficient adaptation algorithms for creating their structures or training them for desired performance. Only recently effective techniques for training deep SNNs were proposed in [13], [14] and used in the form of classifiers [15], [16].

Although spiking neural networks have great potential, they are often infeasible due to their complexity, especially when larger datasets are considered. This justifies emergence of simulation-oriented approaches in modeling behaviors of such neural networks. One of the biologically-inspired simulation-based models of neuron activity is called Associative Pulsing Neuron (APN) [17]–[20]. The networks built with APNs are called Associative Pulsing Neural Networks (APNNs). APNNs are self-organizing, which means that during their construction and training they automatically create the required number of neurons and establish all necessary connections between them, adapting their synaptic weights. This makes them suitable for modeling semantic memory, as the created network topology fits the input data and stores the knowledge acquired during the construction and training process. The APNNs automatically create reinforced associations between the recognizable objects by using neurons synaptic plasticity.

In this paper, we present a novel method for training APNN which is dedicated to the task of impressions retrieval from the semantic memory. The strategy of creating connections extends a bag of words (BOW) model for the creation of the network topology. The network is expected to extract words that are associated with given terms based on the previously memorized documents during the training. The ability to create and recover word associations is the major advantage over co-clustering techniques [21]–[23], which attempt to find subsets of words appearing in the same documents.

II. Associative Pulsing Neural Networks

Associative Pulsing Neural Networks (APNN) [19] broaden the possibilities of Spiking Neural Networks (SNN) implementing plastic properties of biological neurons for automatic creation of network topologies from the existing relations between data in training data sets. The basic advantage of APNNs over SNNs is the reduction of the model complexity while maintaining an event-based approach for learning and modeling signal propagation within the network. The structure of APNN develops gradually and depends on a training dataset. Connections between neurons are created only when needed and are usually very sparse allowing the specialization of APN neurons. This enhances the computational efficiency of the model for representing the input data in the simulation and simplifies interpretation of the obtained results.

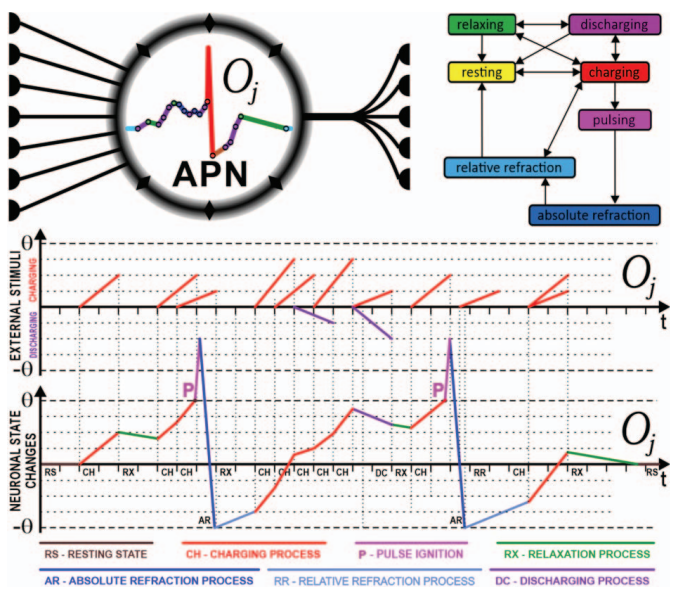

Fig. 1. Dynamic states and internal processes of APN neurons obtained for various external stimuli.

Associative pulsing neuron (APN) [17] can be in one of the following states: charging, discharging, relaxing, pulsing, absolute or relative refracting, and resting. The APNN network

used for classification also includes receptors, responsible for the transformation of raw input data into their internal representation by charging connected neurons. Receptors are linked to sensory input fields (SIF) representing data attributes. The APNN network also contains effectors, which deliver output results on the basis of the number of pulses generated by APN neurons for given input stimulations. In this way, effectors can transform pulses into the analog output signals. They can be used for regression tasks determining the strength of associations of objects represented by these effectors to the objects or features representing an input context.

Fig. 1 shows how the internal processes of APN are induced by external excitation (charging) or inhibition (discharging) stimuli. APN neurons quickly integrate all upcoming input stimuli taking into account their strength, type and time of occurrence. All input stimuli of every neuron are separately integrated and represented by the internal process queue (IPQ) described in [17]. The internal processes are changing the states of APNs. The end-time of each process is checked by the global event queue (GEQ). Thanks to the time-ordering of all internal processes in APN neurons, each neuron can have at most a single process active, so the GEQ watches over a single event of each APN neuron at most. The GEQ orders events over time, so the neurons are always updated at right moments when their internal processes finish. Thanks to the fact that each internal process takes time, there is no problem with the execution of asynchronously parallel processes.

III. Methods

In the presented approach, we used APNN networks calculate weights of connections representing modeled associations between words and documents. Many sentences are constructed from these words, and the problem may be computationally demanding. The APN neurons work in time, so the stronger associations make a stronger influence on post-synaptic neurons forcing them to pulse more frequently. The achieved neuronal frequencies describe the strength of associations of words with documents and indirectly point out other associated words with these documents which were not given in the input context. We used APN neurons and APNN networks because they can be created and configured quickly in comparison to various spiking neuron networks. Moreover, APNN can model different relations between objects represented by APN neurons. Therefore, APN neurons can represent objects in the context of their occurrence, what supply us with semantic relations of these objects to associated objects, defining them, being in the sequence and many other. Since APN neurons aggregate representation of many occurrences of the same object (in this work: words), documents represented by the other neurons connected to the neurons representing words are also indirectly connected, semantically binded, and associated. This allowed creating various semantic memories as was described in [18].

Training of APNNs is very fast. It relies on creating a set of meaningful connections between the neurons representing associations which reproduce relations between objects or their features. The connections are appropriately weighted taking into account the frequency of the modeled relations. Rare features better identify objects, so the connection weights representing such associations should be greater. On the other hand, frequent features, which do not identify objects well, need to be represented by weaker weights [24].

For the task of information retrieval, we modified the topology of the APNN used originally for classification [19]. For detecting impressions, we have adapted a concept of a bag of words (BOW). This popular model in information retrieval allows representing directed the relations between terms that appear in a specific document and the document itself, depending on the number of occurrences of a specific term in a particular document and over all documents.

A. Bag of words

The term bag of words was introduced in the middle 1950s [25]. Since then it has become a classical text representation technique and found its application not only in linguistic tasks [26], [27] but also in text classification, and image recognition [28], [29].

The basic idea of a bag of words is quite simple: for each document we count the number of occurrences of a given word w_i and represent a document d_j as a set of pairs, associating a given word with the number of times it appears in a document. Thus, a bag of words b for a document d_j that comprises of n_i words is equal to (1):

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

where n_i is the number of words in document d_i:

$$d_j = (w_{1j}, w_{2j}, ..., w_{n_{ij}}) \tag{2}$$

and

$$\delta(x, y) = \begin{cases} 1 & if\, x = y \\ 0 & if\, x \neq y \end{cases} \tag{3}$$

The bag of words for a collection of documents is defined as follows (4):

$$b = (b_1, b_2, ...b_N) \tag{4}$$

where N is a number of documents.

The bag of words model allows representing a collection of the documents in the form of a sparse matrix with values equal to a number of occurrences of a given term in a specific document. Although some form of information about each document is represented, the model is not able to describe the relevance of each term. For example, if a word occurs in every single document, it does not have much information value and could be discarded from the collection. Similarly, the relevance of a term should be higher if a term frequently occurs within a document.

One of the techniques of ranking documents according to the relevance to the specific terms is called term frequency (tf). The simplest version of tf counts the number of occurrences

of a term in a document and averages by the total number of terms in a document (5). Thus, associations between a term and a document are stronger if the term appears more frequent within the document.

$$tf(w_i, d_j) = \frac{\#w_i}{length(d_j)} \tag{5}$$

A more sophisticated technique of ranking the relevance of documents is called Inverse document frequency (idf). The adjustment of the importance of the documents is based on the number of documents that contain a given word in the whole collection of documents. If a given word is less common in the collection of the documents, it is less informative. One of the most common weighting schemes for idf is

$$idf(w_i, d_j) = \log \frac{N}{\#\{d_j : w_i \in d_j\}} + 1 \tag{6}$$

where N is a number of documents. The denominator needs to be higher than zero (i.e a word needs to appear at least once in document corpus). The value $+1$ that is added doesn't completely ignore the words that are present in all documents.

The combination of tf and idf (7) creates a scheme called term frequency and the inverse document frequency (tf-idf) [30], which is a popular scheme in information retrieval.

$$tfidf(w_i, d_j) = tf(w_i, d_j) \cdot idf(w_i, d_j) \tag{7}$$

Combination of term frequency and inverse document frequency constitute a scheme that adjusts the weights of the documents by incorporating both the frequency of a given term in the document and overall popularity of the term within the collection.

B. Model of APNN-BOW

APNN network was spanned on top of the bag of words BOW created for input text. As a result, a bipartite directed graph was created with nodes that are either documents or words. Each node was represented as a separate neuron. If a word was present in a given document, a weighted edge in the network was created between the word and the document. We attached receptors to each of the word neurons that allows the neuron to be charged over time. Activation of a neuron sent the signal through all outgoing edges to its neighbors and increased their potential.

The original implementation by Horzyk was ported from C# into Python by the first co-author of this paper in order to maintain scalability for the larger networks. Graph-tool[1] was used for modeling and storing the structure of the network. We used Pandas[2] for data manipulation, Natural Language Toolkit (NLTK)[3] for extracting sentences from texts, and Scikit-learn[4] for tokenizing the sequences.

[1]https://graph-tool.skewed.de/
[2]https://pandas.pydata.org/
[3]https://www.nltk.org/
[4]https://scikit-learn.org/

Some important modifications have also been made to the previous implementation. First, neuron attributes have been stored in dictionaries, instead of Attribute-Value B-Trees (AVB-trees) [24]. Secondly, internal neuron processes queue stores only external stimuli events which are current. This means that relaxation or refraction are not stored at all in the list, but activated if there is no process on IPQ. Furthermore, the logic of the neuron activity has been shifted towards the neuron controller, which controls the launching of the specific functions, and global coordinator, which launches specific activities when they are due.

C. Relation to other approaches

In the literature, some other approaches have been proposed for the task of information retrieval. In a popular Latent Semantic Analysis (LSA), which is also called Latent Semantic Indexing (LSI), the representation of the data in a bag of words format is further processed using Singular Value Decomposition (SVD) in order to find words contribution to specific topics [31]. Each word may contribute in either a positive or negative manner for a given topic. The topic modeling has also been one of the aims of Latent Dirichlet Allocation (LDA) [32], which takes a different approach by using probabilistic generative modeling. The result of LDA is the more interpretable positive contribution of a set of words to modeled topics. These aforementioned approaches do not yield possibility to point out the nearest word associations to the given stimuli, instead only the relationship between a given sentence presented as input and a modeled topic could be presented.

TABLE I
PARAMETERS OF THE APNN USED IN THE SIMULATIONS.

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

IV. RESULTS

In this paper, we try to assess if APNN-BOW could successfully retrieve impressions that are associated with a specific set of words. We compare two strategies of setting weights for the network. The first one uses *CountVectorizer* and sets the weights from documents to words according to (5). The second one uses *TfIdfVectorizer* and sets weights according to (7). In both scenarios, a directed bipartite graph is created with two types of edges: $word \rightarrow document$ and $document \rightarrow word$. The weights of the first type of edges in both scenarios are set to 1 in order to reflect the relation of the word appearing in a document. The weights of the opposite edges are set either to tf or tf-idf value. That means that if a neuron associated with a given term is activated, the neurons of documents, in which the term is present, are activated. Activation of a neuron

associated with a document leads only to sending a signal to neurons with terms appearing in the same document. The values of parameters with which the simulation was run are presented in Fig. I.

A. Toy example

In order to present the creation and behavior of the network, we have adapted a toy example with 15 sentences from [18] presented in Fig. 2.

```
1. I have a monkey.
2. My monkey is very smart.
3. It is very lovely.
4. It likes to sit on my head.
5. It can jump very quickly.
6. It is also very clever.
7. It learns quickly.
8. My monkey is lovely.
9. I also have a small dog.
10. I have a sister.
11. My sister is lovely.
12. She is very lovely.
13. She likes to sit in the library and to read.
14. She quickly learns languages.
15. I also have a brother.
```

Fig. 2. Toy example with 15 different texts used for modeling connections in the APNN network.

The network of the connections generated for the sentences shown in Fig. 2 is presented in Fig. 3.

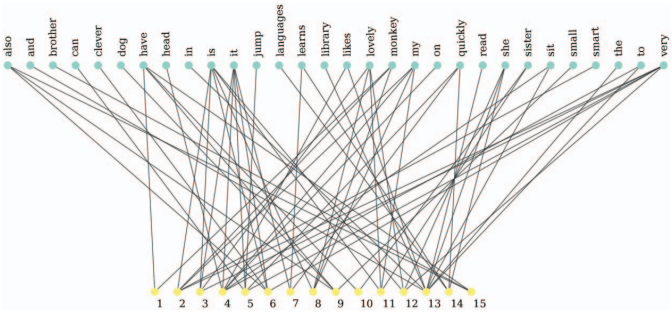

Fig. 3. The topology of the APNN network created using a bag of words built with TfIdfVectorizer for the toy example. The documents are painted in yellow and words in blue. One letter words were omitted.

We have simulated the response of APNN to toy example in order to observe the behavior of the network. In our tests we have stimulated the network using different words or phrases, e.g. 'monkey', 'monkey is', 'she is' etc. and observed the responses. The neurons that spiked for the first scenario using term frequency weights are presented in Table II. The values in brackets correspond to the number of spikes observed during the simulation.

The responses of the network in the second scenario, in which the weights were set according to tf-idf score are presented in Table III.

Setting tf-idf values as a score allow retrieving from the network more and much deeper associations compared to the scenario of using tf weights only. For example, for term 'monkey', using tf-idf weights allows recovering such impressions as *monkey learns*, *smart monkey*, or association between *monkey* and *it*, which are not retrieved using tf scenario only. On the other hand, retrieving more impression is associated with more noisy response, e.g. monkey becomes associated with *she* and *sister* (as both appear in contexts of being lovely).

TABLE II
RESPONSES OF APNN FOR STIMULATION OF DIFFERENT WORDS FOR TOY EXAMPLE IN TF SCENARIO (COUNTVECTORIZER). WE HAVE INCLUDED NEURONS THAT SPIKED AT LEAST ONCE DURING 100 SIMULATION UNITS.

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)	-

TABLE III
RESPONSES OF APNN FOR STIMULATION OF DIFFERENT WORDS FOR TOY EXAMPLE USING TF-IDF SCORE (TFIDFVECTORIZER). WE HAVE INCLUDED NEURONS THAT SPIKED AT LEAST TWICE DURING 100 SIMULATION UNITS.

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) monkey(35)	lovely(15), very(15), my(14), smart(9), it(8), have(5), sister(5), she(5), also(5), quickly(5), learns(5), clever(4), to(2)
she(35) is(35)	lovely(15), very(15), my(14), monkey(8), it(8), quickly(7), learns(7), to(6), languages(5), clever(5), likes(4), sit(4), sister(4), smart(4), have(3), also(3)
she(35) sister(35)	lovely(9), is(8), very(8), my(8), it(8), quickly(8), learns(8), monkey(6), 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) brother(35)	have(9), dog(4), small(4), clever(4), very(2), it(2), 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)

The performance of the method could be observed in two examples. One with simultaneously presenting words 'sit' and 'library'. Tf scenario captures only one neuron (*to sit library*), while tf-idf was able to extract words associated with the right context (*she likes to sit in the library*), as well as a bit of noise (head, on). For the other scenario with the unique term, 'jump', none of the other neurons spiked for tf example, while tf-idf extracted words appearing in the same context *can quickly jump* and associated them with a monkey. It also added multiple impressions of the monkey, such as that it is *my*, as well as *lovely*, and that *it learns quickly*.

An exemplary response of the neurons over time for stimulation of neuron 'lovely' for tf-idf scenario is presented in Fig. 4. It may be observed, that the most frequent terms (considered "impressions"), which are associated with words 'lovely' are in first place 'is' (*is lovely*), 'very' (*very lovely*), and 'my' (*my lovely*), then in the second place 'it', 'monkey', 'sister' and 'she' (all the above appear in the context of training data).

Fig. 4. The response of APNN learned on toy example and tested using 'lovely' neurons.

Some of the remaining neurons also become activated, but their spiking frequency is less frequent.

We have also extracted ten topics using Gensim software [33] with LSA/LSI (see Table IV) and LDA (see Table V). In both aforementioned approaches, impressions could be modeled by referring given stimuli to the topics presented in Tables IV and V and retrieving topic probability distribution for a given stimulus. An impression retrieval might be modeled here as querying the topics and finding which topic has the highest likelihood of being associated with a given stimulus. For example, the closest answer for the stimulus of the word 'monkey' will be the topic numbered 6 for LSA/LSI and 3 for LDA. It might be noticed that those approaches are conceptually distinct from BoW-APNN presented herein. Instead of modeling topics present in the collection of the documents and associating given terms with one of the topics, BoW-APNN shows associations between words appearing in the same text and extracts association of other words to the one presented on the input.

B. Fairytale 'The golden bird' example

For a better verification of our approach, we have taken a more complex example and trained APNN on the example of a fairy tale by Grimms' brothers called 'The golden bird'. The bipartite graph created by the text covered 206 sentences and 884 words. The histogram of the length of the sentences in the fairy tale is presented in Fig. 5.

We have run tf-idf simulation only, as it produced more advanced insights for the toy example. We run the simulation for the similar time and observed which neurons spiked. The list of the most frequent words associated with

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"

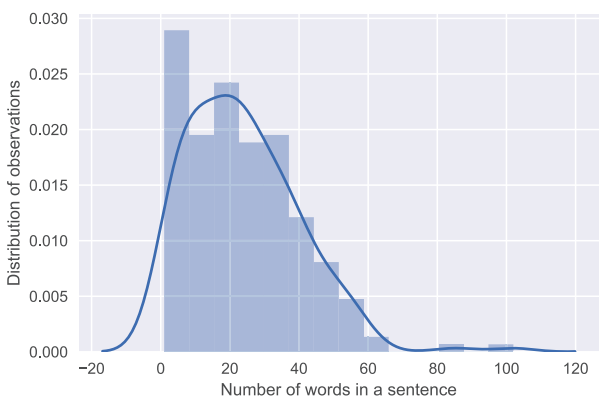

Fig. 5. The length distribution of the fairy tale 'The golden bird' sentences.

the phrase bird are presented below (only those impressions are presented, for which at least 15 spikes of word neurons were observed): **bird(36)**, the(16), and(16), to(16), he(16), golden(15), it(15), was(15), him(15), but(15), his(15), you(15), princess(15), that(15), horse(15), of(15), in(15), then(15), fox(15), said(15), as(15), will(15), at(15), went(15), all(15), king(15), so(15), away(15), not(15), for(15), came(15), on(15), when(15), they(15), have(15), son(15), had(15), this(15), is(15), would(15), into(15), very(15), no(15), if(15), be(15), with(15), give(15), took(15), my(15), out(15), cow(15), one(15), by(15), did(15), good(15), off(15), up(15), your(15), go(15), me(15), hans(15), do(15), man(15), what(15), where(15), great(15), down(15), take(15), upon(15), could(15), last(15), more(15), pig(15), again(15), thought(15), set(15), can(15), her(15), himself(15), side(15), must(15), there(15), find(15), while(15), their(15), silver(15), began(15), gave(15), stone(15), about(15), were(15), well(15), till(15), how(15), matter(15), only(15), now(15), are(15), hand(15), put(15), get(15), them(15), such(15), head(15), goose(15), who(15), fat(15), fine(15).

We looked back to the original text, validated the results, and found out that 'bird' appears in 21 sentences where 'the' is also always present. This justifies why this impression was detected as the most related to the word 'bird'. For comparison, the term 'bird' appears together with 'and' in 20 sentences, with 'to' in 16 sentences, with 'he' in 21 sentences, with 'it' in 13 sentences and with 'was' in 11 sentences. The word 'golden' appears in context of the 'bird only 10 times, 'princess' - 7 times, and 'horse' - 8 times. A phrase *the golden bird* appears only in 7 sentences from the corpus.

V. CONCLUSIONS AND DISCUSSION

This paper presents the proof of concept of modeling the semantic memory using APNN spanned on a bag of words model using two concepts: tf and tf-idf. The behavior of APNN was simulated on a novel topology. We have presented that APNN may successfully modeling the recognition of terms that are associated with the given word, based on simplistic connections of APN neurons. This suggests that the approach could be further adapted for modeling memory and impression retrieval. We have compared two different scenarios of weighting edges in APNN. First used term frequency (tf), the second one-term frequency-inverse document frequency (tf-idf). Both techniques have pros and cons. For the toy example, we have observed that the first one presents more direct associations, while the second one extracts more meaningful associations, but at the cost of returning more noise. Determining the threshold for filtering out the noise requires further experiments.

The behavior of the APNN network for longer documents requires further analysis. The network was capable of retrieving word associations for a short document. Up to date, we have successfully managed to apply our method to the text with over 4500 of sequences in order to prove that it is working correctly. Also, the bag-of-words model has some caveats. First, the information about the context is not used directly. The context needs to be provided by using a few words instead of a single one. Second, the order of the words within this approach does not matter. This may be a drawback if we want to extract phrases, instead of single words only. We believe it may be beneficial to add another type of contextual information, e.g. add n-grams, or build connections based on the grammar rules.

In our future research, we would like to assess APNN suitability for extracting patterns from continuous data and compare it to some popular biclustering approaches [34]–[37]. 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.

VI. Acknowledgments

This work was supported by National Institutes of Health (USA) grant AI116794, AGH 11.11.120.612, and the grant from the National Science Centre DEC-2016/21/B/ST7/02220.

References

[1] S. Kumar, *Neural networks: a classroom approach*. Tata McGraw-Hill Education, 2004.

[2] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[3] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[4] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[6] B. K. Beaulieu-Jones, P. Orzechowski, and J. H. Moore, "Mapping patient trajectories using longitudinal extraction and deep learning in the mimic-iii critical care database." *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, vol. 23, pp. 123–132, 2018.

[7] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman *et al.*, "Opportunities and obstacles for deep learning in biology and medicine," *bioRxiv*, p. 142760, 2018.

[8] L. Abbott, B. DePasquale, and R.-M. Memmesheimer, "Building functional networks of spiking model neurons," *Nature neuroscience*, vol. 19, no. 3, p. 350, 2016.

[9] J. W. Kalat, *Biological psychology*. Nelson Education, 2015.

[10] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[11] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[12] E. M. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on neural networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[13] P. O'Connor and M. Welling, "Deep spiking networks," *arXiv preprint arXiv:1602.08323*, 2016.

[14] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in neuroscience*, vol. 10, p. 508, 2016.

[15] M. Schmuker, T. Pfeil, and M. P. Nawrot, "A neuromorphic network for generic multivariate data classification," *Proceedings of the National Academy of Sciences*, vol. 111, no. 6, pp. 2081–2086, 2014.

[16] E. Stromatias, M. Soto, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven classifier for spiking neural networks fed with synthetic or dynamic vision sensor data," *Frontiers in neuroscience*, vol. 11, p. 350, 2017.

[17] A. Horzyk and J. A. Starzyk, "Fast neural network adaptation with associative pulsing neurons," in *Computational Intelligence (SSCI), 2017 IEEE Symposium Series on*. IEEE, 2017, pp. 1–8.

[18] A. Horzyk, J. A. Starzyk, and J. Graham, "Integration of semantic and episodic memories," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 12, pp. 3084–3095, 2017.

[19] A. Horzyk and J. A. Starzyk, "Multi-class and multi-label classification using associative pulsing neural networks," in *IEEE World Congress on Computational Intelligence*. IEEE, 2018.

[20] A. Horzyk, "Neurons can sort data efficiently," in *International Conference on Artificial Intelligence and Soft Computing*. Springer, 2017, pp. 64–74.

[21] P. Orzechowski and K. Boryczko, "Text mining with hybrid biclustering algorithms," in *International Conference on Artificial Intelligence and Soft Computing*. Springer, 2016, pp. 102–113.

[22] G. Bisson and F. Hussain, "Chi-sim: A new similarity measure for the co-clustering task," in *Machine Learning and Applications, 2008. ICMLA'08. Seventh International Conference on*. IEEE, 2008, pp. 211–217.

[23] P. Orzechowski and K. Boryczko, "Propagation-based biclustering algorithm for extracting inclusion-maximal motifs," *Computing & Informatics*, vol. 35, no. 2, 2016.

[24] A. Horzyk, "Deep associative semantic neural graphs for knowledge representation and fast data exploration," in *International Conference on Knowledge Engineering and Ontology Development*. SCITEPRESS Digital Library, 2017, pp. 67–79.

[25] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[26] S. Scott and S. Matwin, "Feature engineering for text classification," in *ICML*, vol. 99, 1999, pp. 379–388.

[27] H. M. Wallach, "Topic modeling: beyond bag-of-words," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 977–984.

[28] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo, "Evaluating bag-of-visual-words representations in scene classification," in *Proceedings of the international workshop on Workshop on multimedia information retrieval*. ACM, 2007, pp. 197–206.

[29] L. Wu, S. C. Hoi, and N. Yu, "Semantics-preserving bag-of-words models and applications," *IEEE Transactions on Image Processing*, vol. 19, no. 7, pp. 1908–1920, 2010.

[30] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[31] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse processes*, vol. 25, no. 2-3, pp. 259–284, 1998.

[32] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[33] 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, http://is.muni.cz/publication/884893/en.

[34] P. Orzechowski, M. Sipper, X. Huang, and J. H. Moore, "Ebic: an evolutionary-based parallel biclustering algorithm for pattern discovery," *Bioinformatics*, p. bty401, 2018. [Online]. Available: http://dx.doi.org/10.1093/bioinformatics/bty401

[35] P. Orzechowski and J. H. Moore, "Ebic: an open source software for high-dimensional and big data biclustering analyses," *arXiv preprint arXiv:1807.09932*, 2018.

[36] P. Orzechowski, A. Paszczyk, X. Huang, and J. H. Moore, "runibic: a bioconductor package for parallel row-based biclustering of gene expression data," *Bioinformatics*, p. bty512, 2018. [Online]. Available: http://dx.doi.org/10.1093/bioinformatics/bty512

[37] 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.