Learning social network embeddings for predicting information diffusion

Conference Paper · February 2014

DOI: 10.1145/2556195.2556216

Sylvain Lamprier
Sorbonne Université

Ludovic Denoyer
Sorbonne Université

Learning Social Network Embeddings for Predicting Information Diffusion

Simon Bourigault, Cedric Lagnier, Sylvain Lamprier, Ludovic Denoyer, Patrick Gallinari

Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005, Paris, France
CNRS, UMR 7606, LIP6, F-75005, Paris, France

firstname.lastname@lip6.fr

ABSTRACT

Analyzing and modeling the temporal diffusion of information on social media has mainly been treated as a diffusion process on known graphs or proximity structures. The underlying phenomenon results however from the interactions of several actors and media and is more complex than what these models can account for and cannot be explained using such limiting assumptions. We introduce here a new approach to this problem whose goal is to learn a mapping of the observed temporal dynamic onto a continuous space. Nodes participating to diffusion cascades are projected in a latent representation space in such a way that information diffusion can be modeled efficiently using a heat diffusion process. This amounts to learning a diffusion kernel for which the proximity of nodes in the projection space reflects the proximity of their infection time in cascades. The proposed approach possesses several unique characteristics compared to existing ones. Since its parameters are directly learned from cascade samples without requiring any additional information, it does not rely on any pre-existing diffusion structure. Because the solution to the diffusion equation can be expressed in a closed form in the projection space, the inference time for predicting the diffusion of a new piece of information is greatly reduced compared to discrete models. Experiments and comparisons with baselines and alternative models have been performed on both synthetic networks and real datasets. They show the effectiveness of the proposed method both in terms of prediction quality and of inference speed.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Learning; E.1 [**Data**]: Data Structures—*Graphs and networks*

Keywords

Machine learning; Information diffusion; Social networks

WSDM'14, February 24–28, 2014, New York, New York, USA.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2351-2/14/02 ...$15.00.
http://dx.doi.org/10.1145/2556195.2556216.

1. INTRODUCTION

The emergence of Social Networks and Social Media sites has motivated a large amount of recent research. Various generic tasks, such as Social Network Analysis, Social Network annotation, Community Detection or Link Prediction, have been explored. An important research topic is the study of temporal propagation of information through this type of media. It aims at studying how interactions between users, such as sharing a link on Facebook or retweeting something on Twitter, effect the spread of items such as pictures, videos or gossip on the internet. While the study of this word-of-mouth phenomenon pre-dates the development of computer science, the amount of data made available by the growth of online social networks offers an unprecedented opportunity and has enabled new developments (see Section 4 for a review of related works). Most of the initial work in this area is derived from the literature on epidemiology or social science. Different propagation models such as the independent cascade models (IC) [7, 22] or the linear threshold models (LT) [13] have been developed or adapted to internet data. These models attempt to predict and understand the dynamic of observed propagation. Recently, works have also focused on prediction tasks such as *Buzz detection* - predicting whether a particular content will have an important impact over the network [20], *Opinion Leader identification*, detecting whether a node in a network will play an important role in content spread [13, 17], or *Diffusion prediction*, predicting which users (or nodes of the network) will be reached by a given piece of information in the future [19].

Most of the existing propagation models rely on a probabilistic modeling of information diffusion based on explicit relationships between nodes in the network. It has been shown that these models are well adapted when the underlying link structure is representative of the diffusion channels, but may suffer from different drawbacks when used on social networks extracted from the Web [19]. Since they rely on the structure of the network, these models assume that the information only propagates through links between users and usually aim at estimating propagation probabilities along these links. This limits their ability to explain future diffusion since the underlying process is much more complex : diffusion is the result of various interactions between heterogeneous users on several interleaving networks. Users interactions themselves are difficult to detect and quantify [25, 29]. Moreover, theses approaches, which usually come down to learning propagation paths along links of a graph, require a large amount of observations to avoid over-fitting.

Recently, various works have suggested to discover the implicit structure of diffusion from users' behaviours *before* modelling diffusion w.r.t. the extracted graph [29, 8, 25]. These works first find the graph structure that best explains observed diffusion under some hypothesis on the diffusion mechanics, and then use the extracted graph to perform prediction. These approaches are often grounded on a discrete cascading process - i.e., the information iteratively jumps from a user to another following some transfer probabilities on links. Such an iterative process implies successive decisions when inferring diffusion, which may lead to low performance results when mistakes are made at the early steps of the diffusion process. Moreover, inference by such models requires the use of expensive Monte-Carlo simulation techniques to predict the spread of information.

In this paper, we focus on modeling how information diffuses, with the goal of predicting which users will be contaminated by a particular content, knowing the user at the source of the diffusion. Instead of adopting a graph-based approach which would imply dealing with discrete structures, we propose to work in continuous spaces where we learn the temporal dynamics of diffusion from observations. Grounded on the heat diffusion theory, our approach consists in learning heat diffusion kernels that define, for each node of the network, its likelihood to be reached by the propagated information, given the initial source of diffusion. An advantage of this framework is that diffusion does not depend on a prior graph structure, and the model is directly built from observed diffusion cascades. Also, the use of a continuous space representation allows very fast inference when dealing with new cascades. The contribution of this paper is threefold:

- We present an original way to learn diffusion processes by embedding users in a continuous latent space.
- We propose an extension of the model allowing us to take into account the nature of the content being propagated, resulting in differentiated diffusion processes that depend on the features of the information that diffuses.
- We compare this approach with baseline and alternative methods on three corpora extracted from the Web and on synthetic datasets.

The paper is organized as follows: Section 2 defines the notations used throughout the paper, and presents our general approach and the models we propose. In Section 3, we compare our models to several baselines on real and artificial datasets. Section 4 reviews related work on the topic of diffusion in social networks. Finally, Section 5 concludes our works and gives some insight of possible future works.

2. NOTATIONS AND MODEL

Traditionally, diffusion on networks is represented with the notion of *cascade*. A cascade is a sequence of users infected by some information (for instance, it could be the list of users who "liked" a specific YouTube video). A cascade describes to whom and when an item spreads through the network, but not *how* diffusion happens: while it is easy to know *when* a user got infected by some content, it is usually not possible to know *who* infected him.

Given a social network composed of a set of N users[1] $\mathcal{U} = (u_1, ..., u_N)$, cascades correspond to sets of users infected by the propagated information. Depending on the kind of network and the task in concern, the propagated information can for instance correspond to a given topic, a particular url, a specific tag, etc. In the following, we consider $\mathcal{C}$ to be a set of cascades over a given network, divided in two subsets of distinct cascades: $\mathcal{C}_\ell \subseteq \mathcal{C}$ the set of training cascades and $\mathcal{C}_t \subseteq \mathcal{C}$ the set of testing cascades. A cascade $c \in \mathcal{C}$ is defined as:

- A source $s^c \in \mathcal{U}$ which is the user at the source of the cascade - i.e, the first user that published the item concerned by the diffusion.
- A set of contaminated users $S^c \subset \mathcal{U}$ such that $u_i \in S^c$ means that u_i has participated to the cascade c - i.e., the user has performed some action (retweet, like, comment...) that is considered as an infection by c (liking a video, publishing a tweet with a specific hashtag...); $\bar{S^c}$ is the set of users who have not participated in c.
- A contamination timestamp function defined over S^c such that $t^c(u_i)$ corresponds to the timestamp at which $u_i \in S^c$ has first participated in the cascade. We consider that the contamination timestamp of the source is equal to 0.
- A feature vector $q_c \in \mathbb{R}^Q$ that characterizes the content of the cascade c, with Q the size of the *content features space*[2]. This features vector is usually defined as the content of the publication.

2.1 Proposed Approach

The proposed models aim at predicting information diffusion. The central idea of these models is to map the observed information diffusion process into a heat diffusion process in a continuous (euclidean) space. To perform this, we learn diffusion kernels that capture the dynamics of diffusion from a set of training cascades. Let us denote $\mathcal{Z} = \mathbb{R}^n$ an euclidean space of n dimensions - also called *latent space*[3]. Learning such a diffusion kernel comes down in our case to learning a mapping of each node of the network to a particular location in $\mathcal{Z}$ such that, for a given metric, the latent space explains the contamination timestamps observed in the training cascades. Figure 1 depicts a diffusion process where users have been projected in a latent space w.r.t. their timestamps of contamination in training cascades. This approach has several advantages:

- The learning problem is mapped to a continuous optimization problem that can easily be solved using classical optimization methods.
- The propagation model is learned directly from the observations, without the need of a graph structure, and without making strong assumptions about how information propagates.

[1] We talk about users throughout the paper, but the results remain valid for any other kinds of nodes.

[2] For example, when dealing with textual information, the feature vector q_c may be a *tf-idf* vector.

[3] See Section 3 for a discussion concerning the choice of the dimension n

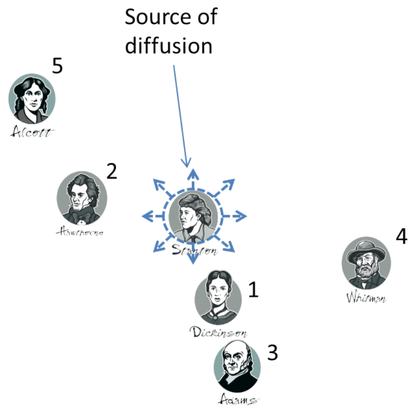

Figure 1: Diffusion in a latent space: each user is associated with a location in this space. The source of the diffusion is at the center of the picture and information spreads uniformly from it in all directions. The numbers indicate the contamination order over the different users according to the modelized diffusion process: the closer a user is from the source, the sooner he is infected by information from the source.

- The inference of the diffusion can be performed in the continuous space, which allows a very fast computation of the prediction without the need of an explicit discrete simulation. Simulation has an expensive processing cost and may lead to unreliable results, with a high variance between results of successive simulations of the same diffusion.
- Finally, the approach allows an easy integration of the content information by making the geometry of the latent space dependent on the information that spreads.

Diffusion Kernel.

Let us consider a geometric manifold $\mathcal{X}$. We define heat diffusion as a function $f(x,t) : \mathcal{X} \times \mathbb{R}^+ \rightarrow \mathbb{R}$ where $f(x,t)$ is the heat at location x at time t. Such a process can be described by the following heat equations:

$$\begin{cases} \frac{\partial f}{\partial t} - \Delta f = 0 \\ f(x,0) = f_0(x) \end{cases} \tag{1}$$

where $f_0(x)$ is the initial condition of the process; Δ is the *Laplace operator*. The heat diffusion kernel is the fundamental solution to these heat diffusion equations in a specified domain with appropriate boundary conditions [11]. We define a diffusion kernel $K(t,y,x)$ such that $K : \mathbb{R}^+ \times \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ computes the heat at location x and time t knowing that the heat source is y. It models the heat diffusion when an initial unit heat is positioned at location y at time $t = 0$. This initial condition corresponds to:

$$K(0,y,x) = \delta(y-x) \tag{2}$$

where δ is the *dirac function*. In an Euclidean space of n dimensions, the diffusion kernel can be written as:

$$K(t,y,x) = (4\pi t)^{-\frac{n}{2}} e^{-\frac{||y-x||^2}{4t}} \tag{3}$$

The diffusion kernel is at the root of our approach and will be used to model how information spreads between nodes of the network.

Learning Diffusion Kernel in a Latent Space.

While some works consider specific cases of heat diffusion based on the structure of the network [14, 17], we want our model to be independent from any predefined explicit network. For that we propose to model the information propagation in an Euclidean space using kernels as defined in equation 3 and to learn these kernels directly from observed cascades. The goal is thus to learn a representation of nodes in this latent space in a way that let the diffusion kernel explain the cascades observed in the training set. This can be seen as a problem of **learning the "best" diffusion kernel** w.r.t. a particular training set of cascades. Let us rewrite the diffusion kernel as a function $K(t, s^c, u_i)$ which returns a value corresponding to the contamination score of node u_i at time t knowing that the source of the contamination - the initial contaminated node - is s^c. We define $Z = (z_{u_1}, ..., z_{u_N})$ such that $z_{u_i} \in \mathbb{R}^n$ denotes the location of user u_i in the latent space $\mathbb{R}^n$. The obtained diffusion kernel is:

$$K_Z(t, s^c, u_i) = (4\pi t)^{-\frac{n}{2}} e^{-\frac{||z_{s^c} - z_{u_i}||^2}{4t}} \tag{4}$$

The problem of modeling how information propagates corresponds to finding the optimal value of Z according to every cascade $c \in \mathcal{C}$. The empirical risk of the model is then defined as:

$$\mathcal{L}(Z) = \sum_{c \in C_l} \Delta(K_Z(., s^c, .), c) \tag{5}$$

where $\Delta(K_Z(., s^c, .), c)$ is a measure of how much, given a source s^c, predictions performed by the diffusion kernel K_Z differs from the observed diffusion in c. Different Δ functions can be defined and we focus on a particular case based on a ranking measure. The final learning problem is an optimization problem which consists in finding Z^* such that:

$$Z^* = argmin_Z \mathcal{L}(Z) \tag{6}$$

Learning Diffusion as a Ranking Problem.

The diffusion kernel models the contamination propensity of *any* node at time t given a particular information source. For learning the kernel function, there is however no full supervision available - this would correspond to a continuous time function giving the heat evolution at any point. The observations only provide the contamination time of the different nodes in a cascade. This partial supervision will be used to constrain the kernel to contaminate the different nodes **in their actual temporal order of infection**.

In practice, we will use the following constraints:

- Given two nodes u_i and u_j such that u_i and u_j are contaminated during cascade c - i.e $u_i \in S^c$ and $u_j \in S^c$ - and respecting $t^c(u_i) < t^c(u_j)$, K_Z must be defined such that $\forall t, K_Z(t, s^c, u_i) > K_Z(t, s^c, u_j)$
- Given two nodes u_i and u_j and a cascade c such that u_i is in S^c and u_j not in S^c, K_Z must be defined such that $\forall t, K_Z(t, s^c, u_i) > K_Z(t, s^c, u_j)$

The constraints basically aim at finding embeddings such that users who are contaminated first are closer to the source of the contamination than users contaminated later (or not contaminated at all). With the expression of K_Z given in equation 4, we can easily rewrite these two constraints as:

$$\begin{aligned}&\forall (u_i, u_j) \in S^c \times S^c, \\ &\qquad t^c(u_i) < t^c(u_j) \Rightarrow ||z_{s^c} - z_{u_i}||^2 < ||z_{s^c} - z_{u_j}||^2 \\ &\forall (u_i, u_j) \in S^c \times \bar{S^c}, \\ &\qquad ||z_{s^c} - z_{u_i}||^2 < ||z_{s^c} - z_{u_j}||^2 \end{aligned} \tag{7}$$

By the use of classical hinge loss functions, these constraints can be handled by defining a ranking objective Δ_{rank} such as:

$$\begin{aligned}&\Delta_{rank}(K_Z(., s^c, .), c) = \\ &\sum_{\substack{u_i, u_j \in S^c \times S^c \\ t^c(u_i) < t^c(u_j)}} max(0, 1 - (||z_{s^c} - z_{u_j}||^2 - ||z_{s^c} - z_{u_i}||^2)) \\ &+ \sum_{u_i, u_j \in S^c \times \bar{S^c}} max(0, 1 - (||z_{s^c} - z_{u_j}||^2 - ||z_{s^c} - z_{u_i}||^2)) \end{aligned} \tag{8}$$

2.2 Learning Algorithm

The final training objective is:

$$\mathcal{L}_{rank}(Z) = \sum_{c \in \mathcal{C}_\ell} \Delta_{rank}(K_Z(., s^c, .), c) \tag{9}$$

We name this model "Content diffusion Kernel" (CDK). Different methods can be used to optimize the objective function. We propose to use a classical stochastic gradient descent method, as illustrated in Algorithm 1, which iterates until a stop criterion is met (typically a number of iterations without significant improvement of the global loss). After having randomly initialized[4] all embeddings for users in $\mathcal{U}$ (line 3), the algorithm samples at each iteration a cascade c from the training set $\mathcal{C}_\ell$ and two users u_i and u_j with u_j a user that is either non-infected, or contaminated after u_i in the diffusion process described by cascade c (lines 6 to 8). If constraints defined in equation 7 are not respected with a sufficient margin[5] for this cascade c and the pair of users u_i and u_j, embeddings z_{u_i}, z_{u_j} and z_{s^c} are modified towards their respective steepest gradient direction with a learning rate α (lines 13 to 15) which is a decreasing function of the number of iterations.

Learning and Inference Complexity.

Let T be the number of iterations. The learning complexity is $O(T \times n)$, where n is the size of the latent space. Once Z has been learned, the inference process is simple. For a cascade $c \in C_t$, we just compute the distance between the user s^c and every other user in $\mathcal{U}$. The inference complexity for every cascade is then $O(N \times n)$, where N is the number of users. Considering that $n \ll N$, this turns out to be much smaller than the complexity of most alternative discrete methods. For instance, the inference step of the very famous Independant Cascade model[6], which is a probabilistic model where diffusion propabilities are defined on edges of the network's graph, requires to consider at each time step of the diffusion t every possible infection situation at previous time $t-1$, which quickly becomes untractable. In practice, inference of graphical models is done by employing a Monte-Carlo approximation that consists in performing a high amount of simulations of the diffusion process starting from the source of the cascade and following the diffusion probabilities on links of the graph. The inference complexity of this approximation of IC is $O(r \times \hat{Succs} \times |\hat{S^c}|)$, where $|\hat{S^c}|$ is the average number of infected nodes in the performed simulations, $\hat{Succs}$ is their average outdegree and r is the number of simulations used for the MCMC approximation. The weaker the probabilities defined on links are, the greater r must be set to obtain a correct approximation of the distribution of final infection probabilities. More information about computation times are given in section 3.

Algorithm 1 Stochastic gradient descent algorithm

```
 1: procedure SGD RANK DIFFUSION KERNEL LEARNING($\mathcal{U}, \mathcal{C}_\ell, T$)
 2:     $\tau \leftarrow 0$
 3:     $\forall u \in \mathcal{U}, z_u^{(\tau)} \leftarrow random$
 4:     while $\tau < T$ do
 5:         $Z^{(\tau+1)} \leftarrow Z^{(\tau)}$
 6:         Sample $c \in \mathcal{C}_\ell$
 7:         Sample $u_i \in S^c$
 8:         Sample $u_j \in \mathcal{U}$ with $t^c(u_i) < t^c(u_j)$ or $u_j \in \bar{S^c}$
 9:         $d_i \leftarrow ||z_{s^c}^{(\tau)} - z_{u_i}^{(\tau)}||^2$
10:         $d_j \leftarrow ||z_{s^c}^{(\tau)} - z_{u_j}^{(\tau)}||^2$
11:         $\delta_d \leftarrow d_j - d_i$
12:         if $\delta_d < 1$ then
13:             $z_{u_i}^{(\tau+1)} \leftarrow z_{u_i}^{(\tau)} + \alpha(\tau) \times 2(z_{s^c}^{(\tau)} - z_{u_i}^{(\tau)})$
14:             $z_{u_j}^{(\tau+1)} \leftarrow z_{u_j}^{(\tau)} + \alpha(\tau) \times 2(z_{u_j}^{(\tau)} - z_{s^c}^{(\tau)})$
15:             $z_{s^c}^{(\tau+1)} \leftarrow z_{s^c}^{(\tau)} + \alpha(\tau) \times 2(z_{u_i}^{(\tau)} - z_{u_j}^{(\tau)})$
16:         end if
17:         $\tau \leftarrow \tau + 1$
18:     end while
19:     $Z \leftarrow Z^{(\tau)}$
20: end procedure
```

2.3 Content-based Diffusion Kernel

We now propose an extension of the previous model able to take into account the content of each cascade by considering that different contents will propagate differently in the network. Our goal is thus to model different propagation schemes depending on the source and also on the content of the information that spreads. Our extension is based on the following ideas: (i) First, the propagation will still be modeled by a diffusion kernel in a latent space - each user corresponding to a particular location. (ii) Second, the content will influence **the metric** of the latent space - i.e instead of being isotropic around the source node projection for any content, propagation in the latent space will also depend on the content. Each possible content will then correspond to a specific metric in the latent space resulting in differentiated propagation schemes. The metrics and the users locations will be learned simultaneously from training cascades. In this work, the content metric has been developed in the way such that the content operates as an offset affecting the lo-

[4] Different initialization strategies can be adopted. In our experiments, we used an uniform initialization between -1 and 1.

[5] As defined by the hinge loss function, see equation 8.

[6] We present details and results of this model in section 3

cation of the source in the latent space[7] This model, named *Content-based Source Diffusion Kernel (CSDK)*, is described in the following.

We consider a parametrized function called *content embedding function* and denoted $f_\theta : \mathbb{R}^Q \rightarrow \mathbb{R}^n$. It will map any content information into a particular location in the latent space, θ being the set of parameters of this function[8]. The function will map two different contents q_c and $q_{c'}$ to two different locations $f_\theta(q_c)$ and $f_\theta(q_{c'})$ in the latent space as illustrated in Figure 2. Let us define the new diffusion kernel as a function $K^{CSDK}_{Z,\theta}(q^c, t, s^c, u_i)$ which measures the contamination of user u_i at time t knowing that the source of the diffusion is s^c and the content of the cascade is $q^c \in \mathbb{R}^Q$. In order to consider both the source of the contamination and the content that diffuses, based on the *content embedding function* f_θ, we propose to model K^{CSDK} such that:

$$K^{CSDK}_{Z,\theta}(q^c, t, s^c, u_i) = (4\pi t)^{-\frac{n}{2}} e^{-\frac{||z_{s^c} + f_\theta(q^c) - z_{u_i}||^2}{4t}} \tag{10}$$

The location of the source $z_{s^c} + f_\theta(q^c)$ now depends on both the latent representation of the source user s^c and on the embedded content $f_\theta(q^c)$. Two different contents will thus correspond to two different initial locations, resulting in two different diffusion kernels - see Figure 2.

The content embedding function and the users locations will be learned simultaneously, resulting in learning problem that consists in minimizing on both θ and Z the following objective function:

$$\mathcal{L}_{CSDK}(Z) = \sum_{c \in \mathcal{C}_\ell} \Delta_{rank}(K^{CSDK}_{Z,\theta}(q^c, ., s^c, .), c) \tag{11}$$

The final learning problem can thus be written as:

$$\Delta_{rank}(K^{CSDK}_{Z,\theta}(q^c, ., s^c, .), c) = \\ \sum_{\substack{u_i \in S^c \\ u_j \in S^c \\ t^c(u_i) < t^c(u_j)}} max(0, 1 - (||z_{s^c} + f_\theta(q^c) - z_{u_j}||^2 - ||z_{s^c} + f_\theta(q^c) - z_{u_i}||^2)) \\ + \sum_{\substack{u_i \in S^c \\ u_j \in \bar{S^c}}} max(0, 1 - (||z_{s^c} + f_\theta(q^c) - z_{u_j}||^2 - ||z_{s^c} + f_\theta(q^c) - z_{u_i}||^2)) \tag{12}$$

and is optimized using a stochastic gradient descent method similar to the one presented previously.

3. EXPERIMENTS

3.1 Datasets

We tested our models on several datasets from various online sources as well as artificial ones.

Real-world datasets.

We have used three datasets extracted from the Web:

- The first dataset comes from the International AAAI Conference on Weblogs and Social Media 2009 (*ICWSM*) which published a corpus containing 44 millions blog posts collected over a 1-year period [4]. We consider each blog to be a "user" in the social network and cascades are composed of sets of posts which are linked to each other: each connected component of the posts graph is a cascade. A cascade is then represented by a set of users (authors of the posts composing the cascade) and by the timestamps at which they have been infected (timestamps of the posts). We also extracted an oriented graph using hyperlinks : if there is at least one link from a post of user a to a post of user b in the training set, we create a link $b \rightarrow a$. This graph is needed by some of the baseline models used in the experiments, but not by our models.

- The second dataset is extracted from *Memetracker* corpus described in [16]. This corpus contains articles from news websites and blogs, collected during the 2008 US presidential campaign. The corpus is built by tracing the flow of short phrases or *memes* through the web. This dataset is similar to ICWSM and we have defined users, cascades and the social graph the same way. The main difference is the lack of posts content, so we did not apply the CSDK model on this dataset.

- The third dataset is extracted from Digg, which is a collaborative news portal on which users can post links to *stories* (articles, blog posts, videos...). Other users can then "digg" these stories if they like it. Stories appear on the front page of Digg based on the amount of "diggs" they have. We use stories as cascades, each "digg" given by a user being considered as a user contamination. We used the Digg stream API to collect the *complete* Digg history (every single story posted, all diggs, and all comments) during a one month time window. We built a graph, which will be used by the IC and Graph diffusion baselines, in the following way:

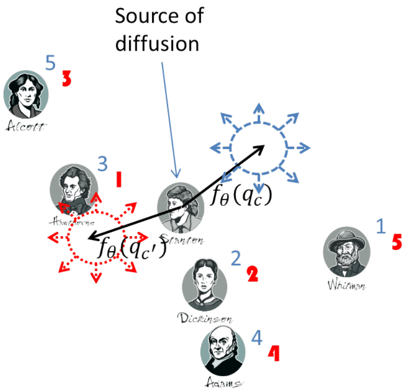

Figure 2: CDSK Diffusion model: The source of the diffusion is a translation of the source user by a vector $f_\theta(q^c)$ that depends on the content of the cascade. Here, two different contents q_c and $q_{c'}$ spread from the source user and correspond to two different embedded locations $f_\theta(q_c)$ and $f_\theta(q_{c'})$. Even if they have the same source, these two cascades thus corresponds to two different source locations in the latent space. The two numbers near each user indicate the contamination order by the two contents.

[7] different possibilities have been tested and this one offered a good compromise

[8] We consider here that f_θ is a linear function

for each user a who has digged a post created by user b, we create a link $b \to a$ in the graph.

We filtered the users of each dataset to keep about 5000 users with the most posts. Table 1 gives some statistics about the datasets sizes.

Synthetic Data.
In order to better understand how the CSDK model handles the content information, we have also generated synthetic datasets for which we can control the correlation between the content information and the diffusion behavior. In order to generate such datasets, we consider that any cascade content is composed of *one* word w in a set of Q possible words. For each word, the diffusion follows a particular IC model denoted E_w where E_w is the transition matrix between nodes - i.e the diffusion probability. The Q transitions matrices are generated randomly: 99 % of the values in these matrices are equal to 0 and, all remaining values are sampled between 0 and 0.01, resulting in sparse matrices. To generate a new cascade, we first randomly choose a source user u and a content word w, and use the corresponding transition matrix E_w to generate the cascade. By making the value of Q higher, we obtain more complex propagation schemes. Note that if $Q = 1$, our generation method corresponds to a classical single IC model. In these datasets we have used 10000 cascades as training cascades, and 10000 cascades for test with a set of 1000 users.

3.2 Evaluation Measures

For each dataset the cascade set C is divided in two subsets C_ℓ and C_t, for training and testing purposes. Our goal is to retrieve, for each cascade in the testing set, which user(s) will eventualy be infected. This can be seen as an Information Retrieval task, with cascades as "queries" and users as "documents". We evaluate the performance through Mean Average Precision (MAP) and Precision-Recall curves. For every cascade in the testing set, each model predicts a "contamination score" for each user in the testing set, indicating how likely that user is to be infected by this cascade. We then sort users in descending order and use Mean Average Precision to evaluate our performance as it is done in [6]. Let $\sigma_{c,k}$ be the rank of user u_k for cascade c. Let $P_{c,k}$ be the precision at rank k for cascade c, i.e. the percentage of infected users among the top k users in the ranking order. Mean Average precision is defined as:

$$MAP = \frac{1}{|\mathcal{C}_t|} \sum_{c \in \mathcal{C}_t} \frac{\sum_{u_k \in S^c} P_{c,\sigma_{c,k}}}{|S^c|}$$

We also use Precision-Recall curves to visualize performances. Most cascades do not reach more than 1 or 2 users leading to a small number of recall points. We then show the average precision at each recall point instead of precision at a recall value. Consequently, only a few cascades contain a lot of users and there is a higher variance for high recall points.

Note that each experiment has been done 10 times and the results are the average over the 10 different runs.

3.3 Baselines

We compare our models to several baselines and state-of-the-art models. We first used two naive baselines:

- **Nb_App**: For each user, we count the proportion of cascades in the training set he participates in. We then use that value as an infection score for that user in every cascade in the testing set. It corresponds to a propensity to be infected by any cascade.
- **Mean_rank**: In this model, the contamination score of any user u_i is the inverse of the average rank of this user in the training cascades: the sooner u_i tends to be infected by cascades in the training set, the higher his infection score for cascades from the testing set.

In addition to these baselines, we compare our approaches to state-of-the-art models:

- **IC model**: We implemented the classic independent cascade model (IC) which is usually used in the literature as a comparison method. IC uses the social graph, and works in a discrete way: when a user u_i becomes infected at time step t, every neighbour u_j of u_i has a probability $p_{i,j}$ to become infected at time step $t + 1$. In the learning phase, all $p_{i,j}$ are learnt in an EM-fashion [22]. Since this is a stochastic model, it is hard to compute the actual probability for a user to become infected at some point in the process. We thus use a Monte-Carlo approximation: given an initialy infected user, we run a large number of simulations. The final contamination score of each user is equal to the number of simulations in which he was infected. Note that, temporal extensions of the IC model have been proposed, but since they did not allow better results in this experimental protocol [15] they were not been used in our experiments.
- **Netrate**: We show results for the *exponential* version of the NetRate model described in [8]. NetRate is used here as a state-of-the-art method which does not need the knowledge of the network structure to predict how information propagates, which is also the case of our approaches. Note that different variants of this model exists (*power law* and *rayleigh* - [8]) that obtain similar results. NetRate is briefly described in Section 4.
- **Graph Diffusion**: At last, we also compare our methods with the model proposed in [17, 11] which is based on a graph diffusion kernel that shares some similarities with the methods we propose. In this model, instead of *learning the best kernel* as we do, the authors of the paper define a particular kernel over the structure of the network. This kernel models the fact that the temperature of any node in the graph diffuses equally on the different outgoing links. In comparison to our approach, the diffusion model is not learned over the set of training cascades, and clearly depends on the knowledge of the network structure.

We do not compare to models like [15] or [23] which also make use of the content because they need a user profile which is not available in our datasets.

3.4 Results

3.4.1 Models without content

Table 2 shows the mean average precision (MAP) for all models on the 3 real datasets. First, we can see that all other baselines models perform worse than the IC model.

	Nb. of Users	Nb. of Links	Nb. of train Cascades	Nb. of test Cascades	Avg cascade size
meme	5000	4372	2377	600	2.17
icwsm	5000	17746	19027	4711	2.22
digg	4751	71263	150000	66744	2.43

Table 1: Some statistics about our real datasets.

Model	n	Memetracker	ICWSM	Digg
CDK	5	0,176	0,660	0.170
	10	0.257	0.721	0.212
	30	0.344	0.769	0.273
	50	0.355	0.774	0.285
	100	0.347	0.771	0.282
	200	0.357	**0.776**	0.302
	500	0.363	0.773	0.280
CSDK	5	-	0.605	0.255
	10	-	0.663	0.304
	30	-	0.714	0.348
	50	-	0.731	**0.352**
	100	-	0.744	0.352
	200	-	0.732	0.350
	500	-	0.748	0.351
IC		0.372	0.712	0.197
Netrate		0.287	0.187	0.162
Graph Diff.		**0.374**	0.483	0.082
Nb_App		0.180	0.112	0.077
Mean_Rank		0.187	0.121	0.206

Table 2: Results on 3 real datasets: *Memetracker*, *ICWSM* and *Digg*. Results of CDK and CSDK are given for several values of n, the dimension of the latent space $\mathcal{Z}$.

This is not surprising for *Nb_App* and *Mean_Rank* which are based on naive heuristics. The low performance of the NetRate model is due to the fact that the learning process it relies on requires too large amounts of training data to avoid over-training when used to predict information diffusion. At last, the Graph diffusion approach outperforms IC on the MemeTracker dataset but gives lower performance on the two other corpora showing that the assumptions it is based on are less adapted to the three datasets than the IC assumption concerning how information spreads.

If we compare our approach (CDK) with IC, we see that on the *Memetracker* and *ICWSM* datasets they both obtain similar results, and CDK clearly outperforms IC on the *Digg* dataset. Note, that, like NetRate, our method is not based on any knowledge concerning the structure of the network meaning that this algorithm is able to do as well or better than IC, using less information. The performance of the CDK model also depends on the size of the latent space: a smaller space gives lower predictions quality, while a larger space can tend to overfit.

Concerning the inference times, the CDK model takes around 15 minutes to infer all scores for all cascades on the *Digg* dataset. In comparison, IC model needs more than 1 day and NetRate model, which is one complexity degree above IC, takes a few days. All these experiments have been done on a standard desktop computer.

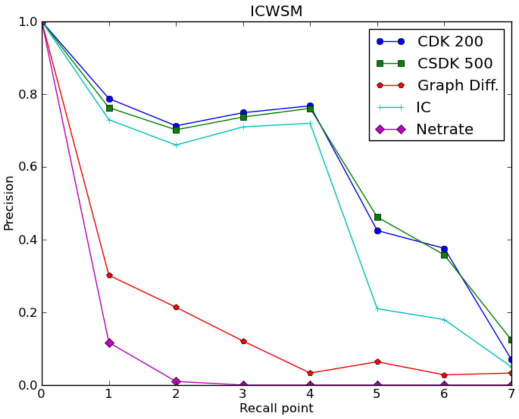

Figure 3: Precision at recall points for the main models on the ICWSM dataset. Results of CDK and CSDK are given for specific latent space dimensionality.

3.4.2 *Integration of content*

When comparing the Content based approach (CSDK) with the other models (Table 2), we can see that the integration of the content degrades the performance on the ICWSM dataset, but clearly increases the predictions quality over the Digg dataset. Actually, the quality of this model depends on the information given by the content of each cascade; in the case of ICWSM, the content is noisy due to the way it has been captured (using RSS feeds that only give part of the content of the blog posts), while the Digg content is clearly more informative since it is composed of the full content of news articles.

In order to further explore and better understand how the CSDK model depends on the quality of the content, we have performed experiments on synthetic datasets with different content sizes (see Table 3). First, one can see that the performance of all the algorithms degrades when the variance of the content (number of words considered) increases. The task becomes more and more complex when multiple content information is considered. CDSK degrades less than all other approaches and is still able to obtain good performance. It always outperforms state-of-the-art methods. These experiments show that CSDK is more robust to a complex content-dependent propagation scheme than classical approaches.

Finally, Figures 3 and 4 gives more details concerning the precision of each model at different recall points - we have only drawn the ”best” versions of CDK and CSDK. On these curves, one can see that our methods obtain a better precision than classical methods, and that CSDK clearly outperforms CDK on the Digg dataset.

Model	n	5 words	10 words	20 words	30 words	40 words	50 words
CDK	10	0.323	0.205	0.147	0.111	0.102	0.098
	30	0.422	0.301	0.207	0.146	0.128	0.121
	50	0.414	0.304	0.207	0.158	0.136	0.128
	100	0.430	0.304	0.210	0.155	0.140	0.126
CSDK	10	0.394	0.243	0.184	0.139	0.135	0.124
	30	0.605	0.442	0.301	0.218	0.200	0.179
	50	0.615	0.466	**0.346**	0.259	0.234	0.219
	100	**0.631**	**0.469**	0.343	**0.271**	**0.248**	**0.228**
IC		0.482	0.317	0.211	0.163	0.125	0.111
Netrate		0.289	0.150	0.175	0.137	0.017	0.017
Graph Diff.		0.308	0.091	0.081	0.084	0.073	0.076
Nb_App		0.118	0.101	0.088	0.085	0.079	0.081
Mean_Rank		0.209	0.196	0.165	0.160	0.151	0.143

Table 3: Results (MAP values) of our models and baselines n artificial datasets generated with different number of words. Results of CDK and CSDK are given for several values of n, the dimension of the latent space $\mathcal{Z}$.

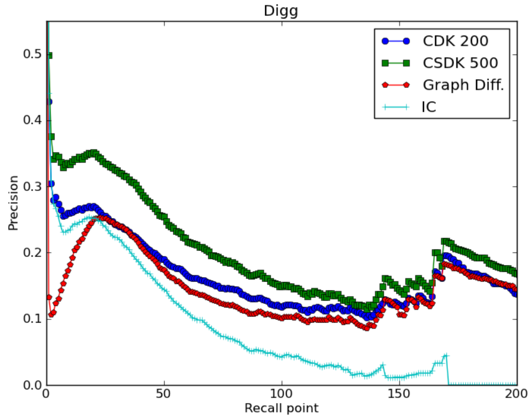

Figure 4: Precision at recall points for the main models on the Digg dataset. Results of CDK and CSDK are given for specific latent space dimensionality.

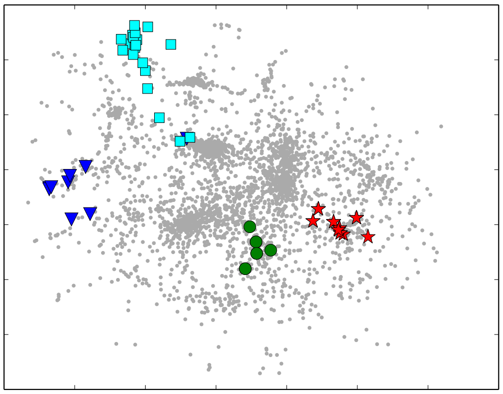

Figure 5: The Digg dataset users in 2D. Light gray dots represent users, and groups of identical symbols represent four cascades from the testing set.

3.5 Discussion

While information diffusion in social network is a widely studied topic (see next section), the concept has not been clearly defined. The phenomenon itself is quite rare: only a very small fraction of the information generated each day on the internet will become popular and ”viral”. In our datasets, most cascades only reach a small percentage of the user set (see table 1). Because of this sparsity, learning the dynamics of users interactions turns out to be quite difficult. In this paper, we project users in a euclidean space in which we use a distance to represent the diffusion. This gives our CDK model an important property: for any triplet of users (u_i, u_j, u_k), we have the triangle inequality $||z_i - z_j||^2 \leq ||z_i - z_k||^2 + ||z_k - z_j||^2$. This means that if users u_i and u_j never interact each with the other in training but both interacts with some third user u_k, our model tends to set u_i and u_j to be relatively close. A model like IC is unable to learn such a property. In order to better visualize this ability to regroup users with similar activities, we have trained a CDK model with a latent space of size $n = 2$ on the Digg corpus and propose to visualize how users have been projected onto this space (Figure 5). We have highlighted users involved in different randomly chosen cascades extracted from the test set. From this figure, we can see that our model naturally tends to build clusters, each cluster corresponding to a group of users generally involved in the same cascades. We suppose that this ability to group users with the ”same behavior” opens the door to other usages of our method, particularly concerning the use of CDK and CSDK for visualization softwares that could allow people to understand the different spreading schemes for a particular dataset. In the CDK model, using a distance also mean that we consider diffusion to be symetric, i.e. diffusion from u_i to u_j is identical to diffusion from u_j to u_i. This is a strong hypothesis which has been recently discussed [3]. We are currently experimenting an extension of this work where users are projected at different locations of the space whether they are senders or receivers of the propagated content.

4. RELATED WORK

Historically, the diffusion process has been studied in the context of product adoption in [1]. In this work, the author models product adoption by consumers as a function of time with two parameters: the influence of word-of-mouth and the weight of a marketing campaign. In the early 2000s, the availability of large amount of internet data enabled researchers to suggest methods based on the social graph like the Independent Cascade model (IC) [7, 22] and the Linear Threshold models (LT) [13], both modeling a user-to-user contamination process. Since then, the rapid growth of social websites like Facebook or Twitter has triggered many new developments.

Several extensions of IC and LT have been proposed. For instance, [21] proposed an asynchronous extension of the IC model (ASIC) which enables IC to integrate the temporal dimension. In [17, 11], a heat diffusion process occurring on the social graph is used to model interactions between users. As we have seen in this paper, many cascades cannot be explained using only user interactions. In [23], the authors take users profiles into account to infer the probability of diffusion. [12] or [15] use information content and user profiles to infer the contamination. The same idea has been used in [26] which integrates the content of tweets to predict the probabilities of diffusion between users. It is important to note that all these models make the assumption that the graph on which the propagation occurs is known. This turns out to be a strong hypothesis: the social graph defined by an online social network (friends, followers, subscriptions...) is often incomplete, irrelevant [25] or unknown. To overcome this limitation, two main families of methods have been studied.

- A first family consists of link prediction methods: given an online social network population and a set of observations (shopping habits, movie reviews, hashtags usage...), the goal is to predict a set of links (followers, friends, influencers...) that best explains the observed user actions. These models have not been designed specifically for diffusion prediction, but they all model the propagation process to infer the most plausible links. NetInf [9] and then Connie [18] use a greedy algorithm to find a fixed number of links between users that maximize the likelihood of a set of observed diffusions under an IC-like diffusion hypothesis. A more general framework have been proposed in [8] where the Netrate model, used in our experiments as a baseline, is used to predict the user-to-user contamination. Netrate is a cascade model like IC: it aims at finding propagation probabilities between pairs of users. The first improvement with regard to IC is that they do not use the social graph, they directly infer probabilities from observed diffusions. Secondly, they use an exponential delay to infer the time after which the diffusion occurs. These works have later been extended in [10]. Recently, [25] used transfer entropy to compute the inter-user influence and infer a graph containing the most "predictive" links.

- The second family makes use of statistical learning instead of using graph-based approaches. One very simple yet efficient method is to study the relation between the number of infected users after a short period of time and after a longer period [24]. [29] predicts the volume of diffusion based on the infection time of a selected subset of users.

The models we have introduced in this article do not need any social graph and are based on a new approach where the propagation is modeled as a heat diffusion process in a continuous latent space. Heat diffusion processes, and particularly diffusion kernels, have been studied recently for different applications: classification [14], dimensionality reduction [27], and also to rank nodes [28]. The work that is closest to our is the one in [17] where the authors use a diffusion kernel to select marketing candidates. The main difference is that we learn the diffusion kernel from data when they use a predefined graph kernel.

At last, the idea of projecting discrete relational data onto a continuous space had already been proposed for different tasks. For example, in [5], the authors propose to learn embeddings for the nodes of a graph such that the resulting distances between vectors is "as close as possible" to the original distance between vertices in the graph. More recently, learning embeddings has also been used for coding relational databases [2].

5. CONCLUSION

We have presented a new family of information diffusion models based on the heat diffusion kernel. Their originality is to formulate diffusion as a process in a continuous space, built using an embedding of the nodes learned from observed cascades. These models have some interesting characteristics 1) they learn directly from the observations and therefore do not require a predefined diffusion graph structure which is often not available for social applications. 2) they run much faster (1 or 2 orders of magnitude) than classical discrete models due to the continuous context and 3) they allow an easy integration of the content by modifying the geometry of the latent space. Performance obtained on real-world and artificial datasets show the ability of these methods to model information spread, and to take into account the content information in the diffusion process. They are competitive with and sometimes better than state of the art reference models.

Two research directions are currently considered: the first one consists in developing alternative models for a better use of the content information. Particularly, we are studying **metric learning** paradigms that should offer new possibilities to incorporate this information in the geometry of the latent space. A second direction consists in applying these methods to other diffusion tasks not restricted to social networks.

6. ACKNOWLEDGMENTS

This work has been partially supported by the DIFAC FUI project and the ANR (French National Research Agency) MLVIS project

7. REFERENCES

[1] F. M. Bass. A new product growth for model consumer durables. *Management Science*, 15:215–227, 1969.

[2] A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *AAAI*, 2011.

[3] R. Bosagh Zadeh, A. Goel, K. Munagala, and A. Sharma. On the precision of social and information networks. In *Proceedings of the first ACM conference on Online social networks*, pages 63–74. ACM, 2013.

[4] K. Burton, A. Java, and I. Soboroff. The icwsm 2009 spinn3r dataset. In *Proceedings of the Third Annual Conference on Weblogs and Social Media*, May 2009.

[5] M. Chen, Q. Yang, and X. Tang. Directed graph embedding. In *IJCAI*, pages 2707–2712, 2007.

[6] W. Feng and J. Wang. Retweet or not?: personalized tweet re-ranking. In *Proceedings of the sixth ACM international conference on Web search and data mining*, WSDM '13. ACM, 2013.

[7] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing letters*, 12(3):211–223, 2001.

[8] M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, ICML '11, pages 561–568. ACM, 2011.

[9] M. Gomez Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, New York, NY, USA, 2010. ACM.

[10] M. Gomez-Rodriguez, J. Leskovec, and B. Schölkopf. Modeling information propagation with survival theory. In *ICML*, 2013.

[11] A. Grigoryan. *Heat Kernel and Analysis on Manifolds.* AMS/IP Studies in Advanced Mathematics. American Mathematical Society, 2009.

[12] A. Guille and H. Hacid. A predictive model for the temporal dynamics of information diffusion in online social networks. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion. ACM, 2012.

[13] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 137–146. ACM, 2003.

[14] R. I. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, volume 2, pages 315–322, 2002.

[15] C. Lagnier, L. Denoyer, E. Gaussier, and P. Gallinari. Predicting information diffusion in social networks using content and user's profiles. In *European Conference on Information Retrieval*, ECIR '13, 2013.

[16] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 497–506, New York, NY, USA, 2009. ACM.

[17] H. Ma, H. Yang, M. R. Lyu, and I. King. Mining social networks using heat diffusion processes for marketing candidates selection. In *Proceedings of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 233–242, New York, NY, USA, 2008. ACM.

[18] S. A. Myers and J. Leskovec. On the convexity of latent social network inference. *CoRR*, abs/1010.5504, 2010.

[19] A. Najar, L. Denoyer, and P. Gallinari. Predicting information diffusion on social networks with partial knowledge. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 1197–1204, New York, NY, USA, 2012. ACM.

[20] D. M. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 695–704. ACM, 2011.

[21] K. Saito, M. Kimura, K. Ohara, and H. Motoda. Generative models of information diffusion with asynchronous timedelay. *Journal of Machine Learning Research - Proceedings Track*, 13:193–208, 2010.

[22] K. Saito, R. Nakano, and M. Kimura. Prediction of information diffusion probabilities for independent cascade model. In *Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems, Part III*, KES '08, pages 67–75. Springer-Verlag, 2008.

[23] K. Saito, K. Ohara, Y. Yamagishi, M. Kimura, and H. Motoda. Learning diffusion probability based on node attributes in social networks. In M. Kryszkiewicz, H. Rybinski, A. Skowron, and Z. W. Ras, editors, *ISMIS*, volume 6804 of *Lecture Notes in Computer Science*, pages 153–162. Springer, 2011.

[24] G. Szabo and B. A. Huberman. Predicting the popularity of online content. *Communications of the ACM*, 53(8):80–88, 2010.

[25] G. Ver Steeg and A. Galstyan. Information-theoretic measures of influence based on content dynamics. In *Proceedings of the sixth ACM international conference on Web search and data mining*, WSDM '13, pages 3–12, New York, NY, USA, 2013. ACM.

[26] L. Wang, S. Ermon, and J. E. Hopcroft. Feature-enhanced probabilistic models for diffusion network inference. In *Proceedings of the 2012 European conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II*, ECML PKDD'12, pages 499–514. Springer-Verlag, 2012.

[27] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the twenty-first international conference on Machine learning*, page 106. ACM, 2004.

[28] H. Yang, I. King, and M. R. Lyu. Diffusionrank: a possible penicillin for web spamming. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 431–438. ACM, 2007.

[29] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, ICDM '10, pages 599–608, Washington, DC, USA, 2010. IEEE Computer Society.