Neural Networks 178 (2024) 106494

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Full Length Article

Leveraging spiking neural networks for topic modeling

Marcin Białas [a,*], Marcin Michał Mirończuk [a], Jacek Mańdziuk [b]

[a] National Information Processing Institute, al. Niepodległości 188b, 00-608, Warsaw, Poland
[b] Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland

ARTICLE INFO

Keywords:
Spiking neural network
STDP
Topic modeling
Spiking topic model
Unsupervised learning

ABSTRACT

This article investigates the application of spiking neural networks (SNNs) to the problem of topic modeling (TM): the identification of significant groups of words that represent human-understandable topics in large sets of documents. Our research is based on the hypothesis that an SNN that implements the Hebbian learning paradigm is capable of becoming specialized in the detection of statistically significant word patterns in the presence of adequately tailored sequential input. To support this hypothesis, we propose a novel spiking topic model (STM) that transforms text into a sequence of spikes and uses that sequence to train single-layer SNNs. In STM, each SNN neuron represents one topic, and each of the neuron's weights corresponds to one word. STM synaptic connections are modified according to spike-timing-dependent plasticity; after training, the neurons' strongest weights are interpreted as the words that represent topics. We compare the performance of STM with four other TM methods Latent Dirichlet Allocation (LDA), Biterm Topic Model (BTM), Embedding Topic Model (ETM) and BERTopic on three datasets: *20Newsgroups*, *BBC news*, and *AG news*. The results demonstrate that STM can discover high-quality topics and successfully compete with comparative classical methods. This sheds new light on the possibility of the adaptation of SNN models in unsupervised natural language processing.

1. Introduction

Spiking neural networks (SNNs) are artificial neural networks (ANNs) inspired by the structure and functioning of biological neurons. Unlike traditional ANNs, which operate on continuous values, SNNs operate on discrete time scales: neurons communicate by sending spike trains or pulses of binary activity through synapses. If a neuron participates in the activation of another neuron repeatedly, the synaptic connection between them strengthens. This learning paradigm was initially proposed by Donald Hebb (Hebb, 1949).

One of the biologically plausible forms of Hebbian learning (HL) is spike-timing-dependent plasticity (STDP) (Song et al., 2000). According to the STDP rule, if the presynaptic neuron fires shortly before the postsynaptic one, the synapse between them strengthens; in contrast, it weakens if the presynaptic neuron fires after the postsynaptic one.

Suppose we can represent a document as a sequence of spikes, each representing a single word. Neurons trained according to the STDP rule will become responsive only to groups of words that co-occur frequently. Also suppose that the neurons compete with each other. In that case, each neuron will become active only in the presence of the words that occur in a particular context, and the contexts will differ between neurons. From the linguistic perspective, *words that occur in the same contexts tend to have similar meanings* (Harris, 1954; Karlgren & Sahlgren, 2001), so detecting the co-occurrence of words can be used to infer the topics being discussed in text (Abdelrazek et al., 2023; Chauhan & Shah, 2022; Jelodar et al., 2019a; Vayansky & Kumar, 2020a; Zhai, 2017). In natural language processing (NLP), such an inference is called topic modeling (TM) (Chen & Liu, 2017). TM algorithms extract essential groups of words that represent different human-interpretable topics from documents (Kherwa & Bansal, 2018). Subsequently, documents with similar topics can be grouped into thematic clusters. This approach saves resources by avoiding the time-consuming and costly manual labeling of large datasets (Aggarwal & Zhai, 2012; Kherwa & Bansal, 2018; Manning et al., 2008).

However, this seemingly straightforward approach to TM presents a significant challenge due to the vast vocabulary inherent in textual data. This data is typically represented in a sparse high-dimensional vector space and extracting valuable information from such data is a challenging task. Despite these complexities, addressing TM challenges is worthwhile. Improved methods and novel approaches can unlock more profound insights from text data. By transforming text into a structured format using TM, we unlock a deeper understanding of the information landscape. This structured format allows data mining tools to explore data or use machine learning techniques. Also, this empowers us with a range of applications, including understanding public opinion through social media analysis (An et al., 2023; Karas

[*] Corresponding author.
E-mail addresses: marcin.bialas@opi.org.pl (M. Białas), marcin.mironczuk@opi.org.pl (M.M. Mirończuk), jacek.mandziuk@pw.edu.pl (J. Mańdziuk).

https://doi.org/10.1016/j.neunet.2024.106494
Received 17 January 2024; Received in revised form 6 May 2024; Accepted 25 June 2024
Available online 27 June 2024

et al., 2022; Wang et al., 2023) or user feedback (Asnawi et al., 2023), improving travel recommender systems (Alenezi & Hirtle, 2022), powering personalized content recommendations (Kumar & Hanji, 2024; Yang et al., 2021), or exploring vast scientific literature (Facchinetti et al., 2022; Sebastian et al., 2017)

To address the lack of SNNs-based solutions in TM and explore the potential of SNNs for this task, our research aims to answer the following question: *Can competing spiking neurons, trained according to the STDP learning rule, learn topics from the sequence of spikes effectively*? Based on the literature described above, we reason that a network trained according to the HL paradigm should, in principle, be capable of performing such a task in the presence of adequately tailored input. However, to the best of our knowledge, no works in the SNN field consider this observation. A substantial gap remains in SNN's application to NLP. Several articles that address this subject, further discussed in Section 2 of this article, focus on text classification and the adaption of solutions inspired by research on deep neural networks (DNNs). Our study focuses on a simple, biologically plausible network that delivers an intuitive solution to the TM task. Since our network contains one layer only and utilizes the STDP learning rule, it is resource-efficient and need not address the numerous issues that arise in deep learning approaches.

The main contributions of this work can be summarized as follows:

- By transforming text data into spike trains and using SNNs for TM, our research introduces a novel approach to text processing and the TM task. It also showcases the versatility of SNNs by demonstrating their ability to perform unsupervised text analyses and explore data.
- We propose a new TM method based on an SNN and sequential text transformation, named the Spiking Topic Model (STM). Additionally, the transformation enables a novel way of encoding text data, which might benefit neuromorphic hardware implementations.
- The STM achieves competitive performance with other non-spiking-based NLP models. This offers several potential benefits, including more effective uncovering of hidden themes, patterns, and relationships within the data, as well as better understanding and organization of document content.
- We evaluate STM on three real-world text collections (the *20Newsgroups*, *BBC news*, and *AG news* datasets) and compare the results with four other (non-SNN) TM methods.
- We have released the related source code and made it freely available to the research community.[1]

Section 2 of this article presents an overview of the related work. Section 3 offers a detailed presentation of the proposed method. Section 4 discusses the evaluation procedure with an experimental setup and experimental outcomes. Section 5 shows scalability of the methods. Section 6 concerns the learning procedure and parameter selection. Section 7 presents implementation of the STM in GPU. Section 8 offers concluding remarks.

2. The application of SNN in NLP

SNNs, which mimic biological neurons' communication process, hold the potential for improved efficiency in some computational tasks, particularly in event-based or energy-efficient applications (Roy et al., 2019). The literature presents numerous examples of the effective use of SNNs (Yamazaki et al., 2022); however, only a handful address their application to NLP. Those works are summarized below.

First we turn our attention to the text classification tasks inspired by DNNs (Diehl et al., 2016; Huang et al., 2023; Jiang et al., 2023; Lv et al., 2023). The principal motivation for adopting such an approach lies in attaining classification outcomes in an SNN that closely approximate those achieved by ANNs while concurrently mitigating electrical energy consumption. The presented works focus on the conversion of pretrained DNNs to SNNs and their subsequent use in text classification (Diehl et al., 2016; Huang et al., 2023; Jiang et al., 2023; Lv et al., 2023), adapting SNNs for training with gradient descent algorithms (Huang et al., 2023; Lv et al., 2023), and exploring the spiking text representation problem (Jiang et al., 2023; Lv et al., 2023).

Text classification is also the subject of the research of Maciąg et al. (2022), in which the authors evaluate the effectiveness of several SNN architectures in short text classification tasks, with an additional focus on the problem of text representation.

Another compelling approach to NLP is proposed by Zhu et al. (2023), who introduce SpikeGPT, a generative language model inspired by the recently proposed Receptance Weighted Key Value (RWKV) DNN architecture (Peng et al., 2023). SpikeGPT is a back-propagation-trained SNN that incorporates several mechanisms to enhance its robustness. Initial experiments indicate that SpikeGPT maintains competitive performance with nonspiking models in natural language generation and natural language understanding tasks on several benchmark datasets.

Another two notable studies address text representation. Wang et al. (2019) develop three models inspired by biological neurocoding to refine original dense word embeddings. The embeddings are evaluated in text classification and word similarity tasks. Białas et al. (2020) demonstrate how a biologically plausible single-layer SNN can be used as a text document encoder to achieve low-dimensional document representation. The resulting representation is evaluated in a text classification task.

In summary, text classification is a dominant research topic in the application of SNNs to NLP. Additionally, the impulsive representation of text, as well as the generation of natural language, are addressed by researchers. Moreover, most works adapt solutions that have proven effective in feed-forward networks to SNN models, mainly based on supervised learning. In contrast, the leveraging of SNNs for unsupervised NLP tasks like TM remains unexplored.

3. Method

The proposed spiking topic modeling method is presented schematically in Fig. 1. In the first stage, called *Prepossessing*, each text document is transformed into a sequence of tokens, in which each token corresponds to one prepossessed word from the document. In the second stage, called *Transforming tokens to spikes*, the sequences of tokens are converted into spikes according to the proposed novel transformation schema (Section 3.1). In the last stage, called *Training topic model*, the sequences of spikes are used to train the STM according to a variant of the STDP rule (Section 3.2). Section 3.3 explains how the network can be used as a topic model after training.

3.1. Transforming text to spikes

One common way of representing text documents is based on the concept of a feature space (Jurafsky & Martin, 2009; Tang et al., 2020), a multidimensional space in which each dimension corresponds to one word from a predefined dictionary, which contains all of the words of a dataset.

Assume the presence of a document corpus C described by a dictionary D that comprises M words x_1, x_2, $\dots x_M$. Each word $x_i \in D$ is represented by exactly one neuron from the input layer of the STM. The transformation of the text into spike trains occurs in two stages. First, each document is transformed into a sequence of words from the dictionary. Next, each document is read, starting from the first word of the sequence to the last one, and transformed into spike trains, which determine the activity levels of the input layer neurons. When a

[1] https://github.com/mioun/stm-ssn-topic-model.git

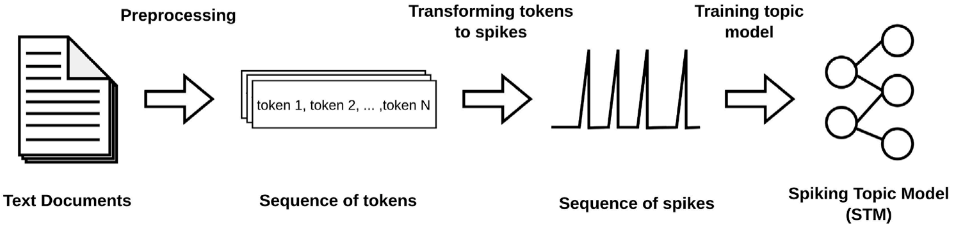
Preprocessing
Transforming tokens to spikes
Training topic model
token 1, token 2, ... ,token N
Text Documents
Sequence of tokens
Sequence of spikes
Spiking Topic Model (STM)

Fig. 1. An overview of the spiking topic modeling method.

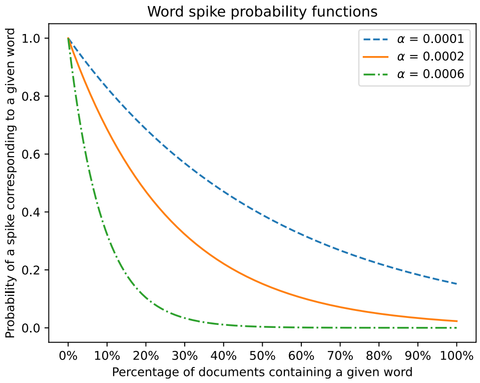

Fig. 2. A plot of the probability function of activating a neuron that represents a given word with respect to the percentage of documents in the corpus that contain that word. The function is plotted for three different values of α (cf. Eq. (1)).

particular word is read, the corresponding neuron may be activated and produce a spike. The neuron's activation is governed by the probability function $P(x_i)$:

$$P(x_i) = e^{-\alpha|D(x_i)|} \tag{1}$$

where $|D(x_i)|$ is the number of documents containing word x_i, and α is a small constant. If x_i occurs in a low number of documents, the probability of generating a spike for that word is close to 1—the maximum possible value. Fig. 2 presents the probability functions for three different values of α.

As the frequency of a word's occurrence increases, the probability decreases exponentially. The rate at which the function values decrease depends on the value of α, which influences the spike representation of the processed corpus strongly, and is generally corpus-dependent. Probability function (1) aims to marginalize common words with low informational value. Its application during the sequential reading of a document can also be considered a random sampling mechanism. Each time a document is processed, its spiking representation differs slightly, thus increasing training samples' variability and protecting networks from overfitting.

Text-to-spike transformation is a challenging task due to the diversity of the documents in the datasets. In the same dataset, one can observe both brief documents that comprise only a few to a dozen words, as well as lengthier ones that might extend to several hundred words or more. Consequently, reading short and long documents only once will result in significantly different numbers of spikes representing the documents from both groups.

Such disparities can impact the learning process substantially, favoring longer passages of text. In extreme cases, short documents that are represented by just a few spikes might not activate neurons at all, leading to the information they carry being lost. To balance the variations in document length, we repeat the reading of each document until it is represented by a predefined number of spikes ns. The predefined number was estimated empirically, and this is explained in Section 6.3.

Another challenge arose from word frequencies. Since the emitting of a spike depends on the probability function and the α parameter value (Eq. (1)), the number of spikes emitted might be low in some time intervals. Low neuronal activity time intervals relate to a document's fragments, which contain many common words. This leads to time intervals without spikes, resulting in the sparse representation shown in Fig. 3 (top plot). In addition, higher α values lead to sparser representations. In contrast, text fragments that contain many uncommon words (words that occur infrequently in the dataset) are converted into higher numbers of impulses.

Such variations in the spike rates could affect training and the stability of the network dynamics and always increase the processing time of a document. This stems from the requirement to emit ns spikes for each document. If the representation is sparse, a document must be processed longer to satisfy the limit of spikes ns. To address this issue, if a word is skipped due to the evaluation of the probability function, we evaluate the next word of the sequence using the probability function (Eq. (1)). If the next word is omitted, we continue evaluating the subsequent words. After applying this approach, exactly one impulse is emitted in each time unit. This process leads to the dense representation illustrated in the bottom plot in Fig. 3.

3.2. Network architecture and plasticity

This section explains the STM architecture and the plasticity model. Fig. 4 presents a neural circuit used for text processing.

The input layer in Fig. 4 represents words from the dictionary. The activity level of each input neuron is determined by the sequential processing described in Section 3.1. The next layer is a topic layer, composed of leaky integrate and fire (LIF) (Long & Fang, 2010) neurons, i.e. topic neurons $(T_1, \dots, T_i \dots, T_K)$. The input and topic layers are fully connected through excitatory synapses. The strength of the synaptic connections between the layers is modified during the learning stage according to the plasticity model described below. The topic neurons are additionally connected to each other via inhibitory synapses. These do not change during the learning process: their sole purpose is to ensure competition between neurons. Implementing lateral inhibition enables the neurons to learn according to the competitive learning principle (Fritzke, 1997) and to become responsive only to a subset of the training patterns.

The topic neuron membrane's potential u is governed by the following equation:

$$\tau \frac{du}{dt} = (u_{rest} - u) + g_e(u_{exc} - u) + g_i(u_{inh} - u) \tag{2}$$

If no presynaptic activity occurs, excitatory and inhibitory conductance – g_e and g_i, respectively – decay to zero, and the potential membrane decays to u_{rest}. The firing of the presynaptic neuron connected to the excitatory synapse increases the conductance g_e, and the membrane potential rises towards u_{exc}. With sufficient stimulation, the potential crosses the threshold value u_{th}, and the neuron fires. After activation, the neuron becomes inactive for a refractory period of five milliseconds. When a presynaptic firing neuron is connected to an inhibitory synapse, the membrane potential decreases towards u_{inh}. Conductance decay is governed by the following equations:

$$\tau_e \frac{dg_e}{dt} = -g_e, \quad \tau_i \frac{dg_i}{dt} = -g_i \tag{3}$$

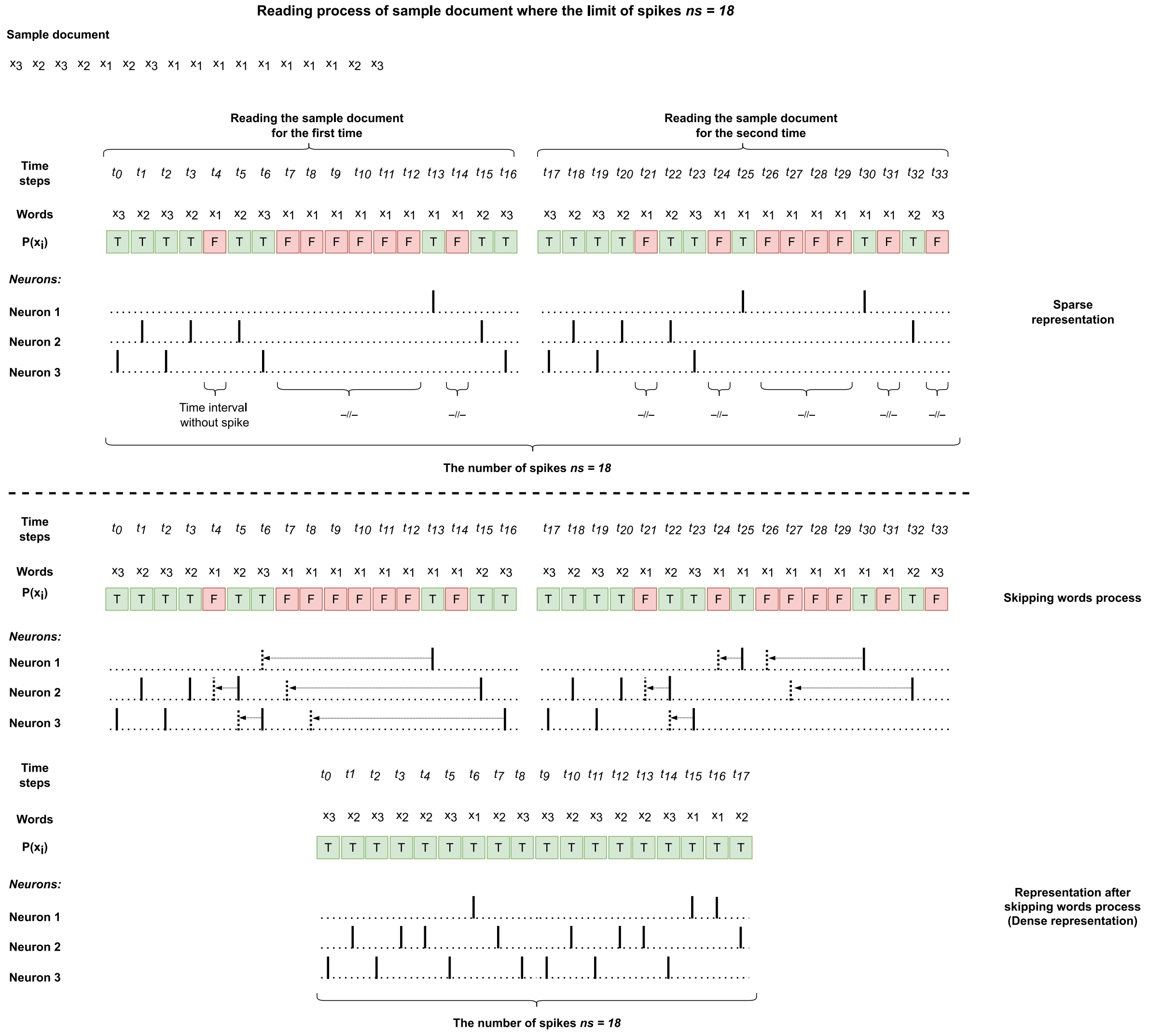
Reading process of sample document where the limit of spikes ns = 18
Sample document
x3 x2 x3 x2 x1 x2 x3 x1 x1 x1 x1 x1 x1 x1 x1 x2 x3
Reading the sample document for the first time
Reading the sample document for the second time
Time steps
t0 t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15 t16
t17 t18 t19 t20 t21 t22 t23 t24 t25 t26 t27 t28 t29 t30 t31 t32 t33
Words
x3 x2 x3 x2 x1 x2 x3 x1 x1 x1 x1 x1 x1 x1 x1 x2 x3
x3 x2 x3 x2 x1 x2 x3 x1 x1 x1 x1 x1 x1 x1 x1 x2 x3
P(xi)
T T T T F T T F F F F F F T F T T
T T T T F T T F T F F F F T F T F
Neurons:
Neuron 1
Neuron 2
Neuron 3
Time interval without spike
–//–
Sparse representation
The number of spikes ns = 18
Skipping words process
Time steps
t0 t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15 t16 t17
Words
x3 x2 x3 x2 x2 x3 x1 x2 x3 x3 x2 x3 x2 x2 x3 x1 x1 x2
P(xi)
T T T T T T T T T T T T T T T T T T
Representation after skipping words process (Dense representation)
The number of spikes ns = 18

Fig. 3. An illustration of how a sample document with 17 words is processed. The document contains nine words that activate *Neuron 1*, four that activate *Neuron 2*, and four that activate *Neuron 3*. It also contains several common words represented by x_1 for simplicity. In an actual document, it is rare for the same word to be repeated several times in a row. However, here, we simplify the situation to illustrate a case in which several common words occur consecutively. To better illustrate the problem of sparse representation, we assume that the probability function evaluation for word x_1 returns *False* for almost all evaluations. This is a reasonable assumption if α is relatively high and word x_1 is present in many documents in the dataset. The top section illustrates the result of the sparse encoding of the example document. Three time intervals can be observed in which no spikes are emitted (reading the sample document for the first time). The bottom section illustrates how evaluating the probability function of the next word in the sequence addresses the problem of sparse representation. Also, the bottom section illustrates the resulting dense representation.

where τ_e and τ_i are decaying time constants.

The strength of the synaptic connections between the input and topic layers is modified according to the STDP rule, equipped with an additional activity-dependent scaling factor analogous to that presented in Białas and Mańdziuk (2022). Weight modification is triggered only when the postsynaptic neuron fires. A modification of the synaptic connection w_{ij} between input layer neuron and topic layer neuron T_j can be expressed using the following equation:

$$\Delta w_{ij} = \eta(A(t) - R(t)w_{ij}^2) \tag{4}$$

where η is a small learning constant, $A(t)$ is a presynaptic trace, and $R(t)w_{ij}^2$ is the activity dependent scaling factor. Please note that the scaling factor in Eq. (4) is modified compared to that of Białas and Mańdziuk (2022), in which it is of the form $R(t)w_{ij}$. We observed that applying smoother scaling (i.e. $R(t)w_{ij}^2$) generally resulted in improved model performance. If the presynaptic neuron fires, the $A(t)$ factor is set to 1 and decays with time according to Eq. (5):

$$A(t) = e^{-\frac{t-t_{pre}}{\tau_+}} \tag{5}$$

where t_{pre} is the time of the last activation of this presynaptic neuron, and τ_+ is a time constant. Depending on the difference between the presynaptic trace $A(t)$ and the value of the scaling factor $R(t)w_{ij}^2$, the weight either increases of decreases. $R(t)$ expresses the postsynaptic neuron activity history and is incremented by one each time the postsynaptic neuron is activated. It decays with time according to the following formula:

$$R(t) = \sum_{k=1}^{N} e^{-\frac{t-t_k}{\tau_r}} \tag{6}$$

where N is the number of all previously observed spikes, t_k is the time of the occurrence of the spike with index k, and τ_r is a decay time constant. If no previous neuron activity had occurred, the scaling factor $R(t)w_{ij}^2$ will be 0, and the dominant factor in the learning rule will be the presynaptic trace, defined by Eq. (5). This will result in an increase in the synaptic efficacy. If the postsynaptic neuron activity rises, the scaling factor increases and might become dominant in Eq. (4). In such cases, the weight of the synaptic connection decreases. The synaptic scaling is a type of homeostatic mechanism (Turrigiano, 2012) that

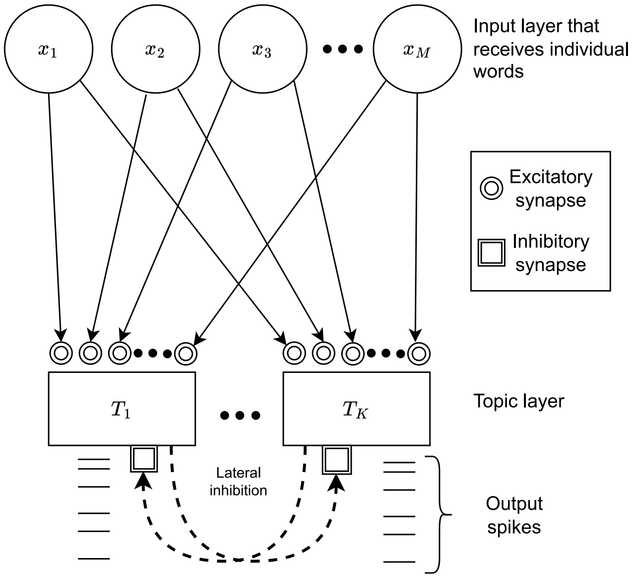

Fig. 4. An overview of the STM neural network architecture.

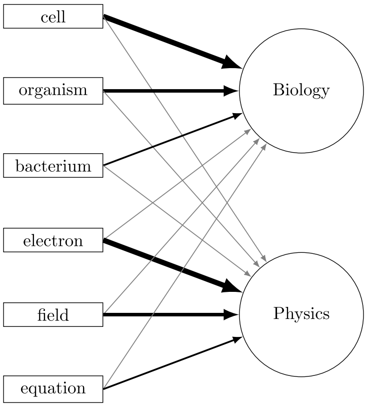

Fig. 5. The last step in topic modeling with two neurons and their synaptic connections is represented by the arrows. The arrows' thickness indicates connection strength. Analyzing the most significant weights from these neurons helps identify recurring themes in the passages that are under consideration. The primary weights in the first neuron concern *biology*, while those of the second neuron concern *physics*.

prevents rapid increases in neurons' activity, as is typically observed in HL.

3.3. The SNN as a topic model

In the STM, each neuron of the topic layer represents a separate topic, and each neuronal weight corresponds to one of the words from the dataset analyzed. Since the neurons compete with each other, we can expect that after training, each neuron will correspond to a subset of the documents from the dataset. Fig. 5 illustrates this phenomenon. Topic models first learn the latent topics from the text, and then each topic can be described by the most significant words (Yamazaki et al., 2022). Moreover, TM (and STM) can be used in document clustering. For this purpose, a document must be processed through the STM network to ascertain the topic distribution. The STM then yields a K-dimensional vector for each document, with each dimension corresponding to a distinct topic. More precisely, each vector coordinate is the rate of the respective topic, which means that a document's membership to a cluster can be determined by identifying the most significant rate (coordinate) in its corresponding vector.

4. Empirical evaluation

4.1. Datasets and preprocessing

Three datasets were used in the experiments: *20Newsgroups*,[2] *BBC news* (Greene & Cunningham, 2006), and *AG news* (Corso et al., 2005). This combination of datasets offers a comprehensive assessment across scales and ranges of complexity. *BBC News* enables initial exploration. *AG News* balances size and manageability, and enables the exploration of a potentially real-world corpus (127,599 documents constitutes quite a moderate collection for analysis). Each of the datasets has rich vocabulary and context. For instance, *20Newsgroups* contains text with informal language, including slang and abbreviations. Furthermore, it is a collection of real-world newsgroup discussions on various topics, with varying lengths (from 5 to 4465 words) and containing imperfections of real conversations. All of these can be challenging for topic modeling algorithms to parse and accurately represent the underlying themes.

For each dataset, the following prepossessing was applied: (1) *Cleaning*—stop-words, punctuation, and all nonalphanumeric tokens were removed; (2) *Lemmatization*—each word was reduced to its basic form (its lemma), e.g. raining to rain, books to book; and (3) *Feature selection*—we counted all occurrences of each word in a dataset and defined a dictionary of M most frequent words. Restricting the number of features to reduce computation time is applied frequently in vector space methods (Deng et al., 2019; Kou et al., 2020).

Table 1 presents the main statistics of the preprocessed datasets.

4.2. Comparative models

We compared the performance of various instances of the STM model with four well-known TM methods. To ensure a comprehensive comparison, we selected two probabilistic methods called the latent Dirichlet allocation (LDA) (Blei et al., 2003) and the biterm topic model (BTM) (Cheng et al., 2014), as well as two neural network-based approaches: the embedding topic model (ETM) (Dieng et al., 2020) and the BERTopic model (BT) (Grootendorst, 2022). The comparative methods are widely adopted models, each used commonly in practical applications. They all serve as standard references in TM (Abdelrazek et al., 2023; Chauhan & Shah, 2021; Jelodar et al., 2019b; Murshed et al., 2022; Qiang et al., 2022; Vayansky & Kumar, 2020b). A brief overview of these methods is presented in Appendix.

4.3. Training procedure

The training of a single STM model comprised two stages. During the first stage, we transformed the text into spikes according to the procedure described in Section 3.1. During the second stage, the network processed the spike trains and modified the weights. All experiments were performed using the *brian2* framework (Stimberg et al., 2019).

We trained the models with three topic sizes, $K = [20, 30, 40]$. For each K, training was repeated five times for further statistical analysis. The same training procedure was performed for all datasets, resulting in $3 \times 5 \times 3 = 45$ trained models. The model's parameters and details of the training procedure are presented in Section 6. The comparative models were trained according to the same schema.

4.4. Quantitative evaluation of topic quality

An essential feature of TM is the interpretability of its topics. Quantifying interpretability is a difficult task, however. One method

[2] http://qwone.com/~jason/20Newsgroups/20news-bydate.tar.gz

Table 1
Statistics of the datasets used in our experiments.

	No. of examples	No. of categories	No. of features	Q1 percentile (in words)	Q2 percentile i.e. median document length (in words)	Q3 percentile (in words)	Min. document length (in words)	Max. document length (in words)	Example categories
20Newsgroups	18,846	20	5,000	36	57	94	5	4,465	*politics, religion, computers, hobby*
BBC News	2,225	5	2,000	83	116	164	31	1,200	*business, entertainment, technology*
AG News	127,599	4	5,000	15	18	22	1	75	*world, sports, business, sci/tech*

widely used in its evaluation involves examination of topic coherence (Newman et al., 2010), which measures the degree of semantic similarity between words that describe a topic. The semantic similarity of the words is measured based on word co-occurrences in a reference document corpus. Our evaluation uses one coherence metrics: C_{NPMI} (Röder et al., 2015), which is available in the Palmetto library.[3] The metric use English-language Wikipedia[4] as a reference corpus. Consider a topic T_k[5] described with the most significant words $T_k = [x_{1_k}, x_{2_k}, \dots, x_{s_k}]$. The C_{NPMI} coherence (Aletras & Stevenson, 2013) utilizes normalized pointwise mutual information (NPMI) as a measure of the semantic similarity between words in the topic, which is defined as follows:

$$NPMI(x_i, x_j) = \frac{log_2 \frac{P(x_i,x_j)+\epsilon}{P(x_i)\cdot P(x_j)}}{-log_2 P(x_i, x_j)} \tag{7}$$

where $P(x_i, x_j)$ is the probability of co-occurrence of a pair of words x_i, x_j in the sliding window that moves over the reference corpus. $P(x_i), P(x_j)$ are the probabilities of independent occurrences of the respective words, and ϵ is a small constant. C_{NPMI} is defined as the $NPMI$ arithmetic mean calculated for all possible pairs of words (x_i, x_j) that describe a topic, and is defined by the following equation:

$$C_{NPMI} = \frac{2}{s(s-1)} \sum_{i=1}^{s-1} \sum_{j=i+1}^{s} NPMI(x_i, x_j) \tag{8}$$

where s is the number of words that represent the topic. We use $s = 10$, a common choice in the experimental TM research.

Another measure that we used to evaluate each topic model is the proportion of unique words (PUW) (Dieng et al., 2020), defined as follows:

$$PUW = \frac{z}{K \cdot s} \tag{9}$$

where K is the number of topics of the model, z is the number of unique words in all of those topics, and s is the number of words that define the topic ($s = 10$). PUW aims to express the diversity of the topics quantitatively.

The last indicator we used is topic quality (TQ) (Dieng et al., 2020), a multiplication of the diversity measure (PUW) and the coherence measure C_{NPMI}. The following equation describes the respective TQ measure:

$$TQ_{NPMI} = C_{NPMI} \cdot PUW \tag{10}$$

The evaluation of topic models on single indicators can be misleading. A comprehensive approach should consider multiple metrics. A high PUW score indicates a well-diversified topic set, in which topics contain fewer repeated words. For example, a model might extract high C_{NPMI} value, which indicates that it performs well. However, in extreme cases, the model might copy the same topic several times or modify it only slightly, thus obtaining high values of the indicator. Combined metrics like TQ_{NPMI} address these limitations by incorporating PUW, which offers a more nuanced view of TQ.

4.4.1. Evaluation results

Before analyzing the experimental results, it is worth noting that no definitive benchmark exists to delineate high or low coherence scores categorically. Consequently, the most suitable method of interpreting coherence values involves a comparative analysis across various models. A higher coherence value typically indicates that the topic is more interpretable and that the words are more closely related semantically.

Table 2 presents the results for the average coherence metrics C_{NPMI}, as well as the diversity (PUW) and TQ (TQ_{NPMI}) values.

High TQ_{NPMI} scores in Table 2 indicate a model's ability to capture well-diversified, coherent topics. Achieving high results in this measure when evaluation is performed against an external corpus means that the topic's words co-occurrence are getting closer to that observed on the large corpus used for evaluation. BERTopic and STM achieve the best results for all datasets in the TQ_{NPMI} measure. On the *BBC News* dataset, STM achieves the best performance of all models. The modern BERTopic model, which leverages a powerful language model based on multilayer transformer neural architecture, takes the lead on the remaining two datasets. It is worth noting, however, that a marginal difference can be observed between the STM and BERTopic models on the *AG news* dataset. This is an important observation because it demonstrates the applicability of the STM to short-text topic modeling. The *AG News* dataset is a large collection of 127,599 documents, and the median document length is only 18 words. This kind of dataset causes additional difficulty for the TMs due to its high sparsity.

The quality of identified topics is contingent upon the method employed and the intricacy of the dataset, influenced by factors such as document length, semantics, and vocabulary richness. The poorest results are on the *20Newsgroups* dataset due to its diverse document lengths and inclusion of noisy elements like colloquialisms, headers, and misspellings. Classical models like LDA and BTM struggle to capture and organize such data effectively, resulting in considerably lower TQ_{NPMI} scores compared to ETM, BERTopic, and STM methods. However, even with a robust method like BERTopic, some problems can be observed in the case of the *20Newsgroups* dataset. This concerns the situation when the number of topics is $K = 40$; in that case, the coherence value drops by half compared to $K = 20$. The BERTopic selects words by clustering the documents and choosing representative words from each cluster. Certain words that are good or characteristic from a clustering perspective do not always create a coherent topic.

In contrast, all methods exhibit better performance on both the *AG News* and *BBC News* datasets, attributed to their higher data quality. Nonetheless, it is worth mentioning that classical LDA and BTM methods still attain TQ_{NPMI} scores nearly half those achieved by the top-performing methodologies (BERTopic and STM).

4.5. Quantitative cluster evaluation

As presented in Section 3, STM output takes the form of a vector in which each coordinate is the rate of the respective topic, so that a document's membership to a cluster can be determined by identifying the most significant element of its vector representation. To evaluate the accuracy of the clusters quantitatively, we utilize the cluster *purity* indicator (Manning et al., 2008), a well-known metric used to assess the degree to which a clustering algorithm has grouped data into homogeneous clusters. Purity is calculated by assigning the majority class label to each cluster based on the ground truth labels of the

[3] https://github.com/dice-group/Palmetto/wiki/How-Palmetto-can-be-usedl

[4] https://en.wikipedia.org/wiki/English_Wikipedia

[5] T_k can be used to represent both a topic neuron and the set of words most strongly associated with that neuron. In the latter sense, the topic refers to a collection of words that have high activation values for a specific T_k neuron.

Table 2
The average results of the coherence, diversity, and TQ metrics. The best outcomes for each type of experiment are presented in bold. The second-best results are marked with asterisks. The average results across all topic configurations are denoted by rows labeled as *avg*. BT is stand for BERTopic method.

	K	20Newsgroups					BBC News					AG News				
		BTM	LDA	ETM	BT	STM	BTM	LDA	ETM	BT	STM	BTM	LDA	ETM	BT	STM
C_{NPMI}	20	0.034	0.030	0.052	0.079	0.076	0.064	0.063	0.075	0.094	0.118	0.073	0.067	0.068	0.126	0.120
	30	0.034	0.031	0.050	0.069	0.058	0.070	0.064	0.072	0.093	0.110	0.081	0.074	0.070	0.120	0.116
	40	0.034	0.027	0.047	0.039	0.050	0.064	0.054	0.066	0.090	0.104	0.084	0.069	0.076	0.115	0.114
	avg	0.034	0.029	0.049	**0.062**	0.061*	0.066	0.06	0.071	0.092*	**0.110**	0.079	0.070	0.071	**0.120**	0.117*
PUW	20	0.686	0.734	0.879	0.916	0.829	0.660	0.765	0.810	0.893	0.841	0.800	0.877	0.912	0.918	0.942
	30	0.683	0.704	0.805	0.878	0.780	0.621	0.777	0.778	0.831	0.776	0.746	0.861	0.865	0.923	0.912
	40	0.659	0.694	0.768	0.850	0.745	0.588	0.730	0.730	0.793	0.739	0.701	0.874	0.803	0.894	0.881
	avg	0.676	0.711	0.817*	**0.881**	0.785	0.623	0.757	0.773	**0.839**	0.785*	0.749	0.871*	0.860	**0.912**	**0.912**
TQ_{NPMI}	avg	0.023	0.021	0.040	**0.055**	0.048*	0.041	0.045	0.054	0.077*	**0.087**	0.059	0.061	0.061	**0.110**	0.107*

Table 3
The $purity$ values of the BTM, LDA, ETM, BT, and STM document topic representations ($K = 20, 30, 40$) calculated for the *BBC News*, *20Newsgroups*, and *AG News* datasets. The best results in each group are presented in bold. The second-best results are marked with asterisks. The values in brackets indicate standard deviation. The $purity$ values are expressed as percentages. BT is stand for BERTopic method.

	K	$purity$				
		BTM	LDA	ETM	STM	BT
20Newsgroups	20	45.47 (1.07)	49.23 (2.3)	47.75 (1.06)	**60.59 (1.73)**	57.87 (2.02)*
	30	48.92 (1.78)	54.35 (1.87)	53.51 (1.81)	**61.88 (1.13)**	61.24 (1.87)*
	40	52.41 (1.06)	56.78 (1.88)	55.12 (2.83)	**60.51 (1.59)**	60.02 (4.17)*
BBC News	20	91.93 (0.83)	90.8 (0.78)	90.17 (0.42)	91.97 (1.02)*	**93.11 (0.25)**
	30	91.46 (0.63)	90.69 (1.39)	90.44 (0.93)	91.86 (0.99)*	**93.10 (0.31)**
	40	92.26 (0.94)	90.02 (0.78)	90.27 (0.65)	92.34 (1.01)*	**94.18 (0.28)**
AG News	20	**83.00 (0.73)**	78.8 (1.65)	79.66 (0.44)	80.92 (1.35)*	78.21 (0.41)
	30	83.47 (1.0)*	79.85 (0.77)	78.34 (0.49)	81.07 (1.12)	**85.33 (0.18)**
	40	84.02 (0.56)*	79.02 (0.97)	79.61 (0.66)	81.27 (0.92)	**86.12 (0.25)**

documents before calculating the fraction of items across all clusters that represent the majority class:

$$purity = \frac{1}{|C|}\sum_{i=1}^{K}\max_{c} n_i(c) \tag{11}$$

where $|C|$ is the total number of documents in the dataset, K is the number of clusters,[6] and $n_i(c)$ is the number of documents in the ith cluster belonging to class c. The max function selects the class with the highest number of documents.

4.5.1. Evaluation results

The results of experimental evaluation are presented in Table 3 in the form of clustering purity.

The *20Newsgroups* dataset has the most complicated categories structure and the highest number of classes. This dataset is organized into a hierarchy of topics. For example, the broader category of *comp* (computers) includes more specific groups such as *comp.os.ms-windows.misc* and *comp.sys.ibm.pc.hardware*. The hierarchical structure brings additional challenges for the algorithms because an algorithm needs to learn to distinguish between very diverse topics (like *computer* versus *sport*) and between closely related subtopics within the same broader category. This is well illustrated in Section 4.7, where we further analyze this dataset with STM. Because of that, $purity$ results are the lowest for this dataset across all the methods studied.

The best-performing model on this dataset is STM, followed closely by BERTopic a topic modeling technique that leverages BERT embeddings for texts to create dense clusters of documents. BERTopic did not achieve the highest $purity$ results on *20Newsgroups* for the two following reasons. The first one is a semantic similarity between the categories mentioned above. The second one is how the BERTopic defines the topics. This method assumes that each cluster is one topic. The clustering is performed using the state-of-the-art (SOTA) method HDBSCAN. In HDBSCAN, one cannot determine the number of clusters; the algorithm at each run delivers a different number of clusters. If BERTopic is used with the default setup, the number of clusters for *20Newsgroups* is huge. Also, the topics represented by such clusters have much lower coherence. To achieve SOTA coherence scores, we followed the approach recommended by the author and used in the original paper (Grootendorst, 2022). With such an approach, the HDBSCAN needs to be configured to deliver a smaller number of dense clusters, and for the evaluation, we use only K's most popular clusters (topics). Because of that adaptation, the $purity$ scores can be lower even though the BERTopic is based on a SOTA clustering method.

For *AG News* dataset, the best results were achieved for BERTopic and followed by BTM. *AG News* is an extensive collection of short documents. For this dataset, utilizing contextual embedding gives BERTopic a significant advantage over other models, resulting in the highest $purity$ for the configuration $K = 30$ and $K = 40$. In the case of $K = 20$, lower results for BERTopic come from the topic selection approach described in the paragraph related to *20Newsgroups*.

Interestingly, BTM achieves high results for this dataset as it is designed to work with extensive collections of short documents. The reason for that is the way the model represents documents, namely not with words but with bi-terms. Bi-terms refer to pairs of words that co-occur within a short, fixed-size text window. For the *AG News*, that kind of document representation resulted in more distinctive clusters indicated by the higher $purity$ score.

Finally, for the *BBC News* dataset, the best results were achieved by BERTopic, followed by STM. This dataset is the simplest of the tested ones, and contains long documents that belong to five categories. The usage of effective clustering technique and contextual embedding gives the BERTopic an advantage over other models.

4.6. Qualitative topic illustration

We extracted, ten topics for each dataset from one of the 40-neuron STM models. These include the five with the highest coherence (top-5)

[6] K, the number of clusters is equivalent to a number of topics or topic neurons.

20newsgroups	BBC News	AG News
1 patient disease medical doctor treatment infection health cancer medicine drug	1 race olympic champion championship indoor gold medal finish athlete win	1 trial charge prison sentence court judge jail guilty case abuse
2 team player play nhl hockey game season league year canada	2 club chelsea cup league manager season arsenal football game premiership	2 coach touchdown football quarterback bowl yard season nfl play college
3 space launch nasa shuttle moon program mission cost satellite orbit	3 tory minister mp secretary labour party government conservative howard blair	3 phone mobile wireless communication cell service nokia internet broadband network
4 christian god jesus bible church christ christianity paul book word	4 tv show series star channel programme viewer television film celebrity	4 rate economy growth interest reserve economic federal rise consumer inflation
5 god sin jesus christ lord heaven life son father hell	5 search google web user engine information site yahoo tool internet	5 league arsenal manchester champion club england unite goal striker cup
36 fire fbi waco koresh burn atf batf child compound survivor	36 music urban joss stone black soul award artist term describe	36 iran nuclear uranium enrichment tehran atomic weapon iranian program agency
37 henry orbit pat mission spencer prb zoology toronto earth planet	37 france england side wale ireland scotland nation kick penalty victory	37 police kill people bomb dead injure blast death japan explosion
38 keith livesey morality jon objective schneider allan atheist moral political	38 bid shareholder takeover club board glazer deutsche boerse exchange unite	38 lawsuit settle file sue industry court pay insurance suit commission
39 max bhj gk giz bj kn qax nrhj lj biz	39 drug test ban sprinter thanou kenteris greek iaaf olympics athletics	39 drug scientist study researcher human find vioxx merck specie risk
40 gordon bank geb pittsburgh shameful skepticism surrender intellect chastity univ	40 liverpool club summer gerrard steven madrid parry deal benitez chelsea	40 eu european union trade boeing airbus commission turkey wto brussels

Fig. 6. Sample topics extracted from the selected 40-neuron STM models. For each dataset, blue represents the top-5 coherent topics, while orange represents the bottom-5 topics in terms of coherence.

and the five with the lowest (bottom-5). We used a ranking according to the C_{NPMI} coherence metric to select the topics. Fig. 6 presents the selected topics for all datasets.

The top-5 topics of the *20Newsgroups* dataset are easily interpretable and correspond to the predefined categories in the dataset. They include subjects that relate to *medicine*, *hockey*, *space flights*, and *religion*. Among the bottom-5 topics extracted from the *20Newsgroups* dataset, *Topic 36* initially appeared challenging to interpret. However, during a more profound analysis of the words that define the topic, it transpired that it pertains to tragic events that unfolded in the city of Waco, USA in 1993, which involved a standoff and assault between the Branch Davidians religious sect, led by David Koresh, and the law enforcement agency, ATF/BATF. The Waco Siege ended in a fire that claimed the lives of many members of the sect, including Koresh and several children. This is a prime example of a topic that requires some domain knowledge to be interpreted, and its coherence score is insufficient to judge the topic's interpretability. *Topic 38* seems to relate to discussions on *atheism*, *morality*, and *politics*; however, this is not immediately apparent because the topic is quite noisy and contains several proper nouns. *Topics 37*, *39*, and *40* are truly challenging to interpret. A particularly interesting case is *topic 39*, whose appearance is caused by a set of several highly specific documents that contain difficult-to-interpret sequences of letters.

In the case of the *BBC news* and the *AG news* datasets, the differences observed between the highest and lowest ranked topics is relatively smaller. For instance, in *Topic 1* of the *BBC News* dataset, most words relate to sports terms, specifically to the Olympics, while in *Topic 4*, the words relate to show business. Even topics from further positions are rather informative. For example, *Topic 40* generally describes sport-related subjects, despite its lower coherence. In this case, domain knowledge and the analysis context are more important than in previous topics (this is similar to *Topic 36* of the *20Newsgroups* dataset). When analyzing each word from this topic separately, it might be difficult to identify which common concept underlines the words. Individually, terms such as *liverpool* and *madrid* are city names, and the term *gerrard* might suggest any person's name, but a soccer fan will interpret this topic rather easily.

The same observations apply to the *AG News* dataset. For example, *Topic 1* contains words that relate to legal terminology, and *Topic 4* acquires terms from economics. For other topics, the words remain thematically coherent but their interpretation requires context understanding to reveal it. For instance, *Topic 36* contains city names and nuclear-related terms, and probably relates to the Middle East's political situation. Again, the topics with relatively lower coherence seem more specific and might require more profound knowledge to be interpreted; however, even for the average reader, they offer valuable insights.

4.7. Topic modeling use cases

4.7.1. Thematic similarity of documents

The main purpose of topic modeling is to discover latent topics in collections of documents that can be used to analyze textual data. The presented example demonstrates how the STM can be used to analyze a dataset. As an example, we use the *20Newsgroups* dataset due to its complex class structure. Our model generates a set of topics represented by neurons. Each document activates a subset of these neurons based on its thematic content. The activity of each neuron signifies the document's relevance to that topic. Additionally, each document can be represented as a vector whose coordinates are defined by the neurons' activity. Using the vector representation of documents, the cosine similarity between them can be determined, as illustrated in

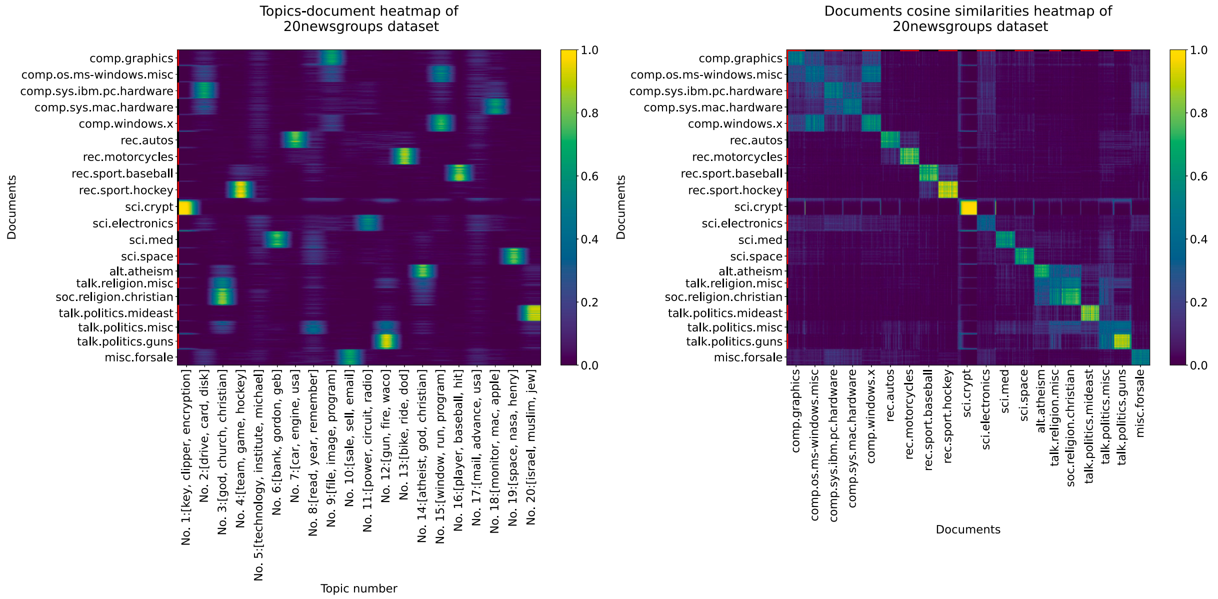

Fig. 7. A visualization of the *20Newsgroups* data with the STM topics. On the left, the x-axis presents each topic's first three key terms.

the right heatmap of Fig. 7. The left section of Fig. 7 presents the topic distribution across the documents.

To enhance the visualization of topic distribution across classes and their relationships with documents, we arranged them according to the original categories from the *20Newsgroups* dataset. The names of the categories are positioned on the y-axis of the chart. Many topics demonstrate strong correlations with particular thematic groups in the dataset collection. For instance, *Topic 3* – characterized by the key terms *god*, *church*, and *christian* – predominantly appears in passages from *religion* related categories and to a lesser extent, in documents from the categories of *atheism* and *politics*. Applying a similar analysis to a dataset in which the classes are unknown would facilitate the grouping of the documents into a thematic cluster defined by the constituent words of the topic.

Some topics are more distinct and correlate strongly with only one category. For example, *Topic 19* – defined by the terms (*space*, *nasa*, and *henry*) – is found primarily in documents classified under the *space* category. From the standpoint of TM, the presence of a topic in multiple classes of a dataset is considered appropriate if the topic itself is coherent. For example, *Topic 2* – which includes the words *drive*, *card*, and *disk*, as well as *buy* and *price* – is informative from the perspective of topic modeling, as it suggests that the documents relate to the sale of computer equipment, as well as discussions about that subject on a computer forum. However, from the perspective of document classification this topic is too generic to discriminate classes. This phenomenon is well exemplified by the document similarity heatmap. Despite several artifacts, the heatmap reveals a few notable clusters that group documents by similar themes, yet which belong to separate categories. The largest of these clusters relates to computer equipment, encompassing all categories that begin with the prefix *comp*. The emergence of this cluster results from the co-occurrence of *Topic 2*, *Topic 9*, *Topic 15*, and *Topic 18* in various documents that concern computer equipment.

Based on the visualization developed from the extracted topics, it can be concluded that the *20Newsgroups* dataset contains several large semantically similar groups of documents, whose contents interpenetrate. Additionally, more homogeneous clusters that relate to the categories – such as *sci.crypt*, *sci.electronics*, *sci.med*, and *sci.space* – can be organized around a singular extracted theme.

4.7.2. Disambiguation words meaning

The STM model can also disambiguate word meanings based on context. For instance, *player* in the *20Newsgroups* dataset can be associated with *baseball* or *hockey*. Similarly, *driver* can refer to a car operator or a software program. To check and evaluate the words and phrases disambiguation capabilities of the STM, we input ambiguous words or phrases into the network and observe the detected topic in the output. The following examples in Table 4 illustrate this process using *player* and *driver*. Table 4 presents eight phrases (four with *player* and four with *driver*). It also shows each phrase's most active words from the associated topic.

The examples presented in Table 4 focus solely on the strongest topic, i.e. the one which exhibits the highest activity for the given word or phrase. This means that in each case, we select the strongest topic from other topics activated by a specific word or phrase. For instance, when presenting the word *player* without contextual clues, both sports-related topics will be activated, i.e. the first one, which relates to baseball and the second one, which relates to hockey. However, the activity of the baseball-related topic surpasses that of hockey, and thus it is selected. This occurs because the word *player* has a stronger synaptic connection in the baseball-related topic than in the hockey one. A similar analysis applies to the processing of the word *driver*. The word activates two different topics. One relates to the class of cars and another one to computers, which is characterized by the words *card*, *video*, *monitor* and *window*. Finally, the selected topic is that which relates to computers because the word *driver* has a stronger synaptic connection in the computer-related topic than in the car one. Moreover, Table 4 demonstrates that STM distinguishes phrases' meanings based on the context provided. By adding context to the words *driver* and *player*, their meanings can be disambiguated and they can be categorized accurately.

4.8. Summary

The following conclusions can be drawn from the experiments:

- Quantitative results:
  - STM, a single-layer network, achieves impressive results, rivaling modern methods based on artificial neural architectures like ETM and those leveraging powerful transformer-based language models like BERTopic. Notably, it surpasses well-established TM techniques like LDA and BTM.
  - Combining C_{NPMI}, and PUW metrics enables a more nuanced understanding of a model's performance (e.g.

Table 4
Topic-based phrases disambiguation examples.

Phrase	Topic with highest neuronal activity
player	*game baseball player team hit win pitch pitcher play league*
player ball	*game baseball player team hit win pitch pitcher play league*
player ice	*game team play hockey playoff player nhl win fan goal*
player canada	*game team play hockey playoff player nhl win fan goal*
driver	*card video driver monitor window graphic mode ram version color*
driver computer	*card video driver monitor window graphic mode ram version color*
driver ford	*car engine buy drive mile price usa ford dealer model*
driver toyota	*car engine buy drive mile price usa ford dealer model*

TQ_{NPMI}). Relying solely on C_{NPMI}, or PUW can be misleading, as it does not always capture TQ effectively.

- Different topic modeling methods make different assumptions about the structure of topics within documents. A method that aligns well with the data structure may model topics in a way that assures high $purity$, which does not mean it will offer good TQ.
- In the context of document clustering, the STM model demonstrates its effectiveness on the *20Newsgroups* and *BBC news* datasets. This suggests that STM is a strong contender for document clustering, though its performance might vary depending on the specific characteristics of the data.

- Qualitative results:
  - Qualitative analysis indicates that the top-5 topics in *20Newsgroups* are easily interpretable in the range of their predefined categories, while lower-ranked topics pose more challenges in their comprehension. In contrast, *BBC news* and *AG news* exhibit less distinction between higher- and lower-ranked topics.
  - Additionally, it is demonstrated that valuable information can be extracted even from the topics with lower coherence scores, thus underscoring the potential for mining insights from less coherent content.
  - The qualitative assessment of topics extracted from the STM models aligns with the quantitative indicators of TQ. The most difficult to interpret are those found in the *20Newsgroups* dataset, which also have the lowest average coherence values among all datasets.
  - The provided use cases, particularly the topic and document heatmaps, demonstrate the core strengths of STM as a topic model: data exploration and explanation. STM is capable of creating interpretable topics-based representation. In the examples, we can clearly see semantic structures emerge, like the main topics and subgroups within the nested *20Newsgroups* dataset. Such interpretable representation serves as a valuable input for various downstream NLP tasks. This interpretability can enhance performance in tasks like document classification (Voskergian et al., 2023), information retrieval (Wang et al., 2007), word disambiguation (Brosseau-Villeneuve et al., 2014; Kim & Yoon, 2015), or sentiment analysis (Seilsepour et al., 2023).

5. Scalability

5.1. Comparison with other models

In Table 5, we report the training times and memory usage for all the methods and datasets studied. The training time is presented for the highest configuration of $K = 40$ topics. The reported memory is the peak memory requirement measured across the entire training process. We also report memory usage from the graphic card for models that use GPU.

The shortest training time was recorded for BERTopic. This model is unique because, unlike other methods used for comparison, it does not learn topic representations, but clusters documents instead. BERTopic leverages a pre-trained BERT language model to convert documents into numerical embeddings, which are then clustered to form topics. Topic representatives are determined based on the words' frequency and distribution across these clusters, providing a semantic interpretation of the data. However, it is worth noting that this model utilizes a previously trained language model, and training the large language models requires a lot of time and resources.

Now, let us consider the remaining models that learn representations based on the datasets provided. In the case of the *BBC News* and *20Newsgroups* datasets, the training times are comparable across all examined models. More significant differences can be observed for the *AG News* dataset, where the BTM model is the fastest. It needs nearly four times less training time than the other models. This stems from the bi-term representation of the input documents used in BTM, which is particularly time-efficient for short documents, because of the relatively lower number of bi-terms representing each document. On the other hand, this model required higher computation time for 20Newsgroups, which, among others, contains also very long documents.

Regarding memory consumption, the most effective model across all datasets is LDA. The memory consumption of STM is the highest among the models, which is further discussed in the following section.

5.2. Implementation details and scalability

It is essential to note that training time and memory consumption of a particular model strongly depend on its implementation. The STM implementation is based on the *Brain2* framework. The relatively high memory consumption of the model comes from the fact that during training the model is fed with the whole training set. This leads to faster training because, for each epoch, the model is compiled only once. The side effect, though, is higher memory consumption. When the model is compiled, we observe the memory peak; however, after several seconds, the memory consumption decreases. For example, when we train the model with *20Newsgroups* dataset, the memory peak at compilation time is more than 2.5 GB. After several seconds, memory consumption decreases to 1 GB and stays at this level till the end of training.

We also analyzed STM scalability with a higher topic number regime, using the *20Newsgroups* dataset and measuring the training time for ten epochs. Increasing the number of neurons 5 times to $K = 200$, led to the training time increase by about 50%. At the same time, memory consumption rose only by 200 MB, which is less than 10% of the memory consumption in the base case of $K = 40$. Similar dependencies are observed for other datasets, which proves good STM scalability in terms of increasing the number of neurons.

STM can process over 100,000 documents in approximately 40 min, proving its time-efficiency for moderate-sized datasets. However, the model's training time increases linearly as the number of documents grows, so it would be required to enhance the model's implementation when the handled corpus would contain millions of documents. A possible solution to this limitation would be parallel implementation, where

Table 5
Comparison of memory consumption and training time across studied methods: BTM, LDA, ETM, BT (BERTopic method), and STM, calculated for the *BBC News*, *AG News*, and *20Newsgroups* datasets.

	No. of features	No. of examples	BTM		LDA		ETM		BT		STM	
			Time	RAM	Time	RAM	Time	RAM	Time	RAM/RAM GPU	Time	RAM
BBC News	2000	2,225	2.6 min	700 MB	1.9 min	201 MB	2.15 min	1.34 GB	0.5 min	400 MB /500 MB	2.2 min	2.1 GB
AG News	5000	127,599	13.5 min	2.5 GB	46.2 min	570 MB	70 min	1.69 GB	1.7 min	2.8 GB/526 MB	43.1 min	5.5 GB
20Newsgroups	5000	18,846	15.0 min	2.5 GB	18.9 min	395 MB	20.5 min	1.76 GB	1.3 min	850 MB /526 MB	16.2 min	2.5 GB

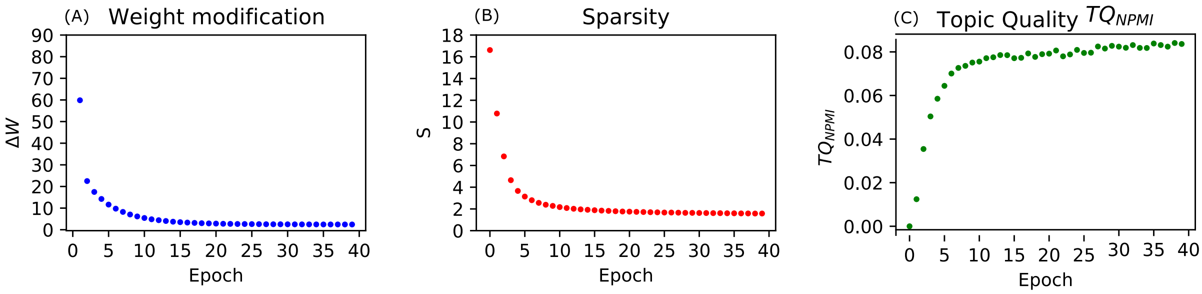

Fig. 8. The dynamics of the model and performance metrics as functions of learning epochs.

distinct instances simultaneously process various parts of the dataset. We intend to address this issue in our future work. Another potential solution is a GPU-based implementation, discussed in Section 7.

Furthermore, the model could benefit from the neuromorphic adaptation, using modern neuromorphic hardware, e.g. various variants of the STDP-based learning rules (Schuman et al., 2022) that could be combined with the proposed encoding schema.

6. Learning procedure and parameters

6.1. Learning convergence and stability analysis

To better describe the network's dynamics during learning, we introduce two metrics. The first, ΔW, quantifies the scale of network weight changes during a single training epoch, and is governed by the following equation:

$$\Delta W_i = \frac{1}{K}\sum_{k=1}^{K}\frac{|\vec{w}_{k,i} - \vec{w}_{k,i-1}|}{|w_{\overrightarrow{k,i-1}}|}100\% \tag{12}$$

where $\vec{w}_{k,i}$ represents the weight vector of neuron k in epoch i, and K denotes the total number of neurons in the network. ΔW_i expresses the average value of changes in weight vectors after the ith epoch of training relative to the weights from the epoch $i-1$.

The second metric, S, quantifies the sparsity of the document representation. S measures how many neurons, on average, are activated during a single document presentation:

$$S = \frac{1}{|C|}\sum_{i=1}^{|C|} s_i \tag{13}$$

where $|C|$ is the total number of documents in the dataset, and s_i is the number of neurons activated during the presentation of the ith document. To calculate S, the whole dataset is presented to the network after each training epoch.

To illustrate how the network dynamic changes in time and how it affects different metrics, we performed separate simulations in which we trained five STM models of 40 neurons, or 40 epochs, using the *BBC news* dataset. In each epoch, we calculated four different metrics: TQ_{NPMI}, ΔW, and S. The mean values of the metrics for each epoch are presented in Fig. 8, starting from epoch 0, which describes the state of the network before the initial training. In this state, the network's weights are assigned random values. ΔW is computed starting from epoch 1.

In Fig. 8 (A), the most substantial weight changes, reaching 60 percentage points, occur after the first training epoch. As the number of epochs increases, the magnitude of weight modifications decreases, and around the 15th epoch, ΔW values reach a plateau. A similar pattern is observed in Fig. 8 (B), which illustrates the changes in the S measure as a function of the learning epochs. However, in this case, the plateau is reached earlier, around the 10th epoch. The highest value of S, close to 16, is observed for epoch 0. After approximately five epochs, this value decreases to 2. Both metrics indicate that in the early stages of learning, neurons engage in intense competition, which results in a higher average number of neurons activated in a single document and more significant weight adjustments. As the learning process progresses, neurons become specialized, leading to selective responses to only particular documents and substantially lower weight adjustments. Fig. 8(C) illustrates the evolution of topics quality, represented by TQ_{NPMI}. For this metric, the initial stage of learning is characterized by the rapid TQ increase, which progresses with each subsequent training epoch.

The behavior of the three metrics suggests that the learning process is stable. Based on observation of the metric, the most suitable point to terminate the training process can be determined. For the *BBC news* dataset, we defined that at the 15th epoch. Similarly, the training was stopped after 10 and five epochs for *20Newsgroups* and *AG news* datasets, respectively.

6.2. Topic formation dynamics

This section explores one of the sport-related topics detected in the experiments in more detail. Fig. 9 presents the neuron weights in a heatmap (the upper and lower left corners) and the associated terms in a word cloud (the upper and lower right corners) related to that topic. The figure illustrates two stages of the training process: the start (Fig. 9 (A)), and after 15 training epochs (Figs. 9 (B)).

At the beginning of the training, weights are randomly generated with uniform distribution, which is illustrated by the heatmap in Fig. 9 (A). The world cloud that corresponds to the heatmap depicts random words that do not describe any meaningful (coherent) topic. After 15 training epochs, significant changes can be observed in both the heatmap and the word cloud: the number of very bright weights has decreased significantly, and a large part of the picture has darkened. This corresponds to the sparsity results presented in Fig. 8 (B). Overall, the figure confirms that the neurons have become highly selective and will be most likely activated by documents that contain sport-related words.

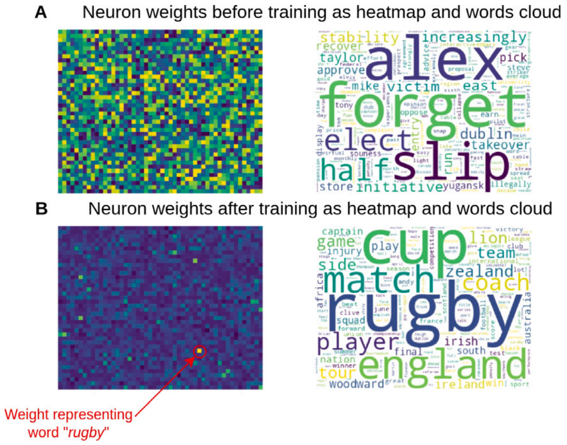

Fig. 9. The changes in the topic weights before (A) and after (B) training, represented as both heatmaps and word clouds. In the heatmap, each point represents a neuron weight, with higher values appearing brighter. These weights correspond to the terms in the word cloud, in which higher weights result in larger font sizes. In section B, the highest-weighted word is *rugby*, indicated by a red arrow.

6.3. Spike limit ns estimation

In the previous experiments, we used a spike limit of 150 spikes, estimated through manual adjustments and observation of the model's performance. We began with a limit of 300 spikes for ten training epochs and gradually decreased the number of spikes until we observed a performance drop for *BBC News* dataset. The spike limit was finally fixed at $ns = 150$. With the fixed ns value, the number of the training epochs was estimated based on the procedure described in Section 6.1, individually for each dataset. This section further investigates how different spike limits affect the model's performance.

We evaluate models performance with $purity$, TQ_{NPMI}, and Silhouette Coefficient (SC). The SC is a clustering quality metric that measures how well a given document fits its cluster compared to others. This indicator does not use the information on class labels like $purity$, so it can be helpful when the dataset is not labeled. SC is defined as follows:

$$SC = \frac{a-b}{max(a,b)} \tag{14}$$

SC is calculated based on the average distance between a document and other documents in the same cluster a and the smallest average distance between the document and points in the nearest other cluster b. Its value ranges from −1 to 1, where 1 indicates a perfect fit (the document belongs to a "strong" cluster), 0 indicates neutrality (the document is on the boundary of clusters), and −1 indicates a poor fit (the document does not fit well in the cluster). The distances between documents are calculated based on the K-dimensional topic representation.

We also estimate TQ_{NPMI} based on each dataset statistics with the Gensim package,[7] which we refer to as internal coherence $TQ_{NPMI-internal}$. This coherence evaluation provides additional insights into how the STM models present word co-occurrences within each dataset. The only difference between the estimation of coherence with Palmetto and Gensim internal evaluation comes from calculating $NPMI$. The Palmetto library evaluates it based on external corpus word co-occurrences (Wikipedia). In case of internal calculation, word co-occurrences are calculated using the same corpus that was used for training.

We trained the models for each dataset three times with varying spike limits ranging from 0 to 180 and $K = 40$. The number of epochs for each dataset was the same as the one previously estimated in Section 6.1. Fig. 10 illustrates the evaluation metrics' mean values and standard deviations across all runs. The values are rescaled to a standard range for better visualization, using the maximum value achieved for each metric across all spike limits on the specific dataset. The figure also includes error bars representing the standard deviations.

6.3.1. AG News dataset

The left plot in Fig. 10 presents the results for the *AG News* dataset. The performance metrics values are rising with an increase of the spike limit ns, achieving maximal values around 90 spikes. When there is no minimal spike limit ($ns = 0$), internally and externally evaluated topic coherence is close to 0, indicating poor topic quality. For low spike limit values, models also perform poorly in clustering tasks. The reason are document lengths in this dataset. In *AG News* the documents are short (see Table 1) and consequently are transformed into a number of spikes which is insufficient for proper learning when the spike limits are low. This showcases that increasing the spike limit is crucial for short documents.

Interestingly, based on this evaluation, the optimal spike limit for this dataset would be $ns = 90$. In our experiments, we used $ns = 150$, which is also a suboptimal value in terms of performance metrics. However, it results in longer training time.

6.3.2. BBC News dataset

The models trained on the *BBC News* dataset (middle plot) show less dependence on the selected spike limit than those trained on the *AG News* regarding performance metrics. The most significant difference between the results is observed in the lower band of the spike limit ns. For *BBC News*, even if the spike limit is 0, the model can deliver moderate quality topics and close to maximal $purity$ scores. The *BBC News*, contrary to *AG News*, contains long documents (cf. Table 1), so reading the documents just once often results in more input spikes than the predefined limit. Because of that, we do not observe significant changes in performance metrics when the spike limit ns varies from 0 to 60 spikes. As ns increases, the SC and topic quality metrics improve and reach optimal values when ns ranges from 120 to 150 spikes.

6.3.3. 20Newsgroups dataset

For the *20Newsgroups* dataset (the right plot in Fig. 10), the performance highly depends on the spike limit ns, and some of the results at first sight can be counterintuitive. Interestingly, the highest topic quality evaluated externally is observed for the lowest ns limit. After further analysis, we discovered that because of the significant imbalances in document length, in a low spike limit regime, the models learn topics mainly from the most extended documents. This leads to potentially informative topics, however, it at the same time affects clustering performance.

The most extended documents are encoded to more spikes because the documents are always read until the last word. On the other hand, short documents are often encoded to the number of spikes equal to the spike limit. It happens when the document length is shorter than the spike limit. With low spike limits, shorter documents may not deliver enough stimuli to activate the topic layer's neurons, resulting in weaker clustering performance.

In case of higher values of spike limit ns, the clustering metrics are higher; however, the topic quality TQ_{NPMI} measured with the Palmetto library decreases. At the same time, internally evaluated topic quality $TQ_{NPMI-internal}$ fluctuates close to the maximal value independently on ns value. This behavior comes from the complexity of the *20Newsgroups* dataset. Documents in this dataset contain many words that co-occur less often than in the reference corpus used by Palmetto software. The records in *20Newsgroups* vary in length and contain headers, footers, cross-posted messages (messages appearing in more than one newsgroup), and a mix of topic-specific and off-topic discussions. Also, the vocabulary of this dataset contains many

[7] https://radimrehurek.com/gensim/

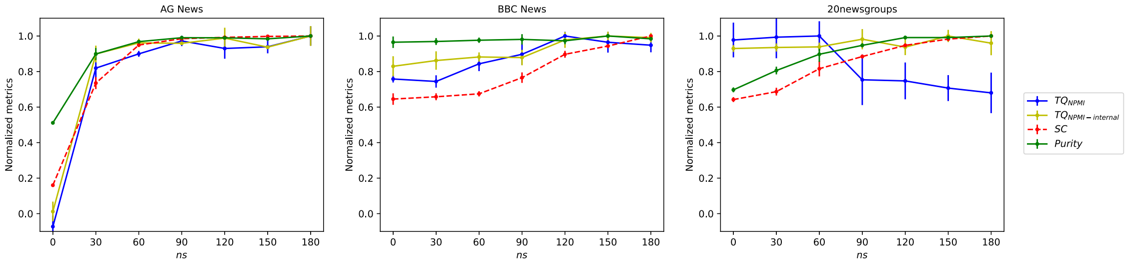

Fig. 10. The changes of models' performance metrics as a function of spike limit ns.

Table 6
TQ: the averaged coherence and diversity for different α values. The best value of α is presented in bold.

α	0	0.0001	0.0005	0.001	**0.005**	0.01	0.05	0.1
C_{NPMI}	0.068	0.070	0.080	0.087	**0.103**	0.082	−0.008	−0.077
PUW	0.531	0.538	0.597	0.638	**0.792**	0.869	0.941	0.905
TQ_{NPMI}	0.036	0.038	0.048	0.055	**0.082**	0.071	0.000	0.000

spelling mistakes and slang words that can decrease the coherence values evaluated on the reference corpus. On the other hand, when topic quality is measured internally with $TQ_{NPMI-internal}$, the STM models correctly capture the topics present in the dataset. It highlights the potential of internal evaluation for domain-specific datasets and those with higher noise.

6.3.4. Summary

The above analysis shows that selecting the spike limit is essential to fine-tuning the model. The proposed approach relies on internally evaluated topic quality and can be adopted to estimating the spike limit of the model also for other datasets. Topic coherence evaluation performed internally requires significantly less time and can indicate how well the STM detects word co-occurrences in the dataset. The SC measure also helps estimate clustering quality when data is not labeled. We recommend selecting a spike limit close to 100–150 for a new dataset. In case of well-balanced datasets containing long, similar documents, setting a spike limit may not be necessary. On the contrary, setting a spike limit is essential for the datasets containing short documents so as to deliver enough input stimuli to the network. It is also recommended when working with imbalanced datasets.

6.4. Text encoding parameters

6.4.1. Word frequency and encoding

One of the model's key parameters is the α coefficient in probability function (1). To illustrate its impact on blue, i.e. the TQ of the models, we trained several STM models on the *BBC News* dataset, each with a different α. The TQ measures for selected α values are reported in Table 6. For each α, the metric values are the averages of five independently trained 40-neuron models ($K = 40$).

The quality of the topics expressed by the TQ_{NPMI} measures depends heavily on the value of α selected. Along with its increase, a gradual improvement in TQ is observed, with the best results in the studied range of α achieved for $\alpha = 0.005$. Further increases in this parameter result in declines in TQ. Partial measures that contribute to TQ_{NPMI} have a significant impact on the variability of the TQ.

For high values of α, low values of C_{NPMI} can be observed. For lower α values, the metric that indicates topic diversity (PUW) performs the worst. This can be explained by the influence of the probability function on the encoding of text into spike representation.

The probability function determines whether a word from a sequence should be converted into an impulse or skipped. If α approaches 0, the probability function values (Eq. (1)) approach 1. Thus, for low values of α, words that appear frequently in the dataset and simultaneously carry low informational value more frequently appear in the impulse representation.

Since HL favors the synaptic connections that more often result in neuron activation, weights that represent frequently occurring words in the dataset are stronger and more often belong to the pool of the neuron weights that define the topic. This, in turn, results in a narrower diversity of words that form the topics and discrimination of the themes represented by less frequently occurring words. For example, the three most common words that occur in the topics extracted from the models trained with $\alpha = 0$ are *year*, *make*, and *people*. The word *year* occurs in 20 topics, *make* in 18, and *people* in nine. Contrarily, in the model trained with $\alpha = 0.005$, the three most frequent words are *film*, *labour*, and *champion*, each occurring in four topics only. The identification of the optimal value of α can be done by observing changes in topic coherence metrics.

6.4.2. Comparison of sparse and dense representation

Text-to-spike encoding is a challenging task due to the complexity of the text data. This stems from the diversity of the documents' lengths and the datasets explained in Section 4.1. We now investigate how the design choices of our method can address these difficulties. To do so, we perform an ablation study on the design choices to examine their influence on the resulting topic models. We study two approaches, both with a minimum spike limit ns= 150. In both cases each document is read with repetitions until a minimum of 150 spikes is reached, but also each document is read in its entirety, i.e. till its end is reached.

1. **Sparse representation**. A single reading might achieve a spike count greater than 150 for very long documents. Additionally, since we allow for sparse representation, if a word read at time t is excluded from the representation by the probability function (Eq. (1)) evaluation, no spike is recorded at time t.
2. **Dense representation**. For very long documents, a single reading might achieve a spike count greater than 150. Furthermore, since we test dense representation, if a word read at time t is excluded from representation by the probability function (Eq. (1)) evaluation, we evaluate the probability function of the next word in the sequence. In this encoding, exactly one spike is emitted at each time step t.

Experiments were conducted on the *20Newsgroups* dataset with a 20-neuron neural network ($K = 20$) for two different values of α. We trained the network for 10 epochs. The model's performance was evaluated using TQ_{NPMI} and *purity* metrics. Additionally, during the

Table 7
The results for different sequential encoding strategies on the *20Newsgroups* dataset with a 20-neuron neural network. The best strategy (the one used in the remainder of the article) is presented in bold. The values in brackets indicate standard deviation. The *purity* values are expressed as percentages.

Metric	Sparse encoding	Dense encoding	Sparse encoding	Dense encoding
	α = 0.0004		α = 0.004	
purity	54.72 (2.47)	**60.00 (1.59)**	44.21 (1.89)	57.11 (4.8)
TQ_{NPMI}	0.059 (0.001)	**0.056 (0.001)**	−0.048 (0.015)	−0.012 (0.008)
Time [s]	1062 (5)	**974.84 (2)**	1523 (21)	960 (4)
Percent of documents	0(0)%	**0(0)%**	4.81(1.14)%	0 (0)

experiments, we observed the activity of neurons in the topic layer. The observation of activity sought to ascertain whether documents encoded through various methods provide sufficient stimulus to activate topic neurons. We report the percentage of documents that did not cause any network activity. The values reported in Table 7 are averages from three independent runs.

The left section of Table 7 presents the results for $\alpha = 0.0004$. The models trained with sparse and dense encodings achieve similar results of topic quality. The more significant differences between models' performance are observed for the *purity* measure, where dense encoding has a substantial advantage in clustering performance over the sparse approach. Also, the training time for dense encoding is shorter.

This phenomenon is more vivid when looking at the results for the higher α value, presented in the right section of the table. There is a significant difference in the training time and clustering performance between the models trained with sparse and dense representation. The longer processing time of the sparsely encoded documents is caused by the higher value of the α parameter, whose application increases the sparsity, so the document needs to be read longer to satisfy the spike limit of 150 spikes. Additionally, for the sparse representation, 4.81% of the inputs do not elicit any neuronal activity in the topic layer, even though each input is encoded to 150 spikes. The reason of this drawback is the low input rate, which decreased to 0.26 spikes per millisecond on average when α was set to 0.004.

In the STM, we use dense representation to minimize the model's complexity and avoid additional drawbacks of sparse document encoding mentioned above. The main difference between these two approaches comes from the additional dependence between the α parameter and the input spike rate, which is introduced to the model if the sparse approach is adopted. In such cases, training the model takes longer time and its effect is less predictable. Selecting a sparse approach would require additional mechanisms to address these issues.

6.5. Network parameters

6.5.1. Neural network dynamics

The initial design of the model was guided by biological principles, employing parameters that closely resemble biological constants. For instance, the rest membrane potential $u_{rest} = -65$ mV in Eq. (2) is in the range of observed biological values (McKhann et al., 1997). Similarly, excitatory membrane potential $u_{exc} = 0$ mV is a common choice in the application of the LIF neural model (Gerstner et al., 2014), as well as the membrane time constant ($\tau = 100$ ms) (Diehl & Cook, 2015). The value of the inhibitory resting potential in our setup is $u_{inh} = -90$ mV. The chosen resting membrane potentials values are conclusive for LIF neuron dynamics. These potentials ensure that the excitatory potential (u_{exc}) is greater than the resting potential (u_{rest}), which in turn is greater than the inhibitory potential (u_{inh}). Finally, the threshold value for our LIF neurons is set to $u_{th} = -55$ mV.

The pace of the inhibitory and excitatory currents' decay are governed by the time constants τ_i and τ_e from Eqs. (3). These values are usually set to several milliseconds. To assess the sensitivity of these parameters, we conducted preliminary studies which involved temporarily changing particular parameters and observing the resulting impact on the TM performance. We tested both parameters in the range of 1–10 ms and observed their cumulative impact on TM. Low values of τ_e (close to 1 ms) significantly reduce neuron's activity, leading to slow or no learning. Values above 5–10 ms yielded sufficient neuron's activity for proper model functioning. Ultimately, we set τ_e to 10 ms. On the other hand, low values of τ_i affect neuron's competition due to the fast decay of inhibitory current. Finally, we set $\tau_i = 5$ ms.

The above values of parameters comprised the initial setup in all experiments conducted in this research. This setup was not fine-tuned for particular tasks or datasets. It is also the recommended initial setup for the future use of the proposed method or encoding.

6.5.2. Plasticity parameters

The learning window in the STM model is defined by the parameter τ_+ in Eq. (5) which governs the neuron's memory of the previous signals that reached the synapse: the higher the value of τ_+ the more expansive a time window. During the experiments, we observed that for τ_+ of an order of a few milliseconds, the quality of the topics was low. When τ_+ reached several tens of milliseconds, TQ reached its highest level. Such observations suggest that broader learning windows that encompass more significant segments of the processed text have a beneficial impact on the quality of the resultant topics. For all datasets, the value of τ_+ was ultimately set to 50 ms. The second parameter defining the learning process is τ_r from Eq. (5), which defines the pace of the scaling factor decay. Larger values of τ_r enforce stronger scaling. From the ablation study presented below, in which clusters *purity* and TQ_{NPMI} are presented as heatmaps, it can be seen that for higher values of both parameters, the STM model scores higher. Selecting higher values of τ_r is usually the safer choice (see Fig. 11).

The last parameter is the learning rate η in Eq. (4), whose value should be adapted for each dataset. η influences both the speed and quality of learning. Relatively high values ($\eta > 0.01$) accelerate learning but increase the likelihood of the network getting stuck in local minima and introduce more significant result variability. For the datasets under consideration, stable learning is observed for $\eta = 0.0006$ (*BBC News*) and $\eta = 0.0003$ for the *20Newsgroups* and *Ag News* datasets. To set this parameter for a new dataset, we recommend using the procedure described in Section 6.1.

7. GPU implementation

To enhance the efficiency of the STM and investigate its integrity with deep learning solutions, we propose its implementation based on the *snnTorch* framework (Eshraghian et al., 2023). The GPU version of STM (STM-G) utilizes a simplified LIF neuron model with the following membrane potential $u(t)$:

$$u(t) = \beta \cdot u(t-1) + \sum_i S_i(t) \cdot w_i(t) - I_{inh}(t) \tag{15}$$

In Eq. (15) β is the coefficient that defines the magnitude of membrane potential decay, $S_i(t)$ represents input layer spike trains, $w_i(t)$ is the neuron weight value connected with the *i-th* input neuron, and $I_{inh}(t)$ is constant inhibitory current. When a neuron in the topic layer is activated, the inhibitory current is sent to all neurons, including the activated one. This mechanism is analogous to that proposed by Białas and Mańdziuk (2022). Additionally, neuronal competition is facilitated

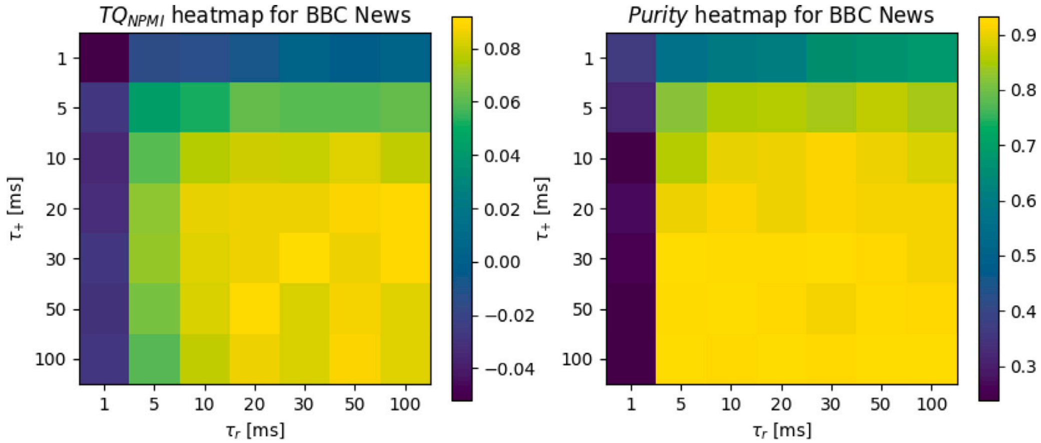

Fig. 11. Different values of TQ_{NPMI} and $purity$ as functions of the learning window τ_+ and scaling decay time constant τ_r.

Table 8
A comparison of the average results between STM and STM-G models calculated for the *BBC News*, *20Newsgroups*, and *AG News* datasets. The values in brackets indicate standard deviation. The best results in each group are presented in bold. The $purity$ values are expressed as percentages.

		20Newsgroups		BBC News		AG News	
		STM	STMG	STM	STMG	STM	STMG
C_{NPMI}	avg	**0.061 (0.013)**	0.052 (0.009)	**0.110 (0.007)**	0.108 (0.008)	**0.117 (0.003)**	0.109 (0.001)
PUW	avg	0.785 (0.042)	**0.817 (0.034)**	0.785 (0.051)	**0.841 (0.052)**	0.912 (0.030)	**0.921 (0.028)**
TQ_{NPMI}	avg	**0.048 (0.013)**	0.043 (0.009)	0.087 (0.011)	**0.090 (0.011)**	**0.107 (0.006)**	0.100 (0.003)
$purity$	avg	60.99 (0.77)	**62.78 (1.10)**	92.05 (0.063)	**92.60 (0.60)**	81.08 (0.17)	**81.22 (0.40)**

through an inhibition mechanism available in the *snnTorch* library. In effect, at time step t only one neuron can be activated, i.e. the one whose membrane potential is the highest among all neurons in the topic layer, and which simultaneously exceeds the threshold value. The learning rule itself (proposed in Section 3) is not altered. The primary distinction lies in its capability for integration with gradient descent methods, analogous to the *STDP-Learner* solution introduced in *SpikingJelly* (Fang et al., 2023).

In the GPU implementation, the model can process several documents in a single time step *t*. The documents are organized in batches, each containing N_B documents. For each document in the batch, a separate synaptic state is kept in the GPU memory. The state contains synaptic traces from Eqs. (5) and (6). If at a given time step *t*, a topic neuron T_j wins the competition with other neurons for B_k different documents, its weights are modified according to the equation:

$$\Delta\vec{w}_j(t) = \sum_{d=1}^{B_k} \Delta\vec{w}_d \tag{16}$$

where $\Delta\vec{w}_d$ is a weight update based on the synaptic state register for dth document, calculated according to Eq. (4).

In the GPU processing, the representation is further optimized by means of compressing, to reduce the number of steps. If the representation is compressed n times, spikes from the n subsequent time steps are merged. After applying the above n–time compression, in each time step there can be n spikes, whereas in the original representation, there is only one spike per time step. Fig. 12 illustrates the 2-time representation compression. Our experiments revealed that compressing representation 3 to 5 times, depending on the dataset, results in good performance while reducing the processing time significantly.

7.1. Performance evaluation on GPU

Table 8 compares the STM and STM-G models. These are the average values of the metrics studied, using the same configurations of neuron count and iterations as in the STM results presented in Section 4.

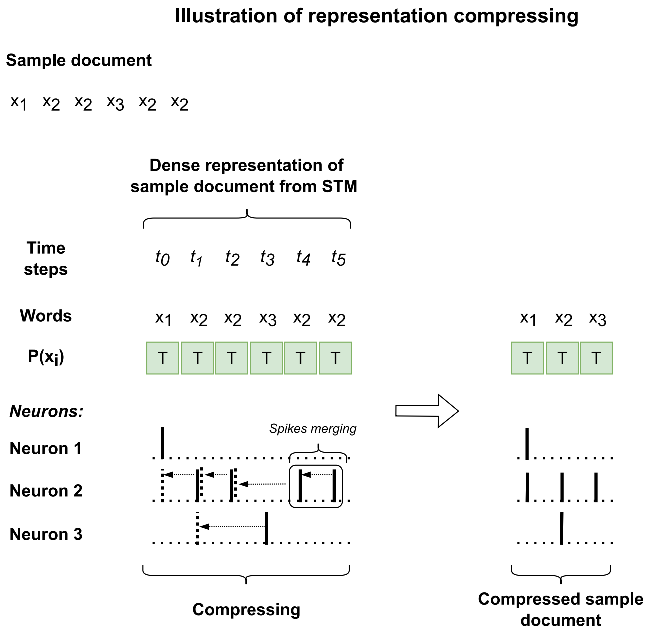
Illustration of representation compressing
Sample document
x_1 x_2 x_2 x_3 x_2 x_2
Dense representation of sample document from STM
Time steps
t_0 t_1 t_2 t_3 t_4 t_5
Words
x_1 x_2 x_2 x_3 x_2 x_2
P(x$_i$)
T T T T T T
Neurons:
Spikes merging
Neuron 1
Neuron 2
Neuron 3
Compressing
x_1 x_2 x_3
T T T
Compressed sample document

Fig. 12. 2-time representation compression suitable for the GPU processing.

The results show that STM achieves higher C_{NPMI} coherence values for the *20Newsgroups*, *BBC News* and *AG News* datasets. At the same time, for all datasets, the STM-G model achieves higher $purity$ and PUW. The differences between the models are primarily attributed to the different data processing methods. In the case of STM-G, data is processed in batches (N_B = 2000), where at each time t, weights are modified based on multiple documents simultaneously. Additionally, in the learning process of STM-G, the representation undergoes compression: 3-fold for the *BBC News* dataset and 5-fold for the remaining two datasets. The TQ_{NMPI} scores show little difference between the models, since higher PUW values for the STM-G model balance out the slightly lower coherence scores. Finally, during manual analysis of topics' listings, we struggled to see clear differences between the models.

The biggest advantage of STM-G is shorter processing time. The training time on consumer GPU card RTX 2070 for *AG News* takes 7 minutes, compared to 40 minutes required to train STM. The memory usage for STM-G for processing batches with 2000 documents is 4.8 GB. The model could additionally benefit from the parallel computation on more than one GPU unit. With such an approach, STM-G could be adapted to process large data collections.

8. Conclusions

The application of SNNs in NLP is an underexplored, though significant, area. This article demonstrates how SNNs trained in an unsupervised, biologically plausible manner can be applied to TM effectively. It also demonstrates how a sequence of words (e.g. a sentence) can be transformed into spikes and further processed by an SNN.

The quantitative evaluation of the STM results confirms that the proposed SNN can detect highly coherent and diverse topics. The results of the comparative analysis demonstrate that the method can compete successfully with other widely used TM algorithms. The qualitative analysis of example topics reveals that even those that are low-ranked according to the coherence score can offer valuable insights regarding the processed datasets. The results emphasize the need to integrate quantitative and qualitative assessments in the evaluation of topic models. A holistic approach yields a more comprehensive understanding of the model's performance and offers promise for potential applications across diverse datasets and contexts.

This paper also discusses significant aspects of STM dynamics, including the process of topic formation and the convergence of the learning process. It demonstrates how analysis of neuron activity and the pace of weight modifications, along with simultaneous observation of the topics' quality, enable the most suitable number of training epochs to be determined. Our studies indicate that the most intense changes in synaptic connections occur during the first few epochs of the learning process. During this period, neurons are significantly more active, which results in their greater mutual competition. In the final stages of learning, the activity of the neurons stabilizes. Ultimately, only a few neurons are activated during the presentation of a single document.

Another important aspect of this work concerns the impact of word frequency on the STM's performance. In the proposed text-to-spike transformation, a text document is represented as a sequence of words, before being transformed into spike trains in the order they occur in the sequence. The probability of a single word's transformation into a spike depends on the frequency of the word's occurrence in the document corpora. We demonstrate that a mechanism that regulates the probability of a word-to-spike transformation is crucial in the development of high-quality topics. Without such a mechanism, Hebbian learning favors frequently occurring words that become over-represented in the topics. This, in turn, affects the diversity of the topics noticeably and decreases the ultimate quality of the learned topics.

Another critical aspect of the proposed method is determining the spike limit for the dataset. In the analysis presented in Section 6.3, we emphasize the importance of selecting the spike limit to optimize model performance. Setting a spike limit may be unnecessary for well-balanced datasets with lengthy, similar documents. Conversely, setting a spike limit for datasets containing shorter documents is crucial to ensure sufficient input stimuli for the network.

Finally, we showed how the model can be tailored for text processing using contemporary graphics cards. This adaptation notably diminishes the learning time because the model can process documents simultaneously.

A natural continuation of the presented solution is further exploration of the STM in the context of unsupervised natural language processing. The model's promising performance in terms of topic modeling and clustering demonstrates the significant potential of this approach. Possible STM development avenues include advancing the model towards hierarchical clustering and topic modeling. Another interesting modification would be to leverage the probability function to select more efficient words for text classification by utilizing feature selection algorithms (Bahassine et al., 2020; Labani et al., 2018; Wang & Lin, 2019).

CRediT authorship contribution statement

Marcin Białas: Writing – review & editing, Writing – original draft, Software, Investigation, Conceptualization. **Marcin Michał Mirończuk:** Writing – review & editing, Writing – original draft, Software, Conceptualization. **Jacek Mańdziuk:** Writing – review & editing, Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Appendix. Description of comparative models

Analogously to the STM models, for each of the methods, 5 independent models were trained for each topic configuration. The results presented in the paper are the (arithmetic) mean outcomes of these 5 executions.

The LDA method (Blei et al., 2003) is one of the most popular models used in the extraction of topics from document corpora. Its underlying assumption is that each document is a mixture of hidden topics with probability distribution ϕ, and each hidden topic is a mixture of words from the corpus with probability distribution ψ. Learning aims to find the probability distributions of both mixtures. When that goal is achieved, documents can be represented by the most probable topics, and topics can be interpreted based on the most probable words. Generally, using an additional assumption that ϕ has asymmetric distribution and ψ is symmetric (Wallach et al., 2009) results in better performance. In the experiments, we used the *gensim* (Rehurek & Sojka, 2011) implementation of the LDA model, which follows this assumption. We tested two asymmetric configurations for ϕ: *asymmetric* and *auto* (Rehurek & Sojka, 2011). Better results were achieved with the *auto* configuration; thus, the results are reported only for that configuration. For more details, see the *gensim* documentation[8] and the related source code of the method.

The BTM model is designed explicitly for short-text topic modeling and directly models word co-occurrence patterns (biterms) in a corpus of text. Cheng et al. (2014) defined a *biterm* as an unordered word pair that co-occurs in a short context (e.g., a small, fixed-size window over a term sequence in a document). BTM assumes that the two words in a *biterm* are associated with the same topic, which is represented as a distribution of words. Other topic models, such as LDA, work similarly. BTM has several advantages over other topic models, including that: (1) it explicitly models word co-occurrence patterns to improve topic learning; and (2) it uses aggregated patterns across the entire corpus to discover topics, which avoids the issue of sparse patterns at the

[8] https://radimrehurek.com/gensim/models/ldamodel.html

document level. Our experiments used a Cython-based implementation of BTM (*Bitermplus*[9]).

ETM (Dieng et al., 2020) models each word in a document as a categorical distribution over topics. The model can be trained efficiently and applied to large corpora using a variational inference algorithm. Additionally, ETM can deal well with stop and rare words: it filters them out from the topics effectively. ETM has a plethora of configuration parameters. In the experiments, we consider the main ones, called *epochs*, *embeddings*, and *train embeddings*. The *epoch* parameter controls the duration of the learning phase. The *embeddings* parameter contains the word-to-vector representation constructed based on the corpora. The *train embeddings* parameter is a flag that signals whether to fix the model parameter called *rho* or to train it. A Python-based implementation of ETM[10] was used in the experiments.

BERTopic (Grootendorst, 2022) generates document embeddings with pretrained transformer-based language models, clusters the embeddings, and generates topic representations using the class-based TF-IDF procedure. Since BERTopic cannot generate a specified, predefined number of topics, as all the above-summarized methods, we trimmed the number of topics to the values selected in the paper. The experiments were run using a Python-based implementation of BERTopic.[11]

[9] https://bitermplus.readthedocs.io/en/latest/

[10] https://github.com/lffloyd/embedded-topic-model

[11] https://github.com/MaartenGr/BERTopic

References

Abdelrazek, A., Eid, Y., Gawish, E., Medhat, W., & Hassan, A. (2023). Topic modeling algorithms and applications: A survey. *Information Systems*, *112*, Article 102131. http://dx.doi.org/10.1016/j.is.2022.102131, URL https://linkinghub.elsevier.com/retrieve/pii/S0306437922001090.

Aggarwal, C. C., & Zhai, C. (2012). A survey of text clustering algorithms. In C. A. C. C., & Zhai (Eds.), *Mining text data* (pp. 77–128). Boston, MA: Springer US, http://dx.doi.org/10.1007/978-1-4614-3223-4_4, URL http://link.springer.com/10.1007/978-1-4614-3223-4_4.

Alenezi, T., & Hirtle, S. (2022). Normalized attraction travel personality representation for improving travel recommender systems. *IEEE Access*, *10*, 56493–56503. http://dx.doi.org/10.1109/ACCESS.2022.3178439.

Aletras, N., & Stevenson, M. (2013). Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th international conference on computational semantics* (pp. 13–22). URL https://aclanthology.org/W13-0102.pdf.

An, Y., Kim, D., Lee, J., Oh, H., Lee, J.-S., & Jeong, D. (2023). Topic modeling-based framework for extracting marketing information from E-commerce reviews. *IEEE Access*, *11*, 135049–135060. http://dx.doi.org/10.1109/ACCESS.2023.3337808.

Asnawi, M. H., Pravitasari, A. A., Herawan, T., & Hendrawati, T. (2023). The combination of contextualized topic model and MPNet for user feedback topic modeling. *IEEE Access*, *11*, 130272–130286. http://dx.doi.org/10.1109/ACCESS.2023.3332644.

Bahassine, S., Madani, A., Al-Sarem, M., & Kissi, M. (2020). Feature selection using an improved Chi-square for Arabic text classification. *Journal of King Saud University - Computer and Information Sciences*, *32*(2), 225–231. http://dx.doi.org/10.1016/j.jksuci.2018.05.010, URL http://dx.doi.org/10.1016/j.jksuci.2018.05.010.

Białas, M., & Mańdziuk, J. (2022). Spike-timing-dependent plasticity with activation-dependent scaling for receptive fields development. *IEEE Transactions on Neural Networks and Learning Systems*, *33*, 5215–5228. http://dx.doi.org/10.1109/TNNLS.2021.3069683, URL https://ieeexplore.ieee.org/document/9400417/.

Białas, M., Mirończuk, M. M., & Mańdziuk, J. (2020). Biologically plausible learning of text representation with spiking neural networks. In T. Bäck, M. Preuss, A. H. Deutz, H. Wang, C. Doerr, M. T. M. Emmerich, & H. Trautmann (Eds.), *vol. 12269*, *Parallel problem solving from nature - PPSN XVI - 16th international conference, PPSN 2020, Leiden, the Netherlands, September 5-9, 2020, proceedings, part I* (pp. 433–447). Springer, http://dx.doi.org/10.1007/978-3-030-58112-1_30, URL http://link.springer.com/10.1007/978-3-030-58112-1_30.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Brosseau-Villeneuve, B., Nie, J. Y., & Kando, N. (2014). Latent word context model for information retrieval. *Information Retrieval*, *17*(1), 21–51. http://dx.doi.org/10.1007/s10791-013-9220-9.

Chauhan, U., & Shah, A. (2021). Topic modeling using latent Dirichlet allocation: A survey. *ACM Computing Surveys*, *54*(7), http://dx.doi.org/10.1145/3462478.

Chauhan, U., & Shah, A. (2022). Topic modeling using latent Dirichlet allocation. *ACM Computing Surveys*, *54*, 1–35. http://dx.doi.org/10.1145/3462478, URL https://dl.acm.org/doi/10.1145/3462478.

Chen, Z., & Liu, B. (2017). Topic models for NLP applications. In C. Sammut, & G. I. Webb (Eds.), *Encyclopedia of machine learning and data mining* (pp. 1276–1280). Boston, MA: Springer US, http://dx.doi.org/10.1007/978-1-4899-7687-1_906.

Cheng, X., Yan, X., Lan, Y., & Guo, J. (2014). BTM: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, *26*, 2928–2941. http://dx.doi.org/10.1109/TKDE.2014.2313872, URL http://ieeexplore.ieee.org/document/6778764/.

Corso, G. M. D., Gullí, A., & Romani, F. (2005). Ranking a stream of news. In *Proceedings of the 14th international conference on world wide web* (p. 97). New York, New York, USA: ACM Press, http://dx.doi.org/10.1145/1060745.1060764, URL http://portal.acm.org/citation.cfm?doid=1060745.1060764.

Deng, X., Li, Y., Weng, J., & Zhang, J. (2019). Feature selection for text classification: A review. *Multimedia Tools and Applications*, *78*, 3797–3816. http://dx.doi.org/10.1007/s11042-018-6083-5, URL http://link.springer.com/10.1007/s11042-018-6083-5.

Diehl, P. U., & Cook, M. (2015). Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, *9*, http://dx.doi.org/10.3389/fncom.2015.00099, URL http://dx.doi.org/10.3389/fncom.2015.00099.

Diehl, P. U., Zarrella, G., Cassidy, A., Pedroni, B. U., & Neftci, E. (2016). Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware. In *2016 IEEE international conference on rebooting computing* (pp. 1–8). http://dx.doi.org/10.1109/ICRC.2016.7738691.

Dieng, A. B., Ruiz, F. J. R., & Blei, D. M. (2020). Topic modeling in embedding spaces. *Transactions of the Association for Computational Linguistics*, *8*, 439–453. http://dx.doi.org/10.1162/tacl_a_00325, URL https://aclanthology.org/2020.tacl-1.29.

Eshraghian, J. K., Ward, M., Neftci, E., Wang, X., Lenz, G., Dwivedi, G., Bennamoun, M., Jeong, D. S., & Lu, W. D. (2023). Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, *111*(9), 1016–1054.

Facchinetti, T., Benetti, G., Giuffrida, D., & Nocera, A. (2022). Slr-kit: A semi-supervised machine learning framework for systematic literature reviews. *Knowledge-Based Systems*, *251*, Article 109266. http://dx.doi.org/10.1016/j.knosys.2022.109266, URL https://www.sciencedirect.com/science/article/pii/S0950705122006335.

Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., Huang, L., Zhou, H., Li, G., & Tian, Y. (2023). SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence. http://dx.doi.org/10.48550/ARXIV.2310.16620, URL https://arxiv.org/abs/2310.16620.

Fritzke, B. (1997). *Some competitive learning methods* (p. 100). Artificial Intelligence Institute, Dresden University of Technology, URL https://demogng.de/papers/sclm.pdf.

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, http://dx.doi.org/10.1017/cbo9781107447615.

Greene, D., & Cunningham, P. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on machine learning* (pp. 377–384). New York, New York, USA: ACM Press, http://dx.doi.org/10.1145/1143844.1143892, URL http://portal.acm.org/citation.cfm?doid=1143844.1143892.

Grootendorst, M. (2022). BERTopic: Neural topic modeling with a class-based TF-IDF procedure. arXiv:2203.05794.

Harris, Z. S. (1954). Distributional structure. *WORD*, *10*, 146–162. http://dx.doi.org/10.1080/00437956.1954.11659520, URL http://www.tandfonline.com/doi/full/10.1080/00437956.1954.11659520.

Hebb, D. (1949). *The organization of behavior: A neuropsychological theory*. New York: Wiley.

Huang, J., Serb, A., Stathopoulos, S., & Prodromakis, T. (2023). Text classification in memristor-based spiking neural networks. *Neuromorphic Computing and Engineering*, http://dx.doi.org/10.1088/2634-4386/acb2f0, URL https://iopscience.iop.org/article/10.1088/2634-4386/acb2f0.

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2019a). Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications*, *78*, 15169–15211. http://dx.doi.org/10.1007/s11042-018-6894-4, URL http://link.springer.com/10.1007/s11042-018-6894-4.

Jelodar, H., Wang, Y., Yuan, C., Feng, X., Jiang, X., Li, Y., & Zhao, L. (2019b). Latent Dirichlet allocation (LDA) and topic modeling: Models, applications, a survey. *Multimedia Tools and Applications*, *78*(11), 15169–15211. http://dx.doi.org/10.1007/s11042-018-6894-4.

Jiang, C., Li, L., Zeng, D. D., & Wang, X. (2023). A character-level short text classification model based on spiking neural networks. In *2023 international joint conference on neural networks*. IJCNN, IEEE, http://dx.doi.org/10.1109/ijcnn54540.2023.10191963.

Jurafsky, D., & Martin, J. H. (2009). *Prentice hall series in artificial intelligence*, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall, Pearson Education International, URL https://www.worldcat.org/oclc/315913020.

Karas, B., Qu, S., Xu, Y., & Zhu, Q. (2022). Experiments with LDA and Top2Vec for embedded topic discovery on social media data—A case study of cystic fibrosis. *Frontiers in Artificial Intelligence*, *5*, http://dx.doi.org/10.3389/frai.2022.948313, URL https://www.frontiersin.org/articles/10.3389/frai.2022.948313.

Karlgren, J., & Sahlgren, M. (2001). *From words to understanding*. CSLI publications.

Kherwa, P., & Bansal, P. (2018). Topic modeling: A comprehensive review. *ICST Transactions on Scalable Information Systems*, Article 159623. http://dx.doi.org/10.4108/eai.13-7-2018.159623, URL http://eudl.eu/doi/10.4108/eai.13-7-2018.159623.

Kim, S., & Yoon, J. (2015). Link-topic model for biomedical abbreviation disambiguation. *Journal of Biomedical Informatics*, *53*, 367–380. http://dx.doi.org/10.1016/j.jbi.2014.12.013, URL https://www.sciencedirect.com/science/article/pii/S1532046414002780.

Kou, G., Yang, P., Peng, Y., Xiao, F., Chen, Y., & Alsaadi, F. E. (2020). Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods. *Applied Soft Computing*, *86*, Article 105836. http://dx.doi.org/10.1016/j.asoc.2019.105836, URL https://linkinghub.elsevier.com/retrieve/pii/S1568494619306179.

Kumar, N., & Hanji, B. R. (2024). Combined sentiment score and star rating analysis of travel destination prediction based on user preference using morphological linear neural network model with correlated topic modelling approach. *Multimedia Tools and Applications*, http://dx.doi.org/10.1007/s11042-023-17995-y.

Labani, M., Moradi, P., Ahmadizar, F., & Jalili, M. (2018). A novel multivariate filter method for feature selection in text classification problems. *Engineering Applications of Artificial Intelligence*, *70*, 25–37. http://dx.doi.org/10.1016/j.engappai.2017.12.014.

Long, L., & Fang, G. (2010). A review of biologically plausible neuron models for spiking neural networks. In *AIAA infotech@aerospace 2010* (p. 1). Reston, Virigina: American Institute of Aeronautics and Astronautics, http://dx.doi.org/10.2514/6.2010-3540, URL https://arc.aiaa.org/doi/10.2514/6.2010-3540.

Lv, C., Xu, J., & Zheng, X. (2023). Spiking convolutional neural networks for text classification. In *The eleventh international conference on learning representations*. URL https://openreview.net/forum?id=pgU3k7QXuz0.

Maciąg, P. S., Sitek, W., Skonieczny, Ł., & Rybiński, H. (2022). A comparative study of short text classification with spiking neural networks. In *2022 17th conference on computer science and intelligence systems* (pp. 79–88). http://dx.doi.org/10.15439/2022F184, URL https://annals-csis.org/proceedings/2022/drp/184.html.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. USA: Cambridge University Press, http://dx.doi.org/10.1017/CBO9780511809071, URL https://www.cambridge.org/core/product/identifier/9780511809071/type/book.

McKhann, G. M., D'Ambrosio, R., & Janigro, D. (1997). Heterogeneity of astrocyte resting membrane potentials and intercellular coupling revealed by whole-cell and gramicidin-perforated patch recordings from cultured neocortical and hippocampal slice astrocytes. *The Journal of Neuroscience*, *17*(18), 6850–6863. http://dx.doi.org/10.1523/jneurosci.17-18-06850.1997.

Murshed, B. A. H., Mallappa, S., Abawajy, J., Saif, M. A. N., Al-ariki, H. D. E., & Abdulwahab, H. M. (2022). Short text topic modelling approaches in the context of big data: Taxonomy, survey, and analysis. *Artificial Intelligence Review*, *56*(6), 5133–5260. http://dx.doi.org/10.1007/s10462-022-10254-w.

Newman, D., Lau, J. H., Grieser, K., & Baldwin, T. (2010). Automatic evaluation of topic coherence. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics* (pp. 100–108).

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Cao, H., Cheng, X., Chung, M., Grella, M., GV, K. K., He, X., Hou, H., Kazienko, P., Kocon, J., Kong, J., Koptyra, B., Lau, H., Mantri, K. S. I., Mom, F., ... Zhu, R.-J. (2023). RWKV: Reinventing RNNs for the transformer era. arXiv:2305.13048.

Qiang, J., Qian, Z., Li, Y., Yuan, Y., & Wu, X. (2022). Short text topic modeling techniques, applications, and performance: A survey. *IEEE Transactions on Knowledge & Data Engineering*, *34*(03), 1427–1445. http://dx.doi.org/10.1109/TKDE.2020.2992485.

Rehurek, R., & Sojka, P. (2011). *Gensim–Python framework for vector space modelling: vol. 3*, Brno, Czech Republic: NLP Centre, Faculty of Informatics, Masaryk University.

Röder, M., Both, A., & Hinneburg, A. (2015). Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on web search and data mining* (pp. 399–408). New York, NY, USA: ACM, http://dx.doi.org/10.1145/2684822.2685324, URL https://dl.acm.org/doi/10.1145/2684822.2685324.

Roy, K., Jaiswal, A., & Panda, P. (2019). Towards spike-based machine intelligence with neuromorphic computing. *Nature*, *575*(7784), 607–617. http://dx.doi.org/10.1038/s41586-019-1677-2.

Schuman, C. D., Kulkarni, S. R., Parsa, M., Mitchell, J. P., Date, P., & Kay, B. (2022). Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, *2*(1), 10–19. http://dx.doi.org/10.1038/s43588-021-00184-y.

Sebastian, Y., Siew, E. G., & Orimaye, S. O. (2017). Learning the heterogeneous bibliographic information network for literature-based discovery. *Knowledge-Based Systems*, *115*, 66–79. http://dx.doi.org/10.1016/j.knosys.2016.10.015, URL https://www.sciencedirect.com/science/article/pii/S0950705116303860.

Seilsepour, A., Ravanmehr, R., & Nassiri, R. (2023). Topic sentiment analysis based on deep neural network using document embedding technique. *Journal of Supercomputing*, *79*(17), 19809–19847. http://dx.doi.org/10.1007/s11227-023-05423-9.

Song, S., Miller, K. D., & Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience*, *3*, 919–926. http://dx.doi.org/10.1038/78829, URL http://www.nature.com/articles/nn0900_919.

Stimberg, M., Brette, R., & Goodman, D. F. (2019). Brian 2, an intuitive and efficient neural simulator. *eLife*, *8*, Article e47314. http://dx.doi.org/10.7554/eLife.47314, URL https://elifesciences.org/articles/47314.

Tang, Z., Li, W., Li, Y., Zhao, W., & Li, S. (2020). Several alternative term weighting methods for text representation and classification. *Knowledge-Based Systems*, *207*, Article 106399. http://dx.doi.org/10.1016/j.knosys.2020.106399.

Turrigiano, G. (2012). Homeostatic synaptic plasticity: Local and global mechanisms for stabilizing neuronal function. *Cold Spring Harbor Perspectives in Biology*, *4*, http://dx.doi.org/10.1101/cshperspect.a005736, a005736–a005736. URL http://cshperspectives.cshlp.org/lookup/doi/10.1101/cshperspect.a005736.

Vayansky, I., & Kumar, S. A. (2020a). A review of topic modeling methods. *Information Systems*, *94*, Article 101582. http://dx.doi.org/10.1016/j.is.2020.101582, URL https://linkinghub.elsevier.com/retrieve/pii/S0306437920300703.

Vayansky, I., & Kumar, S. A. (2020b). A review of topic modeling methods. *Information Systems*, *94*, Article 101582. http://dx.doi.org/10.1016/j.is.2020.101582, URL https://www.sciencedirect.com/science/article/pii/S0306437920300703.

Voskergian, D., Bakir-Gungor, B., & Yousef, M. (2023). TextNetTopics Pro, a topic model-based text classification for short text by integration of semantic and document-topic distribution information. *Frontiers in Genetics*, *14*, http://dx.doi.org/10.3389/fgene.2023.1243874, URL https://www.frontiersin.org/journals/genetics/articles/10.3389/fgene.2023.1243874.

Wallach, H., Mimno, D., & McCallum, A. (2009). Rethinking LDA: Why priors matter. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, & A. Culotta (Eds.), *Advances in neural information processing systems*: *vol. 22*, (pp. 1973–1981). Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2009/file/0d0871f0806eae32d30983b62252da50-Paper.pdf.

Wang, Y., Bashar, M. A., Chandramohan, M., & Nayak, R. (2023). Exploring topic models to discern cyber threats on Twitter: A case study on Log4Shell. *Intelligent Systems with Applications*, *20*, Article 200280. http://dx.doi.org/10.1016/j.iswa.2023.200280, URL https://www.sciencedirect.com/science/article/pii/S2667305323001059.

Wang, Z., & Lin, Z. (2019). Optimal feature selection for learning-based algorithms for sentiment classification. *Cognitive Computation*, *12*(1), 238–248. http://dx.doi.org/10.1007/s12559-019-09669-5.

Wang, X., McCallum, A., & Wei, X. (2007). Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Seventh IEEE international conference on data mining* (pp. 697–702). IEEE.

Wang, Y., Zeng, Y., Tang, J., & Xu, B. (2019). Biological neuron coding inspired binary word embeddings. *Cognitive Computation*, *11*(5), 676–684. http://dx.doi.org/10.1007/s12559-019-09643-1.

Yamazaki, K., Vo-Ho, V.-K., Bulsara, D., & Le, N. (2022). Spiking neural networks and their applications: A review. *Brain Sciences*, *12*(7), http://dx.doi.org/10.3390/brainsci12070863, URL https://www.mdpi.com/2076-3425/12/7/863.

Yang, C., Chen, X., Liu, L., & Sweetser, P. (2021). Leveraging semantic features for recommendation: Sentence-level emotion analysis. *Information Processing & Management*, *58*(3), Article 102543. http://dx.doi.org/10.1016/j.ipm.2021.102543, URL https://www.sciencedirect.com/science/article/pii/S0306457321000509.

Zhai, C. (2017). Probabilistic topic models for text data retrieval and analysis. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval* (pp. 1399–1401). New York, NY, USA: ACM, http://dx.doi.org/10.1145/3077136.3082067, URL https://dl.acm.org/doi/10.1145/3077136.3082067.

Zhu, R. J., Zhao, Q., Li, G., & Eshraghian, J. K. (2023). Spikegpt: Generative pre-trained language model with spiking neural networks. arXiv:2302.13939.