Knowledge and Information Systems (2022) 64:2921–2944
https://doi.org/10.1007/s10115-022-01731-3

REGULAR PAPER

Powered Dirichlet–Hawkes process: challenging textual clustering using a flexible temporal prior

Gaël Poux-Médard[1] · **Julien Velcin**[1] · **Sabine Loudcher**[1]

Received: 16 December 2021 / Revised: 11 July 2022 / Accepted: 16 July 2022 /
Published online: 1 August 2022

Abstract
The textual content of a document and its publication date are intertwined. For example, the publication of a news article on a topic is influenced by previous publications on similar issues, according to underlying temporal dynamics. However, it can be challenging to retrieve meaningful information when textual information conveys little information or when temporal dynamics are hard to unveil. Furthermore, the textual content of a document is not always linked to its temporal dynamics. We develop a flexible method to create clusters of textual documents according to both their content and publication time, the powered Dirichlet–Hawkes process (PDHP). We show PDHP yields significantly better results than state-of-the-art models when temporal information or textual content is weakly informative. The PDHP also alleviates the hypothesis that textual content and temporal dynamics are always perfectly correlated. PDHP retrieves textual clusters, temporal clusters, or a mixture of both with high accuracy. We demonstrate that PDHP generalizes previous work –the Dirichlet–Hawkes process (DHP) and uniform process (UP). Finally, we illustrate the changes induced by PDHP over DHP and UP with a real-world application using Reddit data. We detail how PDHP recovers bursty dynamics and show that its limit case accounts for daily and weekly publication cycles.

Keywords Clustering · Temporal Bayesian prior · Powered Dirichlet process · Hawkes process

1 Introduction

Online information is generated at an unprecedented rate. Every minute, 500,000 comments are posted on Facebook, 400 h of videos are uploaded on YouTube, and 500,000 tweets are published on Twitter. A possible approach to make sense out of this mass of information is to cluster publication events together. Grouping similar publications together help understanding topics of interest or generate summaries of daily news. Many clustering algorithms

✉ Gaël Poux-Médard
gael.poux@yahoo.fr

[1] ERIC Lab, Université de Lyon, 69361 Lyon, France

are based on text similarity, that is, how similar the words of two published documents are Blei et al. [6], Rathore et al. [19] and Bahdanau e al. [3]. Another relevant variable to group information together is the time of publication [5, 9]. For example, two news articles about forest fires might be unrelated if the second article were published years after the first one despite a close lexical similarity. Imagine a news website that publishes a series about history every day at midday. Temporal dynamics would help understand that a publication the next day at midday is likely to be related to previous publications, even if the story (and thus the vocabulary) is different.

Many models that aim at understanding the temporal dynamics of clusters work by selecting a subset of observations according to a temporal sampling function [1, 5, 25]. However, sampling observations in time implies defining a sampling function that might not correctly model the temporal dynamics at stake. Besides, these works are based on a Dirichlet prior (DP) for clustering. The DP considers counts as a parameter, where a document always counts for 1. It has been argued that such modeling is not fit to account for the arrival of documents in continuous-time settings. In [8], the authors combine techniques of standard textual clustering with point processes. The idea is to infer the time-sampling function parameters as well as the rest of the model. Explicitly, they derive the Dirichlet–Hawkes process (DHP) prior for documents cluster allocation that takes time as a parameter and yields non-integer counts. It has been argued that this method cannot handle limited cases where text is less informative (e.g., short texts, overlapping vocabularies) [25].

Our present work develops the Powered Dirichlet Hawkes process (PDHP) that has previously been introduced in [18], as a mean to handle this case. Besides, we highlight other limiting cases in which DHP fails whereas PDHP yields good results, for instance when temporal information conveys little information (overlapping Hawkes intensities, few observations). We also show there are cases where documents within a textual cluster do not follow the same temporal dynamics, which the DHP is not designed to handle. For instance, an article published by a popular newspaper is unlikely to have the same influence on subsequent similar articles (temporal dynamics) as the same article published by a less popular newspaper. We overcome all these limitations by developing the Powered Dirichlet–Hawkes process, which yields better results than DHP on every dataset considered (up to $+0.3$ NMI). It also allows us to distinguish textual clusters from *temporal clusters* (documents that follow the same dynamic independently from their content). Finally, we conduct large scale experiments on real-world datasets from Reddit.

As an extension to our previously published paper [18], we refine the analysis of PDHP's outcome on news subreddits. In particular, we explicitly show and discuss the inferred timelines inferred by PDHP. We observe the role of r from this perspective, which provides us with heuristics on how to determine it; these heuristics are discussed in an additional paragraph. Typically, we show that our method allows to recover more or less bursty events from real-world data streams by controlling r. Finally, we show that in its limit case where r is large, PDHP accounts for daily and weekly publication dynamics.

Our contributions are listed below:

- We highlight and explain the limitations of the DHP prior: it does not handle weakly informative temporal and textual information and it is not designed to consider different dynamics between text and time.
- We derive the powered Dirichlet Hawkes process (PDHP) as a new prior in Bayesian nonparametric for the temporal clustering of a stream textual documents, which is a generalization of the Dirichlet–Hawkes process (DHP) and of the uniform process (UP).

– We show how the PDHP prior performs better than DHP and UP priors through thorough evaluation and comparison on several synthetic datasets and real-world datasets from Reddit.
– We show that PDHP prior allows to select the information clusters are based on; we choose to favor their generation more according to documents' textual content or temporal dynamics.

2 Background

2.1 How publication times carry valuable information

Before reviewing existing methods incorporating a temporal dimension into text clustering, we detail how this information is relevant to the task. Recent works on the online spread of textual documents have highlighted several key properties regarding the link between textual content and date of publication.

First, it has been shown that textual documents do not get published independently one from the other. Often, the arrival of a document is conditional on the publication of earlier documents. A straightforward illustration is that a new research paper is built on previous publications and is likely to treat a similar topic; the present article exists because of all the references it cites. A 2012 research paper highlights the critical role played by interactions in the re-publication of a tweet on Twitter [14]. The authors claim the probability of retweets vary by 71% on average when considering temporal interactions. More recent works find that although the interaction between publications plays a significant role in later publications, the interaction matrix is often sparse [16]—an article on textual clustering is more likely to appear conditional to publications about NLP, whose vocabulary is only a small subset of the scientific literature's one. It highlights the need to cluster words together to retrieve temporal interaction relevant to a textual clustering problem. In this context, a cluster should carry information about the interaction between the documents it contains.

Second, a problem that arises is the temporal aspect of interaction. It has been shown that online information interaction decays quickly with time [7, 15]. Although the rate at which interaction influence decays depends on the dataset, it seems to fade rapidly for most online spreading processes [10]. To keep the temporal information relevant, clusters must depend on time. For example, two series of news articles about vaccines might not be related (one might not trigger the other) if one was published in 2010 and the other in 2021; they are two different clusters since both obey their own dynamics, although their vocabulary is similar.

2.2 Temporal clustering of textual documents

The use of temporal dimension in documents clustering has been studied on many occasions; a notable spike of interest happened in 2006. Many authors tackled the problem of inferring time-dependent clusters from models based on LDA [5, 11, 22]. However, most of these models are parametric, meaning the number of clusters is fixed at the beginning of the algorithm. Depending on the considered time range and the dataset, the number of clusters needs to be fine-tuned with several independent runs, making them hardly usable for many real-world applications. In all three references cited, the authors mention that a nonparametric version of the model might be derivable.

Ahmed et al. proposed the Recurrent Chinese Restaurant Process (RCRP) as an answer to this problem [1]. Instead of considering a fixed-size dataset, this model can handle a stream of documents arriving in chronological order, and the number of clusters is automatically updated. In this model, time is split into episodes to capture the temporal aspect of cluster formation; it considers an integer count of publications within a given time window. A later version of the model from 2010, the Distance-Dependent Chinese Restaurant Process (DD-CRP) tries to alleviate this approximation by replacing fixed-time episodes with a continuous-time sampling function [4]. However, the model still considers integer counts with only their distribution over time changing. Thus, the model is not designed to consider every temporal information in a continuous-time setting.

Du et al. answered this problem by combining the Dirichlet process with the Hawkes process, used to model the appearance of events in a continuous-time setting. The key idea is to replace the counts of a Dirichlet process with the intensity function of the Hawkes process. The resulting Dirichlet–Hawkes process (DHP) is then used as a prior for clustering documents appearing in a continuous-time stream. The inference is realized with a sequential Monte Carlo (SCM) algorithm. Following DHP, two articles have been published extending the idea: the Hierarchical Dirichlet–Hawkes process (HDHP) [13] in 2016 and Indian Buffet Hawkes process in 2018 [20]. Another work proposed an EM algorithm for the inference [24] in 2017. (It uses a heuristic method to update the number of clusters and cannot handle a stream of documents.)

A common feature of all the models we mentioned is that they use a nonparametric Dirichlet process (DP) prior or variations built on it, such as DHP and HDHP. Yet, on several occasions, it has been pointed out that there are no specific reasons to use this process in particular and that alternative forms might work better depending on the dataset. In [23], the author relaxes several conditions associated with DP and shows that alternative priors are an equally valid choice in Bayesian modeling. In [21], the authors derive the uniform process (UP) and show that it performs better on a document clustering task. In [17], the authors generalize UP and DP within a more general framework, the powered Dirichlet process (PDP), and show it performs better than DP on several datasets.

Moreover, it has recently been highlighted that DHP does not work well when the textual information within documents conveys little information, that is when the text is short [25] or when vocabularies overlap significantly. To answer this problem, the authors develop an approach based on Dirichlet process mixtures, which is not designed for continuous-time document streams—the temporal aspect comes from a sampling function as in [1, 4]. There are other limiting cases for DHP, for instance when temporal information is conveys little information (few observations, overlapping temporal intensities) or when documents within textual clusters do not follow the same temporal dynamics. To overcome those limitations, we develop the powered Dirichlet–Hawkes process in the next section.

3 Model and algorithm

3.1 Dirichlet prior and alternatives

We briefly recall the definition of a Dirichlet prior. A Dirichlet prior for clustering implements the assumption that the more a cluster is populated, the more chances a new observation belongs to it ("rich-get-richer" property). Besides, there is still a chance that a new observation gets assigned to a newly created cluster. It is often expressed using a metaphor, the Chinese

Restaurant process (CRP), and it goes as follows: if an ith client arrives in a Chinese restaurant, they will sit at one of the K already occupied tables with a probability proportional to the number of persons already sat at this table. They can also sit alone at a new table $K+1$ with a probability inversely proportional to the total number of clients in the restaurant. When their choice is made, the next client arrives, and the process is repeated. Let c be the cluster chosen by the ith customer, $\vec{C^-}$ the table assignment of previous customers up to $i-1$, N_c the population of table c, C the number of already occupied tables and $\alpha_0 \in \mathbb{R}^+$ the concentration parameter. The process can be written formally as:

$$\text{CRP}(C_i = c|\vec{C^-}, \alpha_0) = \begin{cases} \frac{N_c}{\alpha_0+N} \text{ if } c = 1, 2, ..., C \\ \frac{\alpha_0}{\alpha_0+N} \text{ if } c = C+1 \end{cases} \tag{1}$$

The uniform process [21] has been proposed as an alternative to the DP prior. In this context, a new customer entering the restaurant has an identical chance to sit at either of the occupied tables, and a chance to sit at an empty table inversely proportional to the number of occupied tables. Formally:

$$\text{U-CRP}(C_i = c|\vec{C^-}, \alpha_0) = \begin{cases} \frac{1}{\alpha_0+C} \text{ if } c = 1, 2, ..., C \\ \frac{\alpha_0}{\alpha_0+C} \text{ if } c = C+1 \end{cases} \tag{2}$$

Finally, the powered Dirichlet process [17] generalizes the two above, stating that the probability for a new client to sit at a new table depends arbitrarily on the number of customers already sat at this table:

$$\text{P-CRP}(C_i = c|r, \vec{C^-}, \alpha_0) = \begin{cases} \frac{N_c^r}{\alpha_0+\sum_{c'} N_{c'}^r} \text{ if } c = 1, 2, ..., C \\ \frac{\alpha_0}{\alpha_0+\sum_{c'} N_{c'}^r} \text{ if } c = C+1 \end{cases} \tag{3}$$

where $r \in \mathbb{R}^+$ is an hyper-parameter. Varying r allows to give more or less importance to the "rich-get-richer" hypothesis of DP. Note that we recover previous processes as $\text{P-CRP}(r=0, \vec{C^-}, \alpha_0) = \text{U-CRP}(\vec{C^-}, \alpha_0)$, and as $\text{P-CRP}(r=1, \vec{C^-}, \alpha_0) = \text{CRP}(\vec{C^-}, \alpha_0)$. We will use this more general form in the rest of this work and make r vary to compare those priors in the experimental section.

3.2 Hawkes processes

A Hawkes process is defined as a self-stimulating temporal point process. It is used to determine the probability of an event happening given the realization of all previous events in a continuous space. Point processes are fully characterized by the intensity function $\lambda(t)$, which is related to the probability P of an event happening between t and $t+\Delta t$ by $\lambda(t) = \lim_{\Delta t \to 0} \frac{P(t_{\text{events}} \in [t; t+\Delta t])}{\Delta t}$. In the case of Hawkes processes, $\lambda(t)$ is defined conditionally on all the events that happened at times lower than t. In our setup, we define one Hawkes process for each cluster, independent from the others. The intensity of the Hawkes process associated with cluster c is defined as:

$$\lambda_c(t|\mathcal{H}_{<t,c}) = \sum_{\mathcal{H}_{<t,c}} \vec{\alpha_c}^T \cdot \vec{\kappa}(t_{i,c}) \tag{4}$$

where $t_{i,c}$ is the time of the ith observed in cluster c, $\mathcal{H}_{<t,c} = \{t_{i,c}|t_{i,c} < t\}_{i=1,2,...}$ is the history of events in cluster c up to t, $\vec{\alpha_c}$ is a vector of coefficients, $\vec{\kappa}(t)$ is a vector of kernel functions with the same dimension as $\vec{\alpha}$ and $\cdot$ represents the dot product. The kernel

functions are set on stone. We will later infer the weights vector $\vec{\alpha}$ to determine which entries of the kernel vector are the most relevant for a given situation. This technique has become standard in Hawkes processes modeling and used in several occasions [9, 26]. Finally, we consider an additional time-independent Hawkes process (that is a Poisson process) of intensity $\lambda(t) = \lambda_0$. This process is used as the Dirichlet–Hawkes equivalent of the concentration parameter α_0 in a Dirichlet process [see Eq. (1)]. It translates the probability of opening a new cluster as the realization of a Poisson process. In the same way that in DP no observation is assigned to a cluster whose counts is α_0 but instead to a new cluster, no observation will be associated with the Poisson process but instead to a new Hawkes process.

Finally, the likelihood of a combination of independent Hawkes processes can be written:

$$\begin{aligned}\mathcal{L}(\vec{\lambda}|\mathcal{H}_{<T,c}) &= \mathcal{L}(\lambda_0|\mathcal{H}_{<T,c}) \prod_c \mathcal{L}_c(\lambda_c|\mathcal{H}_{<T,c}) \\ &= e^{-\int_0^T \lambda_0 dt} \prod_c e^{-\int_0^T \lambda_c(t)dt} \prod_{t_{i,c}} \lambda_c(t_i|\mathcal{H}_{<t_{i,c},c}) \\ &= e^{-\lambda_0 T + \sum_c \int_0^T \lambda_c(t|\mathcal{H}_{<t,c})dt} \prod_{t_{i,c'},c'=c} \lambda_c(t_{i,c'}|\mathcal{H}_{<t_{i,c'},c}) \end{aligned} \tag{5}$$

where T is the upper time of the considered observation window, going from 0 to T. Note that $\mathcal{L}(\lambda_0) = e^{-\int_0^T \lambda_0 dt}$ because no event will be assigned to the Poisson process.

3.3 Powered Dirichlet–Hawkes process

Following the reasoning in [8], we substitute the counts N_k of the PDP with the inferred Hawkes intensities in the PDP, resulting in the following form for the powered Dirichlet–Hawkes prior [18]:

$$P(C_i = c|t_i, r, \lambda_0, \mathcal{H}_{<t_i,c}) = \begin{cases} \frac{\lambda_c^r(t_i)}{\lambda_0 + \sum_{c'} \lambda_{c'}^r(t_i)} & \text{if } c \leq C \\ \frac{\lambda_0}{\lambda_0 + \sum_{c'} \lambda_{c'}^r(t_i)} & \text{if } c = C+1 \end{cases} \tag{6}$$

where t_i is the arrival time of document i. We reformulated the Dirichlet–Hawkes process in order to allow nonlinear dependence (r) on the non-integer counts ($\vec{\lambda}$).

3.4 Textual modeling

We choose to model the textual content of documents as the result of a Dirichlet-multinomial distribution. This model is purposely simple to ease the understanding, but can easily be replaced by a more complex one. A more complete textual modeling is out of the scope of this work, which aims to highlight the efficiency of the PDHP. Here, a document will be associated with a given cluster according to words count in every cluster and words count in the document only. The generative process is as follows:

$$\theta_i \sim Dir(\theta_0); \quad \omega_{v,i} \sim Mult(\theta_i) \tag{7}$$

where θ_i is the cluster of document i, and $\omega_{v,i}$ is the vth word of document i. Let $\mathcal{L}_{txt}(\vec{C}_{<i,c}|N_{<i,c}, \theta_0)$ be the marginal joint distribution of every document's cluster allocation up to the ith one. The likelihood of the ith document belonging to cluster c can then be expressed as:

$$
\begin{aligned}
\mathcal{L}(C_i = c | N_{<i,c}, n_i, \theta_0) &= P(n_i | C_i = c, N_{<i,c}, \theta_0) \\
&= \frac{\mathcal{L}_{txt}(\vec{C}_{<i,c} | N_{<i,c}, \theta_0)}{\mathcal{L}_{txt}(\vec{C}_{<i-1,c} | N_{<i,c}, \theta_0)} \\
&= \frac{\frac{\cancel{\Gamma(\theta_0)}}{\Gamma(N_c + n_i + \theta_0)} \prod_v \frac{\Gamma(N_{c,v} + n_{i,v} + \theta_{0,v})}{\cancel{\Gamma(\theta_{0,v})}}}{\frac{\cancel{\Gamma(\theta_0)}}{\Gamma(N_c + \theta_0)} \prod_v \frac{\Gamma(N_{c,v} + \theta_{0,v})}{\cancel{\Gamma(\theta_{0,v})}}} \\
&= \frac{\Gamma(N_c + \theta_0)}{\Gamma(N_c + n_i + \theta_0)} \prod_v \frac{\Gamma(N_{c,v} + n_{i,v} + \theta_{0,v})}{\Gamma(N_{c,v} + \theta_0)}
\end{aligned} \tag{8}
$$

where N_c is the total number of words in cluster c from observations previous to i, n_i is the total number of words in document i, $N_{c,v}$ the count of word v in cluster c, $n_{i,v}$ the count of word v in document i and $\theta_0 = \sum_v \theta_{0,v}$.

3.5 Posterior distribution

The resulting posterior distribution of the ith document over clusters is calculated using Bayes theorem. It is proportional to the product of the textual likelihood Eq. (5) and the temporal powered Dirichlet–Hawkes prior Eq. (8):

$$
\begin{aligned}
&P(C_i = c | r, n_i, t_i, N_c, \mathcal{H}_{<t,c}) \\
&\propto \underbrace{P(n_i | C_i = c, N_{<i,c}, \theta_0)}_{\text{Textual likelihood}} \underbrace{P(C_i = c | t_i, r, \lambda_0, \mathcal{H}_{<t_i,c})}_{\text{Temporal prior}} \\
&= \frac{\Gamma(N_c + \theta_0)}{\Gamma(N_c + n_i + \theta_0)} \prod_v \frac{\Gamma(N_{c,v} + n_{i,v} + \theta_{0,v})}{\Gamma(N_{c,v} + \theta_0)} \\
&\quad \times \begin{cases} \frac{\lambda_c^r(t_i)}{\lambda_0 + \sum_{c'} \lambda_{c'}^r(t_i)} \text{ if c = 1, ..., C} \\ \frac{\lambda_0}{\lambda_0 + \sum_{c'} \lambda_{c'}^r(t_i)} \text{ if c = C+1} \end{cases}
\end{aligned} \tag{9}
$$

We recall that $\lambda_c(t)$ is defined Eq. (4). The textual likelihood of cluster $C + 1$ is computed by setting $N_{C+1,v} = 0$.

3.6 Algorithm and changes induced by PDHP

We use a similar algorithm to the one in [8]. Briefly, the algorithm is a sequential Monte Carlo (SMC) that takes one document at a time in their order of arrival. The algorithm starts with a number N_{part} of particles whose weights are $\omega_p = \frac{1}{N_{\text{part}}}$, each of which will keep track of a hypothesis on documents clusters. After a few iterations, particles that contained unlikely allocation hypotheses are discarded and replaced by more likely ones. The likeliness of a hypothesis is encoded in the weights of each particle ω_p.

For each particle, when a new document arrives, (1) the cluster of the document is sampled according to a categorical distribution over all clusters, whose weights are determined by Eq. (9). After the cluster of the new document has been sampled, (2) the kernel weights $\vec{\alpha}$ from Eq. (4) are updated using Eq. (5). For efficiency purpose, we infer $\vec{\alpha}$ using Gibbs sampling from a set of N_s pre-computed $\vec{\alpha}$ vectors. We finally (3) update the weights ω_p of each particle according to the posterior Eq. (9) such as $\omega_p^{(n+1)} = \omega_p^{(n)} \times$ Eq. (9). If the weight of a particle falls below a value ω_{thres}, the particle is discarded and replaced by another

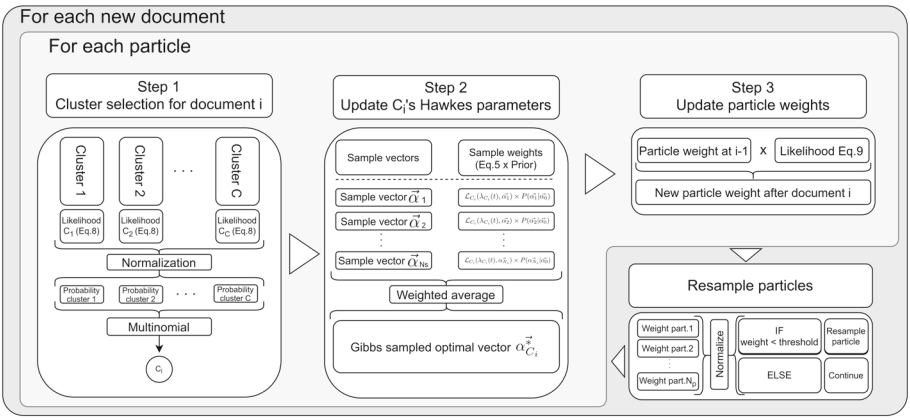

Fig. 1 *Schematic workflow of the SMC algorithm*—for each new observation from a stream of document, we run steps 1 (sample document's cluster), 2 (update sampled cluster's internal dynamics) and 3 (update particle hypothesis' likeliness) for each particle, and then discard particles containing the less likely hypothesis on cluster allocation

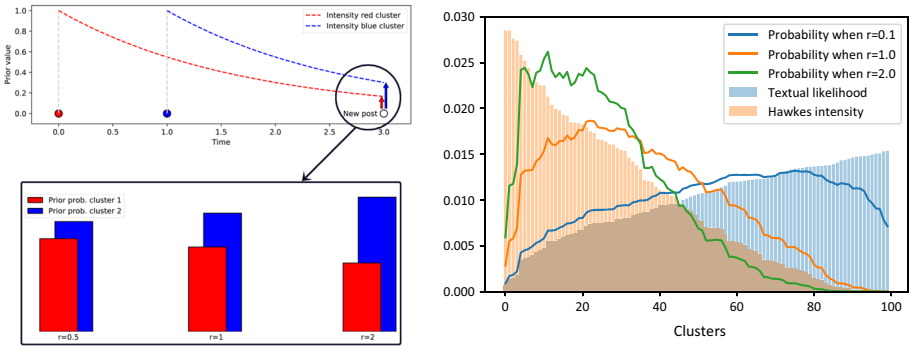

Fig. 2 *Effect of r on cluster selection probabilities*—the probability for each cluster to get chosen (solid lines) for several values of r and fixed individual textual likelihood (blue bars) and Hawkes intensity (orange bars)

existing one with sufficient weight. The whole process is illustrated in Fig. 1. By updating incrementally the likelihood associated with each of the pre-computed $\vec{\alpha}$ sample vectors, the algorithm treats each new observation in constant time—see below.

The task of updating kernels coefficients (2) is the same as in any Hawkes process, and the task of updating particles weights and resampling them (3) is common to any SMC algorithm. The change induced by the PDHP compared to the DHP happens at step (1). First of all, we note that for $r = 1$ the PDHP prior is identical to the DHP prior. From Poux-Médard et al. [17], lowering the value of r reduces the "rich-get-richer" aspect of the PDP ("rich-get-less-richer"), whereas increasing it leads to a "rich-get-more-richer" effect. These metaphors can be translated as follows in our temporal context: For lower values of r, the relative difference between cluster's temporal intensities plays a less important role in cluster selection, whereas higher values of r tend to exacerbate these differences and make the temporal aspect of the greatest consequence on the choice of a cluster. In other words, tuning the value of r allows to give more or less importance to the temporal aspect of the clustering. This is illustrated in Fig. 2. On the left, we plot the situation when a new observation has to get assigned a cluster.

The associated Hawkes intensities are the base to compute the prior probability for either cluster. This quantity is then modulated using r to give more or less importance to intensity differences between clusters. On the right of Fig. 2, we plot the probability for various clusters to be chosen [which is directly proportional to the posterior distribution, Eq. (9)] according to r when their textual likelihood and Hawkes process intensity are known. Note that for $r = 0$, the probability for any cluster to get chosen is directly proportional to its textual likelihood (Dirichlet–uniform process), whereas when r increases, the probability of getting chosen gets closer to a selection only based on the temporal aspect.

This makes the main interest of the PDHP model. Tuning the parameter r allows one to choose whether inferred clusters are based on textual or temporal considerations. It generalizes several state-of-the-art works, which are special cases of the PDHP for different values of r. The DHP [8] is equivalent to PDHP for $r = 1$; the UP [21] is equivalent to PDHP when $r = 0$. In the following sections, we show how fine-tuning r systematically yields significantly better results than setting it to $r = 0$ or $r = 1$ (up to a gain of 0.3 on our experiments' normalized mutual information metric). We also show how varying it allows to recover one kind of clustering or the other (textual or temporal) with high accuracy and see how it affects clustering results on several real-world datasets.

3.7 Time complexity

- (Step 1) When an nth observation is treated, the algorithm depicted in Fig. 1 samples a cluster from C possible clusters. For each possible cluster c, it first computes the textual likelihood from Eq. (8); this operation has a complexity that scales linearly with the vocabulary size $\mathcal{O}(V)$. Then, it computes the Hawkes intensity at the time of the new document from Eq. (4). The time complexity scales with the size of the cluster c's history $\mathcal{H}_{<t_n,c} \propto n$. However, as in [8], observations in the history that are older than a time t_{old} are discarded –the threshold for "being old" is determined by the kernel $\kappa(t)$ from Eq. (4). Therefore, the active history $\mathcal{H}_{>t_{\text{old}},<t_n,c}$ has a constant size, which depends on the density of observations in time, noted ρ—which does not vary in most cases. The difference between t_n and t_{old} is constant and noted Δt. Finally, this operation is repeated $C + 1$ times, once for each cluster and once for the empty cluster, yielding a complexity for the first step of Fig. 1 equal to $\mathcal{O}((C + 1)(V + \rho\Delta t))$.
- (Step 2) The algorithm then updates the coefficients of the triggering kernel. First, for each of N_{samples} sample vectors, we must increment its likelihood given the new observation. Once again, from Eq. (5), this is done by discarding older observations. The considered history $\mathcal{H}_{>t_{\text{old}},<t_n,c}$ has a size that scales with $\rho\Delta t$, with ρ the observations density. The complexity of updating the likelihood therefore takes $\mathcal{O}(N_{\text{samples}}\rho\Delta t)$. Their weighted average is performed in constant time.
- (Step 3) Finally, updating the particles' weight boils down to retrieving their likelihood, which has already been computed earlier, which takes a constant time $\mathcal{O}(1)$.

Each of these steps is performed N_{part} times, once per particle, and N_{obs} times, once per new observation. Resampling unlikely particles can be done in constant time, which leaves us with the final time complexity of the algorithm: $\mathcal{O}(N_{\text{obs}}N_{\text{part}}((C+1)(V+\rho\Delta t)+N_{\text{samples}}\rho\Delta t+1))$. We note that complexity depends linearly on the size of the dataset N_{obs}. The processing time of one new observation mostly depends on the number of existing clusters, given all the other parameters are constant over time.

4 Experiments

4.1 Synthetic data

4.1.1 Synthetic data generation

We simulate a case where only two clusters are considered. Each cluster has its own vocabulary distribution over 1000 words and its own kernel weights $\vec{\alpha}$, with Gaussian Hawkes kernel functions $\kappa(\vec{t})$ of parameters (μ, σ) = (3, 0.5), (7, 0.5) and (11, 0.5) [see Eq. (4)]. Finally, we set $\lambda_0 = 0.05$. We first simulate one independent Hawkes process per cluster using the Tick Python library [2]. The processes are stopped at time $t = 1500$, which makes a rough average of 7000 events per run. Then we associate each simulated observation with a sample of 20 words drawn from the corresponding cluster's word distribution. Inference has been performed using a 8 core processor (i7-7700HQ) with 8GB of RAM on a laptop, which underlines how scalable the algorithm is. As stated before, the algorithm treats each new document in constant time $\mathcal{O}(1)$, which ranged from 0.05 s on synthetic data to maximum 1s on real-world data. Note that this number is directly proportional to the number of active inferred clusters, and thus depends strongly on the dataset.

We generate ten such datasets for every considered value of vocabulary overlap and Hawkes intensities overlap, which leave us with ~ 200 datasets. Overlap is defined as the common area of two distributions, normalized by the total area under the distributions. For example, if the vocabulary of one cluster ranges from words "1" to "100" with uniform distribution, and the vocabulary of another cluster from words "50" to "150" with uniform distribution, the overlap equals 50%. We define the overlap of Hawkes process intensity in the same way. If the triggering Hawkes kernel of one cluster is a Gaussian function with $(\mu, \sigma) = (3, 1)$ and one associated observation at $t = 0$, and the triggering kernel of the other is also a Gaussian function but with $(\mu, \sigma) = (5, 1)$ also with an associated observation at $t = 0$, the overlap equals 32% (see Fig. 3). When computing the Hawkes intensity overlap, every observation within a cluster and its associated timestamp are considered. The definition of overlaps is illustrated in Fig. 3. To enforce a given vocabulary overlap (Fig. 3-right), we shift the word distributions of the clusters from which events' vocabulary is sampled. To

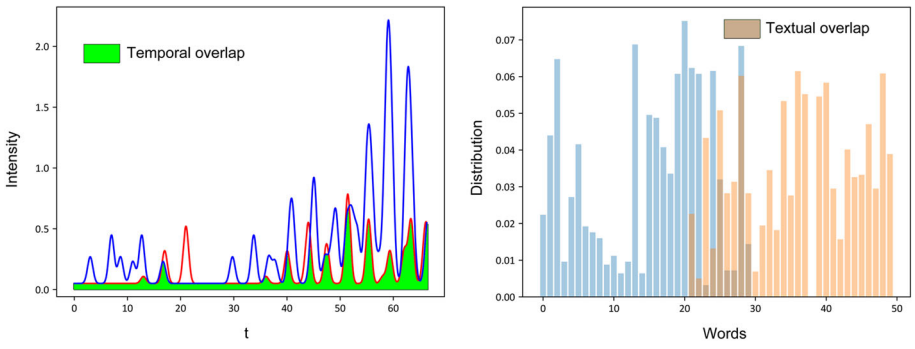

Fig. 3 *Overlaps*—(left) Temporal overlap is defined as the ratio between the area common to two Hawkes intensities and the total area under the intensity functions. (Right) Textual overlap is defined as the proportion of vocabulary that is common to two clusters, weighted by the probability of words within their respective cluster

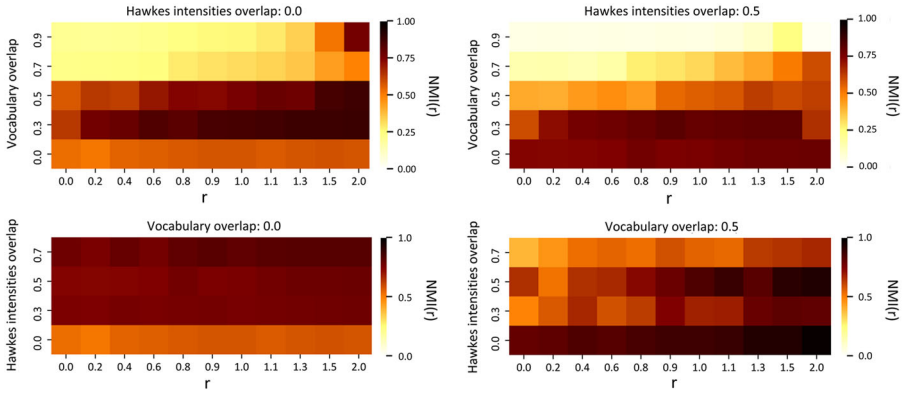

Fig. 4 *PDHP yields good NMI values*—normalized mutual information (NMI) for various values of r, intensities overlap and vocabulary overlap, for one dataset per combination. The results for $r = 0$ are the output of the uniform process, the results for $r = 1$ are the output of the DHP [8], and the other values of r correspond to other special cases of PDHP. The darker the better. Overall, PDHP yields good values of NMI

enforce a given Hawkes intensities overlap (Fig. 3-left), we shift the event times of every event in one of the clusters until we get the correct overlap ($\pm 5\%$).

Note that we consider ten different datasets instead of considering ten runs per dataset for two reasons. First, the generation of Hawkes processes is highly stochastic, so a model might perform significantly better on a single dataset only by chance. Second, given the way the SMC algorithm works, the standard deviation between runs is small: At each iteration, N_{part} clustering hypotheses are tested, which is equivalent to running N_{part} times a single clustering algorithm. We heuristically set $N_{\text{part}} = 8$, as we observe no significant improvement using more particles.

The other parameters we use for clustering synthetic data are: $\alpha_0 = 0.1$, $\theta_0 = 1$, $\vec{\kappa}(t) = [\mathcal{G}(t; 3, 0.5), \mathcal{G}(t; 7, 0.5), \mathcal{G}(t; 11, 0.5)]$ with $\mathcal{G}(t; \mu, \sigma)$ the Gaussian function, $N_{\text{samples}} = 2.000$ and $\omega_{\text{thres}} = \frac{1}{2N_{\text{part}}}$.[1]

We are interested in varying both vocabulary and intensities overlap to exhibit the limits of DHP and how PDHP overcomes them. Note that in the synthetic data experiments in [8] (Fig. 3a, b), the intensities overlap is almost null, which makes the task easier for the Hawkes part of the algorithm. The primary metric we use throughout the experimental section is the normalized mutual information (NMI). During the experiments, we also considered the Adjusted normalized rand index and the V-measure, which are well adapted to evaluate clustering results when the number of inferred clusters is different from the true number of clusters. The observed trends in results from these other metrics are identical to the ones observed for NMI. Therefore we choose to report only the results of the latter for clarity. These additional measurements are provided in the linked repository along with the code and datasets.

4.1.2 PDHP yields better results as vocabulary overlap increases

We report our results when the intensities overlap is null, with varying r and the vocabulary overlap in Fig. 5a. Because we consider ten different datasets for each set of overlap parameters, it makes no sense to report the absolute average NMI since it can vary greatly from one

[1] All codes and implementations are available at https://github.com/GaelPouxMedard/PDHP.

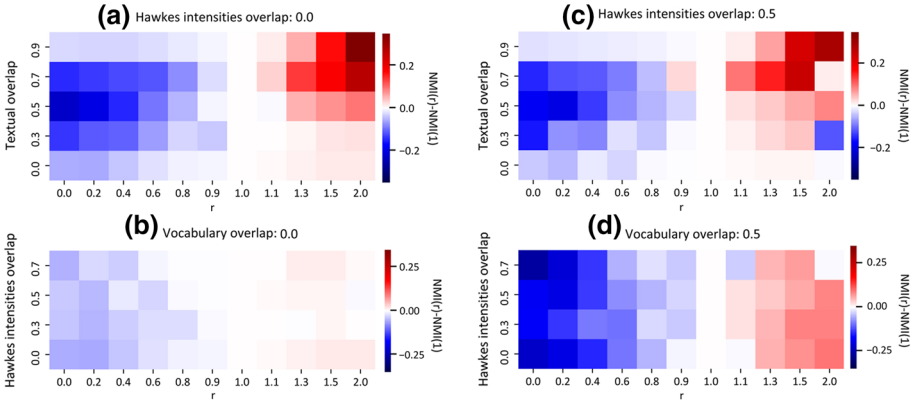

Fig. 5 *PDHP performs better than DHP*—difference between the normalized mutual information (NMI) of PDHP and DHP model [8] for various values of r, intensities overlap and vocabulary overlap, averaged over all the datasets. Red means PDHP performed better, and blue means PDHP performed less well. Because PDHP($r = 1$) = DHP, the column $r = 1$ show no difference. PDHP allows to increase results on NMI by as much as 0.3 over DHP

dataset to the other. Instead, we plot the relative NMI difference between PDHP and DHP ($r = 1$), which we expect to be less dependent on the datasets we consider. However, to give an idea of the typical performance for some parameters, we also provide raw results for one run in Fig. 4.

There is a clear correlation between efficiency, vocabulary overlap and r, with a gain on NMI up to +30% of its maximal value over DHP. As stated at the end of the "Model" section, this result was expected: The more vocabulary overlap grows, the less textual content carries valuable information for clustering the documents. This observation supports the concerns raised in [25] about the efficiency of DHP for clustering short text documents. However, Hawkes intensities overlap being null, the arrival time of events carries highly valuable information when textual content does not allow to distinguish clusters well. Therefore, PDHP provides a way to tackle the problem raised in [25] without the need to sample observations.

Conversely, when vocabulary overlap is null, the textual content provides enough information to distinguish clusters correctly. The temporal dimension only allows refining the results with no significant improvement for all values of r.

Finally, we can see how the Dirichlet–uniform process (DUP, $r = 0$) consistently yields worse performances under these settings. Once again, this is expected since, in this synthetic experiment, intensities overlap carry valuable information about events clustering; DUP only considers textual information and therefore misses valuable clues.

4.1.3 PDHP yields similar results for null vocabulary overlap

We report similar results in Fig. 5b. Here, we consider a null vocabulary overlap for various values of r and of Hawkes intensities overlap. The situation is now the opposite: the textual content always carries valuable information about clusters, whereas the temporal aspect does not. We observe the same trend as in Fig. 5a—note that the color scale is the same. Varying the value of r does not significantly change the performances of clustering, meaning the textual content always carries enough information. This plot shows that PDHP can handle

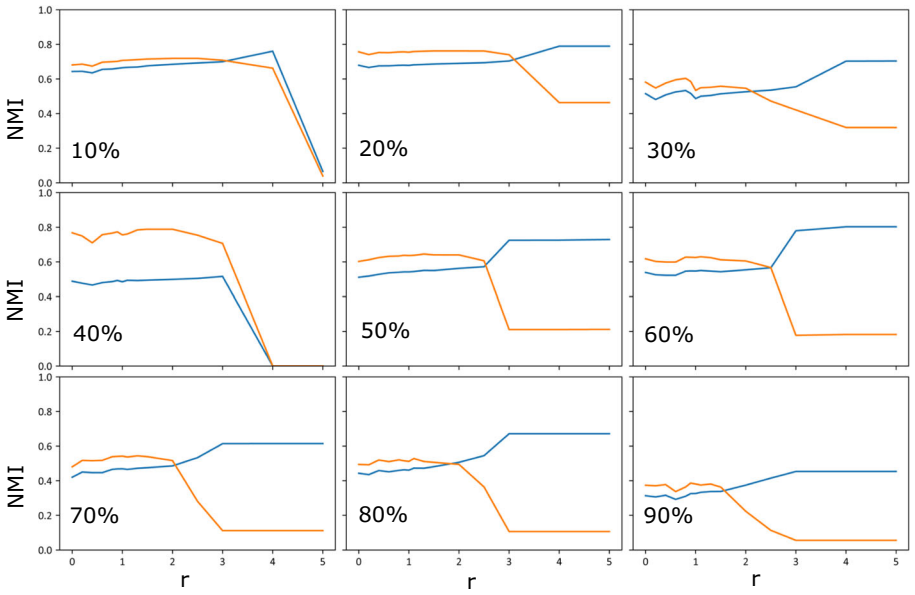

Fig. 6 *Textual (orange) and temporal (blue) NMI vs r when textual and temporal clusters are decorrelated*—from top-left to bottom-right, there are 10, 20, 30, 40, 50, 60, 70, 80 and 90% of generated events that have been randomly re-assigned a textual cluster. The orange curves are the textual NMI vs r that evaluate how well events whose vocabulary has been sampled from the same distribution are clustered together; the blue curves are the temporal NMI vs r that evaluate how well events following the same temporal dynamic are correctly clustered together. Values presented are for one dataset. We clearly see that varying r allows to retrieve the right temporal (r large) or textual clusters (r small)

greater intensities overlap without collapsing into unrealistic clustering. Since in most real-case applications, many clusters with various dynamics may coexist simultaneously, it is comforting that the PDHP can also handle this case.

4.1.4 PDHP yields better results in more realistic situations

We finally report the results for intermediate values of intensities and vocabulary overlaps in Fig. 5c,d. In real-world applications, it seldom happens that topics vocabularies do not overlap at all. For example, a quick analysis of *The Gutenberg Webster's Unabridged Dictionary by Project Gutenberg* shows that there are 22% of English words that are associated with more than one definition. A more detailed analysis would need to consider the usage frequency of words to get correct statistics. Still, this number provides an estimate of the effective vocabulary overlap in real-world situations.

In Fig. 5c, we present the results for a fixed intensities overlap of 0.5 versus various values of r and vocabulary overlaps, and in Fig. 5d for a fixed vocabulary overlap of 0.5 versus various values of r and intensities overlaps. Once again, we see that, on average, using PDHP can increase the NMI over DHP up to +20% of the maximum possible value.

4.1.5 PDHP finds textual or temporal clusters depending on *r*

We now slightly modify our experimental setup. Instead of considering that textual clusters and Hawkes intensities are perfectly correlated, we consider a decorrelated case. A document

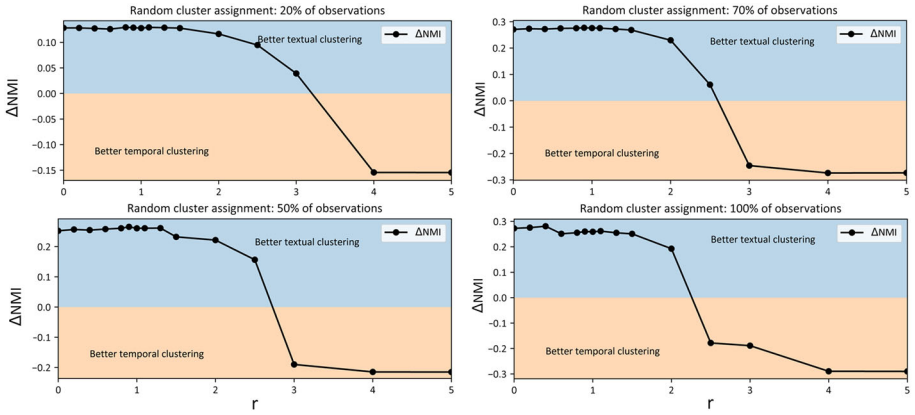

Fig. 7 *Varying r allows to choose between textual or temporal clustering*—the black line plots the difference between the NMI of textual and temporal clustering. For small r, textual clustering is far better than temporal clustering, and for large r, the situation is reversed. This is because r determines the importance given to the temporal dimension and therefore allows choosing between retrieving temporal or textual clusters

whose vocabulary is drawn from cluster C_1 can now follow the same temporal dynamics as cluster C_2. If we imagine a dataset of news articles published online, it is clear why this might happen frequently. If popular newspapers such as New York Times or Reuters publish an article on topic A at time t, it is likely to trigger snowball publications of similar articles from less popular journals. "Popularity" is chosen as an indicator in this example, but it may be any other external parameter (centrality in news networks, support of publications, etc.). In this case, the article's textual content allows to uncover a "story of publication", that is, how the article has been spread, when publication spikes are, etc. However, the temporal information would help understand the dynamics of publications interaction: which reduced set of articles triggered the publication of subsequent ones.

In [8], it is assumed that every document within clusters follows a unique dynamics. We relax this hypothesis in our datasets as follows. For null textual and temporal overlaps, after a dataset has been generated, we resample the textual clusters of a fraction of randomly selected events, as well as the words associated with the event. Doing so, we decorrelate temporal and textual clusters. Therefore, an event is now described by two cluster indicators: its temporal cluster (which Hawkes intensity made the event appear where it is) and a textual cluster (which vocabulary has been used to sample the words the event contains).

For completeness, we also show the results for various decorrelations for one run in Fig. 6. To better understand the tendency of NMIs with respect to r, we plot the average difference between the NMI of textual clustering and the NMI temporal clustering over all the datasets. Explicitly: $\Delta NMI = NMI_{text} - NMI_{temp}$. The results are reported in Fig. 7.

As supposed at the end of the "Model" section, varying r allows retrieving one clustering or the other. Note that the value r of transition from text to time clustering depends directly on the dataset considered: number of words sampled, vocabulary size, overlaps, etc.

4.1.6 PDHP efficiently infers the temporal dynamics of each cluster

Finally, we show that PDHP correctly infers kernels' parameters in every situation where events are correctly assigned to their temporal cluster. The results are reported in Fig. 8. We looked at the mean absolute error (MAE) and the mean Jensen–Shannon divergence (MJS)

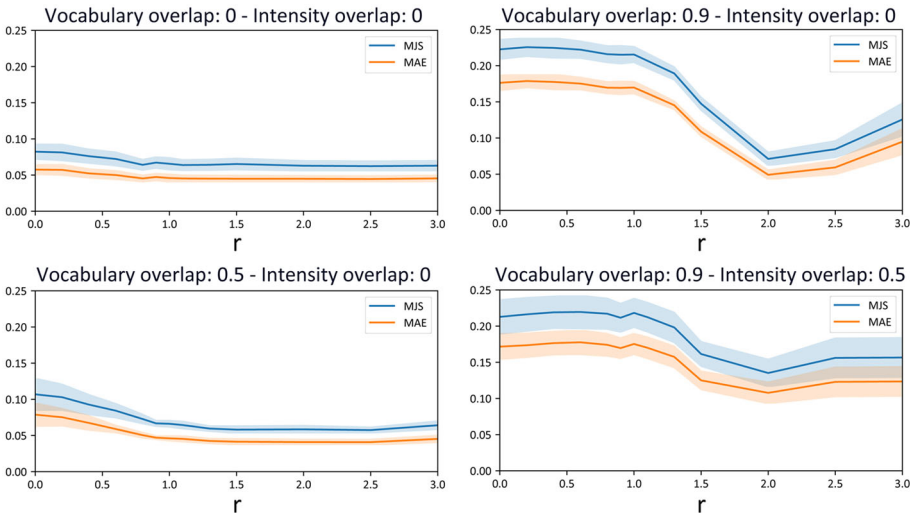

Fig. 8 *Varying r allows to better capture the dynamics at stake*—we plot the mean average error and the mean Jensen–Shannon divergence of the inferred kernel function α with respect to the kernel used to generate the data, for various values of temporal and textual overlaps. The standard error bars are computed over 10 independent runs. The higher the temporal overlap, the larger the error bars; the larger the textual overlap the more influence has r

between the vector $\vec{\alpha}$ used to generate the dataset and the inferred one. We note in Fig. 8 that when textual overlap is small, the inferred kernel is close to the real one and r has little impact on the result. This is because the inferred kernels mostly depend on the correctness of inferred clusters: When observations are allocated to the right clusters, the Hawkes process inference considers relevant information when inferring these clusters' dynamics. However, when observations are misallocated, the Hawkes process infers dynamics also based on times that are not supposed to contribute this cluster's dynamic. When the clustering task is simple and yields good results (that is, when textual overlap is small, see Fig. 4), the PDHP infers correct temporal dynamics ($\sim$ 5% MAE); this show our method correctly accounts for clusters dynamics given the available information.

When vocabulary overlap is large, the value of r significantly influence the kernel inference performances. However, when r is chosen so that clusters are correctly inferred, the kernel inference retrieves well the expected kernels ($\sim$ 5% MAE). Finally, the temporal kernel inference is expected to be less precise when temporal overlap increases, which is what happens in Fig. 8-bottom-right. In this case, the model does not retrieve well the synthetic kernels even for the optimal r. Besides, the error bars get wider as a consequence of the clusters allocation to be more challenging. Overall, provided the right clusters, we conclude that our method correctly retrieves the inferred temporal kernels.

4.2 Real-world application on Reddit

In this section, we extend the results sketched in the original publication [18] on real-world datasets. We use the PDHP prior to model real streams of textual documents. We consider three Reddit datasets[2] about different topics. The *News dataset* is made of 73.000 titles extracted

[2] Available for download at https://files.pushshift.io/reddit/submissions/.

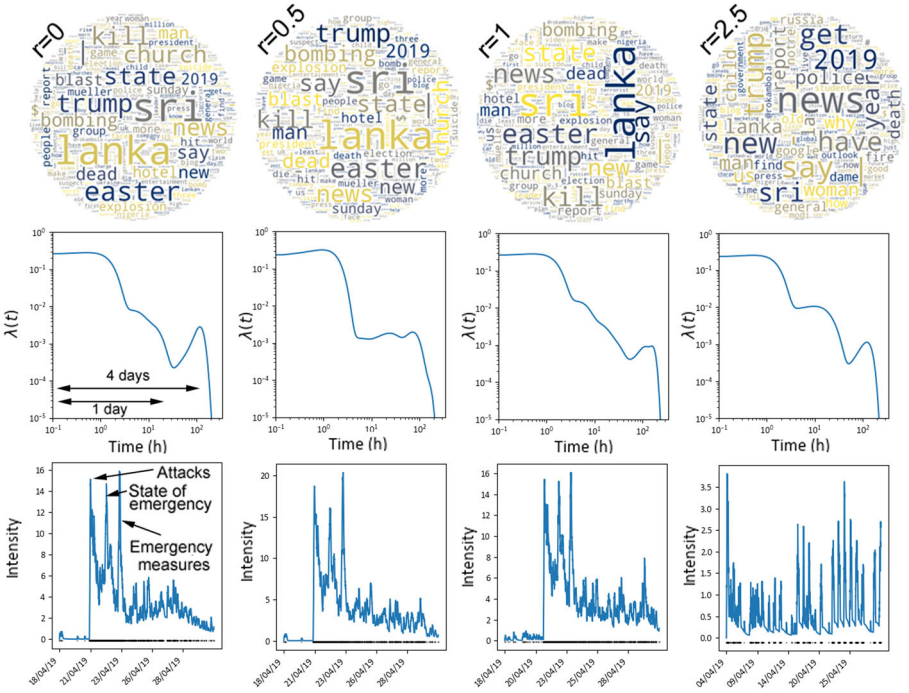

Fig. 9 Wordclouds, triggering kernels and intensities for clusters the most closely related to Sri Lanka 2019 bombings for various values of r. The points at the bottom of the intensity plots are individual publication events. Note that triggering kernels are plot on a log–log scale for visualization purpose, because most of the intensity is focused on small times: Dynamics are bursty

from the subreddits inthenews, neutralnews, news, nottheonion, offbeat, open_news, qualitynews, truenews and worldnews, from April 2019. We chose this month because of the wide variety of events that happened then (for instance, Sri Lanka Easter bombings, Julian Assange arrest, first direct picture of a black hole, Notre-Dame cathedral fire). We also consider 15.000 post titles of the subreddit TodayILearned (*TIL dataset*) and 13.000 post titles of the subreddit AskScience (*AskScience dataset*) on January 2019. We extracted the nouns, verbs, adjectives and symbols from the textual data. We run the experiments using the following parameters: $\alpha_0 = 0.5$, $\theta_0 = 0.01$, $N_{\text{samples}} = 2.000$, $N_{\text{part}} = 8$ and $\omega_{thres} = \frac{1}{2N_{\text{part}}}$. The kernel vector $\vec{\kappa}$ is made of Gaussian functions, with means located at 0.5, 1, 4, 8, 12, 24, 48, 72, 96, 120, 144 and 168 hours. The variance of each are set to 1, 1, 3, 8, 12, 12, 24, 24, 24, 24, 24 and 24 hours. The algorithm will then infer the weights $\vec{\alpha}$ associated with each entry of the kernel vector $\vec{\kappa}$ for each cluster.

4.2.1 PDHP recovers meaningful stories

As an illustrative example, we consider the inferred clusters the most closely related to Sri Lanka Easter bombings of April 2019 in Fig. 9. The main bursts in the news related to this event happened on the 21st, 22nd and 23rd of January and, respectively, correspond to the bombings themselves, the declaration of the state of emergency, and finally their application on the 23rd. We plot the temporal kernels associated with this event on a log-log scale,

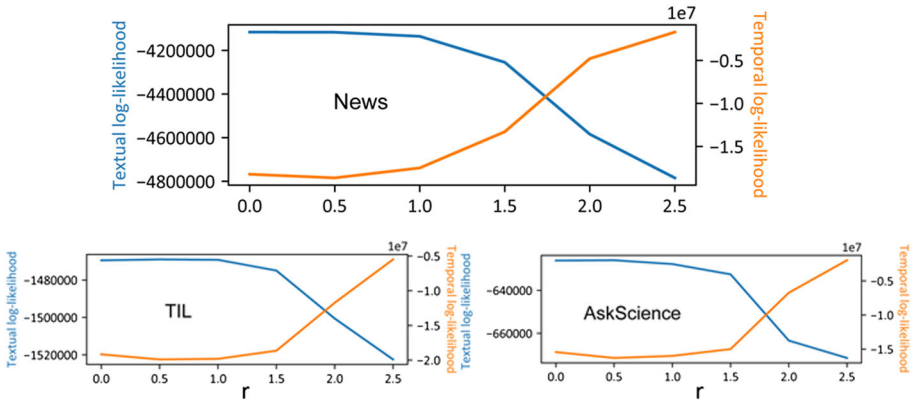

Fig. 10 *r allows to favor text-based of time-based clustering on real world datasets*—textual likelihood and Hawkes process likelihood for various values of r. The lower r the higher textual likelihood is, and the higher r the higher Hawkes process likelihood is

because most of the intensity is focused on small times: dynamics of information spread are bursty [12]. We see that inferred dynamics change with r as well as the cluster's vocabulary, which is expected since clusters do not contain the same documents. For $r = 0$, the uniform process infers clusters based on textual information only; the triggering kernel is inferred afterward. For $r = 2.5$ on the contrary, clusters are formed based on the triggering kernel, and textual information follows; we see from the right-plot that this cluster captures publications exhibiting a daily intensity cycle; this is visible both in the intensity plot (the bump around 2.10^2h which is not present on other temporal kernels) and in the real-time axis where publications seem to be packed around specific points in time roughly corresponding to a daily cycle. Given the intensity spikes on 21st, 22nd, and 23rd, it is not surprising that articles about Sri Lanka bombings are also part of this cluster. Note that the more r increases, the more intense the triggering kernel is around 24h. We see from Fig. 9 that DHP is a specific case of our modeling, and that tweaking the r parameter allows to retrieve completely different results.

4.2.2 PDHP favors temporal or textual clustering depending on *r*

We report the values of log-likelihoods for every dataset and various values of r in Fig. 10. The textual likelihood is defined Eq. (8), and the likelihood of a Hawkes process is defined Eq.(5). Note that r does not appear in either Eq. (8) or Eq. (5); the plot in Fig. 10 thus only reflects the relevancy of the proposed textual modeling or temporal modeling independently from PDHP. Those likelihoods evaluate how well the textual or temporal aspect of the dataset is modeled with no consideration of the model being used. As expected from the synthetic experiments, varying r makes the model more sensitive to either textual or temporal data –note the similarity to Fig. 6. A low r favors the textual information clustering and is thus better at modeling documents' textual content, whereas a high r favors temporal information which makes PDHP better at capturing the publication dynamics.

4.2.3 PDHP infers sharper textual clusters for low *r*

We evaluate how meaningful textual cluster are using an entropy measure. We assume that a cluster is meaningful when it contains a reduced set of words; a cluster talking about one

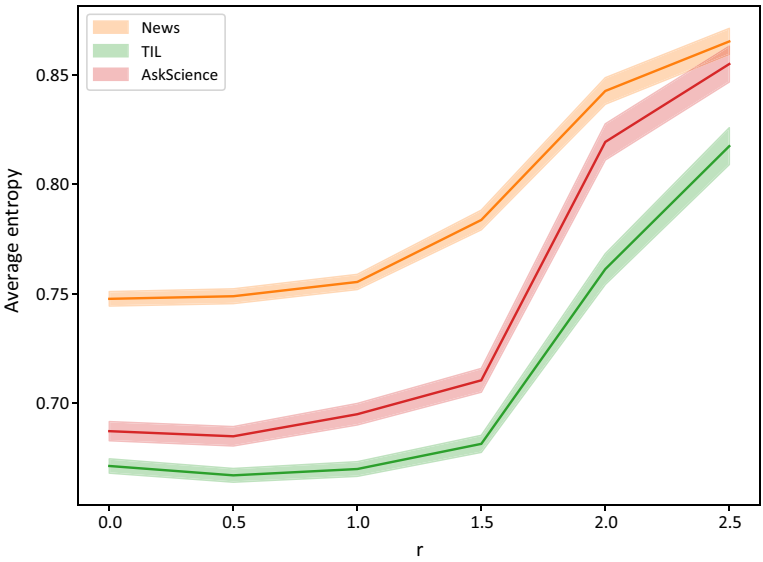

Fig. 11 *Textual clusters are more informative for low values of r*—weighted average entropy of words distribution for every dataset. Weights corresponds to the number of words within clusters. The error bar represents the standard error over all the clusters

topic only is more likely to have a smaller vocabulary that a cluster about two or more topics. A way to measure this is to see how spread the vocabulary of a cluster is using Shannon entropy. Let $N_{c,v}$ be the count of word v in cluster c. The normalized Shannon entropy of a cluster c is defined as:

$$S(\vec{N}_c) = \frac{1}{-\log(V)} \sum_{v}^{V} \log\left(\frac{N_{c,v}}{\sum_v' N_{c,v'}}\right) \frac{N_{c,v}}{\sum_v' N_{c,v'}} \tag{10}$$

An entropy of 0 means the vocabulary of the cluster is concentrated on a single word among the V possible words in the vocabulary; an entropy of 1 means that every of the V words is present to the same extent $\frac{1}{V}$. In Fig. 11, we plot the mean entropy for various values of r for all the datasets, along with the standard error over the clusters. The results show that on average vocabulary is more concentrated within clusters for low values of r. The inflection point of the curves corresponds to what has been previously observed with likelihoods in Figs. 10 and 7. On the contrary, higher values of r lead to clusters that comprise a less concentrated vocabulary. This is expected because as r increases, the textual information is no longer the most relevant data for cluster formation.

4.2.4 PDHP controls the burstiness

In Fig. 12, we plot the intensity function associated with the News dataset on the real time axis for several clusters for $r = 0.5$ and $r = 1.5$. Note that not each of the 300 inferred clusters are represented, but instead we consider only the ones whose intensity went above 10 at least once, the rest being considered as noise. First of all, both values of r allow to recover the major events of April 2019 (in order of appearance): the first direct picture of a black hole (10/04), the arrest of Julian Assange (11/04), the fire of Notre-Dame de Paris

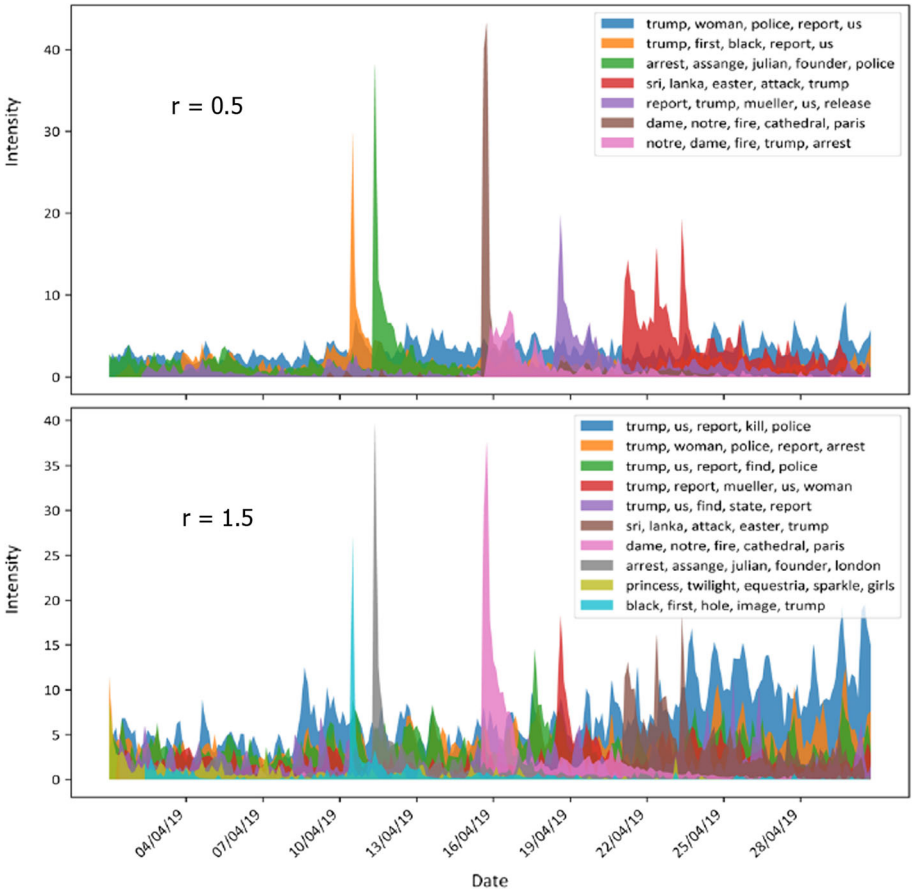

Fig. 12 *PDHP allows for modeling bursty events*—we plot PDHP's intensity for the News dataset over the observation period for two values of r. The clusters plot here are the ones out of ∼300 clusters that have an intensity going above 10 at least once. A lower r finds globally relevant clusters, whereas a higher r allows to recover shorter bursty events

cathedral (15/04), the release of the Mueller report on Donald Trump (18/04) and the Sri Lanka Easter bombings (21/04). The top 5 words of every cluster are reported in the legend.

When r increases, PDHP retrieves new clusters associated with shorter bursty events. For instance, the cluster associated with the release of a new episode of Equestria Girls that went unnoticed for $r = 0.5$ has been detected with $r = 1.5$. This happens because the episode has not been discussed over a long time period, and associated articles have a vocabulary significantly overlapping with other clusters' one. A model relying mostly on textual information might not detect specific words (twilight, equestria, sparkle, etc.). If detected and a new cluster is created, it might then fail to associate subsequent events to this new cluster if temporal information plays a lesser role. On the other side, a model favoring temporal information is much more likely to associate subsequent events to a new cluster despite textual information fitting well older and more populated clusters.

This can be seen in Fig. 9, where the intensity of a kernel peaks at short times. This results in encouraging the burstiness. When r is large, a given event is likely to be associated with

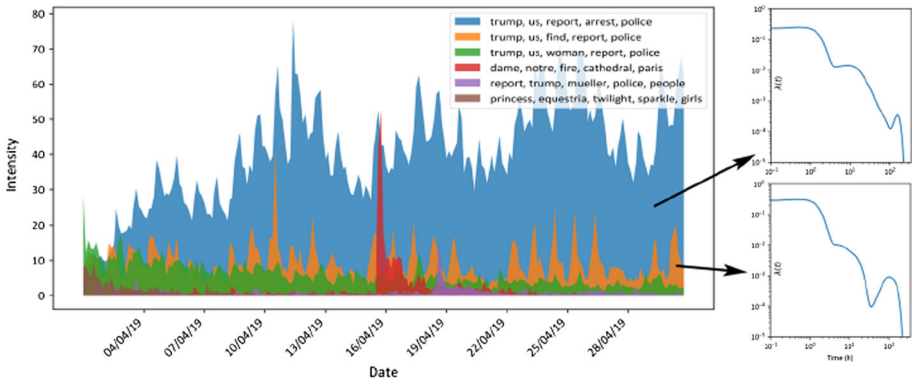

Fig. 13 *Limit case of encouraging bursty events clustering*—we plot PDHP's intensity for the News dataset over the observation period for a large value of r. In this case, textual information plays a marginal role, and clusters are inferred based on the events publication dates only

subsequent ones even when the associated vocabularies are only vaguely similar. On the other side, when r is small, older events with closer vocabularies have more chances of getting associated with it despite their intensity not peaking at the new event time.

4.2.5 Recovering publication cycles

The limit case of encouraging events burstiness is the deterministic allocation of documents to a cluster based only on their relative positions on the time axis. This is achieved when r is large. In this case, textual information does not matter and only regularities in the time distributions are detected. We illustrate such a case on the News dataset in Fig. 13.

In Fig. 13, we plot on the left the intensity associated with the events for each cluster on the real time axis for $r = 2$. We see that the two most populated clusters follow precise dynamics. We added on the right side of the plot the temporal kernel corresponding to each of these clusters. On the right plot, we retrieve the cause of the daily and weekly cycles observed for the largest cluster on the left plot. The second most populated cluster follows similar dynamics, except that its seems to be shifted of half a day on the real-time axis; the peaks are in a phase opposition with the largest cluster. It is worth noting that the Notre-Dame fire cluster is still detected; this is due to its vocabulary being different enough from the existing cluster's ones to trigger its own cluster, and the associated number of documents being consequent in a short time window. Interestingly, immediately after this cluster emerged, the dynamics on the real-time axis also follow a decaying circadian circle over three days.

4.3 Heuristics

4.3.1 Choosing *r*

We saw that in all the previous experiments, the optimal r was predetermined. In synthetic experiments, a grid-search strategy was used to determine the best r. We did not come up with a way to automatically infer the optimal r without trying several values.

However, we provide some heuristics regarding the tuning of r.

– As r increases, we usually get a smaller number of inferred clusters. This is because considering the temporal dimension adds consistency to the language model; the temporal intensity prior for a new observation is likely to be non-null, which increases the probability of *not* opening a new cluster with respect to a model that does not consider time.
– As r goes to infinity, we only infer one large cluster that comprises all the observations. This is because even the slightest difference in the prior intensities leads to a deterministic cluster choice.

Our leads to automatically determine r involve computing an ad hoc objective metric to optimize jointly with the likelihood. Given there is no gold standard for clustering in real-world processes, the choice of r, and therefore the choice of such metric, is left to the user. As we have shown in Fig. 7 and later in Fig. 10, the choice of r simply tunes the information on which clusters are based. The clustering objective is to be defined for each situation, which is made possible by manual tuning of r. Such an objective could consist in minimizing the clusters' word distribution entropy, the standard deviation of the effective triggering kernel, or the average distance between events within a cluster. A possible procedure for such optimization would involve a multi-arms bandit to deal with this trade-off.

4.3.2 Number of clusters

In previous experiments, we compared our clustering results to the ground truth using the NMI score. We chose this metric because it allows us to compare a different number of clusters together. Indeed, it is seldom the case that PDHP infers exactly the right number of clusters.

Typically, in our synthetic experiments with 2 ground-truth clusters, the number of clusters could differ significantly at the beginning of the algorithm (up to 10 clusters at once for small values of r). However, as the number of observations increases, smaller clusters are discarded as the algorithm converges toward the 2 correct clusters.

In real-world data, the number of clusters can grow very high –even more for small values of r. However, the number of observations each of these clusters comprise seems to follow a power-law distribution. Many of the clusters contain very few observations (5 documents or less); they are leftovers from the process as it converges toward more robust statistics. This is why in Figs. 9 and 12, we restrict ourselves to the study of the largest clusters only.

5 Conclusion

We built the powered Dirichlet–Hawkes process as a generalization of the Dirichlet–Hawkes process and uniform process and showed how it improves performance on various datasets. When textual information conveys little information, or when temporal information conveys little information, and when both do, our model is able to correctly retrieve the original clusters used in the generation process with high accuracy. A central consideration in document clustering is that there are no "right" clusters. For instance, we illustrate how textual content and temporal dynamics can be decorrelated in real-life applications. The framework we developed is flexible enough to allow users to choose the weight they wish to give to temporal or textual information depending on the situation; when textual and temporal clusters are decorrelated, the model allows one to choose which of those to infer.

Many future extensions are possible for PDHP. For instance, it would be interesting to develop its hierarchical version (PHDHP) as it has already been done with HDHP for DHP. Another interesting perspective would be to create a version considering multivariate Hawkes processes to study how textual clusters' dynamics relate to each other. Given several recent works have been based on the regular Dirichlet–Hawkes process, it would be insightful to study how their results vary when using the powered Hawkes–Dirichlet process instead. A study on the influence of the language model used along with PDHP would also be interesting since the text model we used here was simple on purpose (our focus being on the PDHP prior and not on the model it gets associated with).

Finally, it would be interesting to see how this model would work in another context where temporal and textual information are intertwined. For instance, in latent social network inference, we may be able to create clusters according to the observed temporal dynamics of publications, or according to the textual information shared between users, or according to a combination of both.

References

1. Ahmed A, Xing E (2008) Dynamic non-parametric mixture models and the recurrent Chinese restaurant process: with applications to evolutionary clustering. In: SIAM international conference on data mining, pp 219–230
2. Bacry E, Bompaire M, Deegan P, Gaïffas S, Poulsen SV (2017) Tick: a python library for statistical learning, with an emphasis on Hawkes processes and time-dependent models. J. Mach. Learn. Res. 18(1):7937–7941
3. Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: Bengio Y, LeCun Y (eds) 3rd international conference on learning representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings
4. Blei DM, Frazier P (2010) Distance dependent Chinese restaurant processes. In: Proceedings of the 27th international conference on international conference on machine learning, ICML'10, Madison, WI, USA. Omni Press, pp 87–94
5. Blei DM, Lafferty JD (2006) Dynamic topic models. In: Proceedings of the 23rd international conference on machine learning, ICML '06, New York, NY, USA. Association for Computing Machinery, pp 113–120
6. Blei DM, Ng AY, Jordan MI (2003) Latent Dirichlet allocation. J Mach Learn Res 3(null):993–1022
7. Cao J, Sun W (2019) Sequential choice bandits: learning with marketing fatigue. In: AAAI-19
8. Du N, Farajtabar M, Ahmed A, Smola A, Song L (2015) Dirichlet–Hawkes processes with applications to clustering continuous-time document streams. In: 21th ACM SIGKDD international conference on knowledge discovery and data mining
9. Du N, Song L, Smola A, Yuan M (2012) Learning networks of heterogeneous influence. NIPS 4:2780–2788
10. Haralabopoulos G, Anagnostopoulos I (2014) Lifespan and propagation of information in on-line social networks: a case study based on reddit. JNCA 56:88–100
11. Iwata T, Watanabe S, Yamada T, Ueda N (2009) Topic tracking model for analyzing consumer purchase behavior. In: Proceedings of the 21st international joint conference on artificial intelligence, IJCAI'09. Morgan Kaufmann Publishers Inc., pp 1427–1432
12. Karsai M, Jo H-H, Kaski K (2018) Bursty human dynamics. Springer
13. Mavroforakis C, Valera I, Gomez-Rodriguez M (2017) Modeling the dynamics of learning activity on the web. In: Proceedings of the 26th international conference on World Wide Web, WWW '17, pp 1421–1430
14. Myers SA, Leskovec J (2012) Clash of the contagions: cooperation and competition in information diffusion. In: 2012 IEEE 12th international conference on data mining, pp 539–548
15. Poux-Médard G, Velcin J, Loudcher S (2021a) Information interaction profile of choice adoption. In: Oliver N, Pérez-Cruz F, Kramer S, Read J, Lozano JA (eds) Machine learning and knowledge discovery in databases. Research track. Springer, Cham, pp 103–118
16. Poux-Médard G, Velcin J, Loudcher S (2021) Information interactions in outcome prediction: quantification and interpretation using stochastic block models. Association for Computing Machinery, New York, pp 199–208

17. Poux-Médard G, Velcin J, Loudcher S (2021c) Powered Dirichlet process for controlling the importance of "rich-get-richer" prior assumptions in Bayesian clustering. ArXiv
18. Poux-Médard G, Velcin J, Loudcher S (2021d) Powered Hawkes–Dirichlet process: challenging textual clustering using a flexible temporal prior. In: 2021 IEEE international conference on data mining (ICDM), pp 509–518
19. Rathore M, Gupta D, Bhandari D (2018) Complaint classification using word2vec model. Int J Eng Technol (UAE) 7:402–404
20. Tan X, Rao VA, Neville J (2018) The Indian Buffet Hawkes process to model evolving latent influences. In: UAI
21. Wallach H, Jensen S, Dicker L, Heller K (2010) An alternative prior process for nonparametric Bayesian clustering. In: Proceedings of the 13th international conference on artificial intelligence and statistics. JMLR, pp 892–899
22. Wang X, McCallum A (2006) Topics over time: A non-Markov continuous-time model of topical trends. In 12th ACM SIGKDD, KDD '06. Association for Computing Machinery, pp 424–433
23. Welling M (2006) Flexible priors for infinite mixture models. In: Workshop on learning with non-parametric Bayesian methods
24. Xu H, Zha H (2017) A Dirichlet mixture model of Hawkes processes for event sequence clustering. In: Advances in neural information processing systems, vol 30. Curran Associates, Inc
25. Yin J, Chao D, Liu Z, Zhang W, Yu X, Wang J (2018) Model-based clustering of short text streams. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '18, New York. Association for Computing Machinery, pp 2634–2642
26. Yu M, Gupta V, Kolar M (2017) An influence-receptivity model for topic based information cascades. In: 2017 IEEE international conference on data mining (ICDM), pp 1141–1146

Gaël Poux-Médard received a B.Sc. degree from Claude Bernard Lyon 1 University, Lyon, France, in 2017, and an M.Sc. degree from École Normale Supérieure, Lyon, France, in 2019. From 2019 to 2022, he conducted research work as a Ph.D. student in the Data Mining and Decision team of the ERIC laboratory, Université de Lyon, France. His research interests revolve around data mining, computational social sciences, information spread and network analysis.

Julien Velcin is Professor of Computer Science at the University Lumière Lyon 2. He works at the ERIC Lab in the Data Mining and Decision team on topics related to artificial intelligence, machine learning and natural language processing. More precisely, his work aims at designing new models and algorithms to deal with information networks. His research is currently focused on the analysis of topics and opinions that flow through social media.

Sabine Loudcher is a full professor in Computer Science in a research lab of data science and business intelligence of the University of Lyon (France). She carries out research on Data Science especially in Digital Humanities. She is more interested about big data analytics and her main research topics are data lakes and mining data coming from documents or social networks. She is involved in several projects especially in Digital Humanities with Social Sciences researchers.