Unsupervised Streaming Feature Selection in Social Media

Jundong Li[†], Xia Hu[‡], Jiliang Tang[◇] and Huan Liu[†]
[†]Computer Science and Engineering, Arizona State University, Tempe, AZ, USA
[‡]Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA
[◇]Yahoo! Labs, Yahoo! Inc, Sunnyvale, CA, USA
{jundong.li, huan.liu}@asu.edu, hu@cse.tamu.edu, jlt@yahoo-inc.com

ABSTRACT

The explosive growth of social media sites brings about massive amounts of high-dimensional data. Feature selection is effective in preparing high-dimensional data for data analytics. The characteristics of social media present novel challenges for feature selection. First, social media data is not fully structured and its features are usually not predefined, but are generated dynamically. For example, in Twitter, slang words (features) are created everyday and quickly become popular within a short period of time. It is hard to directly apply traditional batch-mode feature selection methods to find such features. Second, given the nature of social media, label information is costly to collect. It exacerbates the problem of feature selection without knowing feature relevance. On the other hand, opportunities are also unequivocally present with additional data sources; for example, link information is ubiquitous in social media and could be helpful in selecting relevant features. In this paper, we study a novel problem to conduct unsupervised streaming feature selection for social media data. We investigate how to exploit link information in streaming feature selection, resulting in a novel unsupervised streaming feature selection framework USFS. Experimental results on two real-world social media datasets show the effectiveness and efficiency of the proposed framework comparing with the state-of-the-art unsupervised feature selection algorithms.

Categories and Subject Descriptors

D.2.8 [**DATABASE MANAGEMENT**]: Database Applications—*Data Mining*

Keywords

Unsupervised Feature Selection; Streaming Features; Social Media Data

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806501.

1. INTRODUCTION

The rapid growth and popularity of social media services such as Twitter[1], Facebook[2] provide a platform for people to perform online social activities by sharing information and communicating with others. Massive amounts of high-dimensional data (blogs, posts, images, etc.) are user generated and quickly disseminated. It is desirable and of great importance to reduce the dimensionality of social media data for many learning tasks due to the curse of dimensionality. One way to resolve this problem is feature selection [17, 20], which aims to select a subset of relevant features for a compact and accurate representation.

Traditional feature selection assumes that all features are static and known in advance. However, this assumption is invalid in many real-world applications especially in social media which is imbued with high-velocity streaming features. In social media, features are generated dynamically, new features are sequentially added and the size of features is unknown in most cases. For example, Twitter produces more than 320 millions of tweets everyday and a large amount of slang words (features) are continuously being user generated. These slang words promptly grab users' attention and become popular in a short time. It is not practical to wait until all features are available before performing feature selection. Another example is that after earthquakes, topics (features) like "Nepal" emerge as hot topics in social media shortly, traditional batch-mode feature selection can hardly capture and select such features timely. Therefore, it could be more appealing to perform *streaming feature selection* (SFS) [37] to rapidly adapt to the changes.

In SFS, the number of instances is considered to be constant while candidate features arrive one at a time, the task is to timely select a subset of relevant features from all features seen so far [37]. Instead of searching for the whole feature space which is costly, SFS processes a new feature upon its arrival. A general framework of streaming feature selection is presented in Figure 1. At each time step, a typical SFS algorithm first determines whether to accept the most recently arrived feature; if the feature is added to the selected feature set, it then determines whether to discard some existing features from the selected feature set. The process repeats until no new features show up anymore. The vast majority of existing streaming feature selection algorithms are supervised which utilize label information to guide feature selection process [27, 33, 37]. However, in social media, it is easy to amass vast quantities of unlabeled

[1]https://twitter.com/
[2]https://www.facebook.com/

data, while it is time and labor consuming to obtain labels. To deal with large-scale unlabeled data in social media, we propose to study unsupervised streaming feature selection. Unsupervised streaming feature selection is particularly difficult and challenging: (1) without any label information, it is difficult to assess the importance of features; and (2) features are usually not predefined but are generated dynamically, hence it cannot be carried out by directly applying traditional unsupervised feature selection algorithms.

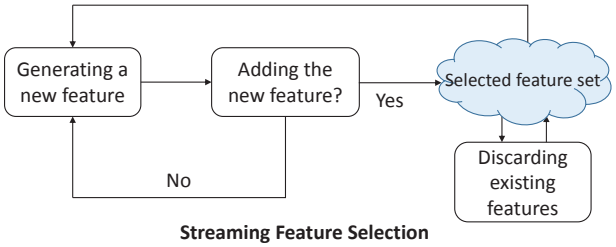


Figure 1: A framework of streaming feature selection. It consists of two phases: testing newly arrived features and testing existing features.

On the other hand, link information is abundant in social media. As observed by homophily [21] from social sciences, linked instances are likely to share similar features (or attributes). Therefore, as label information for supervised streaming feature selection, link information could provide helpful constraints to enable unsupervised streaming feature selection. However, linked social media data is inherently not independent and identically distributed (*i.i.d.*) while existing streaming feature selection are based on the data *i.i.d.* assumption, it is challenging to exploit link information for streaming feature selection.

In this work, we investigate: (1) how to exploit link information for feature selection; and (2) how to perform streaming feature selection in unsupervised scenarios. Our solutions to these two questions lead to a novel unsupervised streaming feature selection framework USFS. The main contributions of this paper are outlined as follows:

- Providing a principled approach to utilize link information to enable unsupervised streaming feature selection in social media;
- Proposing an unsupervised streaming feature selection framework, USFS, which exploits link and feature information simultaneously to select features dynamically and efficiently; and
- Evaluating the efficacy and efficiency of the proposed USFS framework on real-world social media datasets.

The rest of this paper is organized as follows. We formally define the problem of unsupervised streaming feature selection in Section 2. In Section 3, we introduce the proposed framework of unsupervised streaming feature selection. Empirical evaluation on real-world social media datasets is presented in Section 4 with discussion. In Section 5, we briefly review related work. The conclusion and future work are presented in Section 6.

2. PROBLEM STATEMENT

We first summarize some notations used in this paper. We use bold uppercase characters to denote matrices, bold lowercase characters to denote vectors, normal lowercase characters to denote scalars. For an arbitrary matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$, $\mathbf{a}_i$ and $\mathbf{a}^j$ mean the i-th row and j-the column of matrix $\mathbf{A}$, respectively. $\mathbf{A}_{ij}$ or $\mathbf{a}_i^j$ denotes the (i,j)-th entry of matrix $\mathbf{A}$. $\mathbf{A}^{(t)}$ denotes the matrix of $\mathbf{A}$ at time step t, $(\mathbf{a}^{(t)})_i$ and $(\mathbf{a}^{(t)})^j$ represent i-th row and j-th column of matrix $\mathbf{A}^{(t)}$, respectively. $Tr(\mathbf{A})$ is the trace of matrix $\mathbf{A}$ if it is square, the Frobenius norm of the matrix $\mathbf{A} \in \mathbb{R}^{n\times d}$ is defined as $||\mathbf{A}||_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{d}\mathbf{A}_{ij}^2}$.

Let $\mathcal{U} = \{u_1, u_2, ..., u_n\}$ denote a set of n linked data instances. We assume features are dynamically generated and one feature arrives at each time step, thus, at time step t, each linked data instance is associated with a set of t features $\mathcal{F}^{(t)} = \{f_1, f_2, ..., f_t\}$. Then at the next time step $t+1$, each linked instance is tied with a new feature set $\mathcal{F}^{(t+1)} = \{f_1, f_2, ..., f_t, f_{t+1}\}$. The data representation at time step t and $t+1$ can be represented as $\mathbf{X}^{(t)} = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_t]$ and $\mathbf{X}^{(t+1)} = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_t, \mathbf{f}_{t+1}]$, where $\mathbf{f}_1, ..., \mathbf{f}_t, \mathbf{f}_{t+1}$ are the feature vectors corresponding to features $f_1, ..., f_t, f_{t+1}$. We denote the link information between instances in a matrix $\mathbf{M} \in \mathbb{R}^{n\times n}$, where $\mathbf{M}_{ij} = 1$ if u_i and u_j are linked, otherwise $\mathbf{M}_{ij} = 0$. The link information can either be a directed or an undirected graph. Note that in this paper, we do not consider the dynamics of link information and the reason is that link information does not change as fast as feature information; for example, the friend circles of most users are often stable once they are established.

With these notations, the task of *unsupervised streaming feature selection in social media* focuses on finding a subset of most relevant features $\mathcal{S}^{(t)} \subseteq \mathcal{F}^{(t)}$ at each time step t to facilitate clustering by utilizing both the feature information $\mathcal{F}^{(t)}$ and the link information $\mathbf{M}$.

3. UNSUPERVISED STREAMING FEATURE SELECTION IN SOCIAL MEDIA

The work flow of the proposed framework USFS is shown in Figure 2. We can observe that it consists of three components. The first component shows the representation of data. We have a set of linked instances (for example $u_1, u_2, ..., u_5$); for each linked instance, its features arrive through a streaming fashion, for example, $u_1, u_2, ..., u_5$ are associated with features $f_1, f_2..., f_t$ at time step t; are associated with features $f_1, f_2, ..., f_{t+i}$ at time step $t+i$. The second component shows the process of the algorithm, we will first talk about how to model link information via extracting social latent factors and how to use them as a constraint through a regression model in Section 3.1. Then we will introduce how to model feature information to make it consistent with social latent factors in Section 3.2. At last, we will show how to efficiently test new feature and existing features in Section 3.3. After that, as shown in the third component, we obtain a subset of relevant features at each time step (for example, $\mathcal{S}^{(t)}$ at time step t).

3.1 Modeling Link Information

Social media users connect to each other due to different factors such as movie fans, football enthusiasts, colleagues and each factor should be related to certain features (or at-

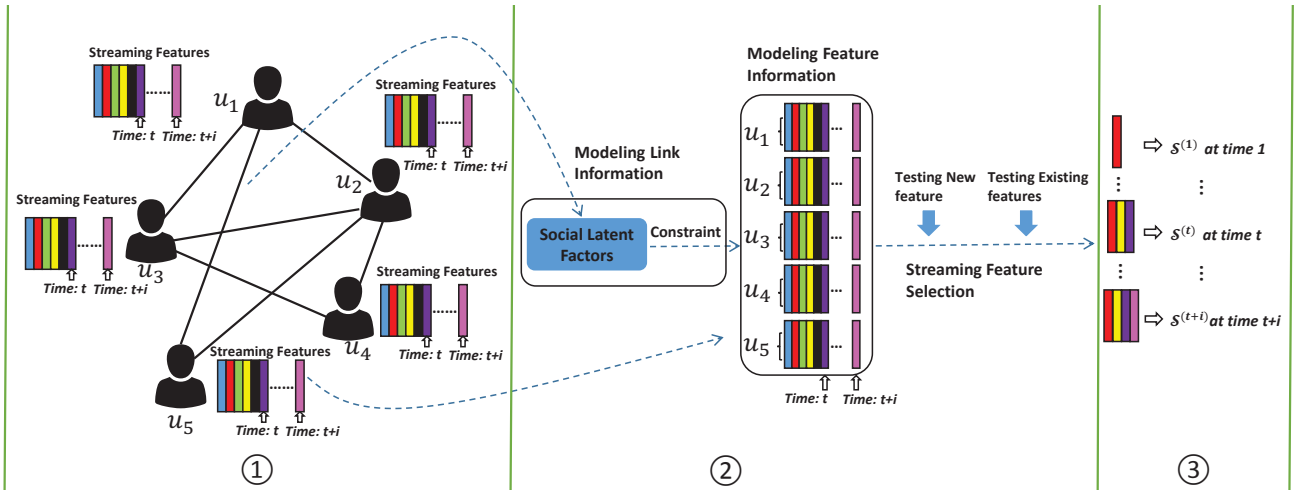


Figure 2: A framework of unsupervised streaming feature selection in social media.

tributes) of users [19]. Therefore, extracting these factors from link information should be very useful to steer the unsupervised streaming feature selection. However, in most cases, these hidden factors are not explicitly available in social media data.

Uncovering hidden social factors has been extensively studied [2, 22, 30]. In this work, we extract the social latent factors for each instance based on the mixed membership stochastic blockmodel [2]. In the blockmodel, it assumes that there exists a number of latent factors, and these latent factors interact with each other with certain probabilities to form social relationships. More specially, each instance is associated with a k-dimensional latent factor vector $\boldsymbol{\pi}_i \in \mathbb{R}^k$, where $\boldsymbol{\pi}_{ig}$ denotes the probability of u_i in factor g. This means that each instance can simultaneously be sided with multiple latent factors with different affiliation strength. For each instance, the indicator vector $\mathbf{z}_{i\rightarrow j}$ denotes the latent factor membership of u_i when it links to u_j and $\mathbf{z}_{i\leftarrow j}$ denotes the latent factor membership of u_i when it is linked from u_j. The interaction strength between different latent factors is encoded in a $k \times k$ stochastic matrix $\mathbf{B}$, in which each element is between 0 and 1. Then the observed link information is generated according to the following process:

1. For each linked instance u_i,
   - Draw a k dimensional vector $\boldsymbol{\pi}_i \sim \text{Dirichlet}(\theta)$.
2. For each pair of linked instance $(i, j) \in \mathcal{U} \times \mathcal{U}$,
   - Draw indicator vector $\mathbf{z}_{i\rightarrow j} \sim \text{Multinomial}(\boldsymbol{\pi}_i)$.
   - Draw indicator vector $\mathbf{z}_{i\leftarrow j} \sim \text{Multinomial}(\boldsymbol{\pi}_j)$.
   - Draw the relationship between u_i and u_j, $\mathbf{M}_{i,j} \sim \text{Bernoulli}(\mathbf{z}_{i\rightarrow j}\mathbf{B}\mathbf{z}_{i\leftarrow j})$.

Motivated by [12], we use a scalable inference algorithm to obtain the social latent factors $\boldsymbol{\Pi} = [\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, ..., \boldsymbol{\pi}_n]^T \in \mathbb{R}^{n\times k}$ for all n instances efficiently.

As we obtain the social latent factors for each linked instances, we take advantage of them as a constraint to perform feature selection through a regression model. We measure the importance of each feature by its ability to differentiate different social latent factors. At time step t, given each social latent factor $\boldsymbol{\pi}^i$ (a column of $\boldsymbol{\Pi}$) for all instances, we are able to find a subset of most relevant features by the following minimization problem:

$$\begin{aligned}\min_{\mathbf{W}^{(t)}} \mathcal{J}(\mathbf{W}^{(t)}) &= \frac{1}{2}\sum_{i=1}^{k} ||\mathbf{X}^{(t)}(\mathbf{w}^{(t)})^i - \boldsymbol{\pi}^i||_2^2 + \alpha \sum_{i=1}^{k} \|(\mathbf{w}^{(t)})^i\|_1 \\ &= \frac{1}{2}||\mathbf{X}^{(t)}\mathbf{W}^{(t)} - \boldsymbol{\Pi}||_F^2 + \alpha \sum_{i=1}^{k} \|(\mathbf{w}^{(t)})^i\|_1,\end{aligned} \tag{1}$$

where $\mathbf{X}^{(t)} \in \mathbb{R}^{n\times t}$, $\mathbf{W}^{(t)} \in \mathbb{R}^{t\times k}$ is a mapping matrix which assigns each instance a k-dimensional social latent vector at time step t. Each column of $\mathbf{W}^{(t)}$, i.e., $(\mathbf{w}^{(t)})^i \in \mathbb{R}^t$ contains coefficients of t different features in approximating the i-th social latent vector of $\boldsymbol{\Pi}$. α is a parameter which controls the trade-off between the loss function and the ℓ_1-norm. One main advantage of ℓ_1-norm regression (Lasso) [31] is that it leads some coefficients of $(\mathbf{w}^{(t)})^i$ to be exact zero. This property makes it to be suitable for feature selection, as we can select features with corresponding non-zero coefficients.

In Lasso, the number of selected features is usually bounded by the number of data instances, which is unrealistic in many applications. Besides, features in social media usually have strong pairwise correlations, such as synonyms or antonyms words in text data. Lasso tends to randomly select features from a group and discards the others. Therefore, we employ the elastic net [38] on the basis of Eq. (1):

$$\begin{aligned}\min_{\mathbf{W}^{(t)}} \mathcal{J}(\mathbf{W}^{(t)}) = &\frac{1}{2}||\mathbf{X}^{(t)}\mathbf{W}^{(t)} - \boldsymbol{\Pi}||_F^2 + \alpha \sum_{i=1}^{k} \|(\mathbf{w}^{(t)})^i\|_1 \\ &+ \frac{\beta}{2}||\mathbf{W}^{(t)}||_F^2,\end{aligned} \tag{2}$$

where the regularization term $\frac{\beta}{2}||\mathbf{W}^{(t)}||_F^2$ controls the robustness of the learned model.

3.2 Modeling Feature Information

In Twitter, if two users post similar contents (features), they are more likely to share similar social latent factors, like hobbies, education background, etc. The similarity of social latent factors reflects the correlation of two linked in-

stances in the feature space. In other words, social latent factors of two instances are more likely to be consistent when their feature similarity (like textual similarity) is high. To model the feature information, we first construct a graph $\mathcal{G}$ to represent the feature similarity between different data instances. The adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ of the graph $\mathcal{G}$ at time step t is defined as:

$$\mathbf{A}_{ij}^{(t)} = \begin{cases} 1 & \text{if } (\mathbf{x}^{(t)})_i \in \mathcal{N}_p((\mathbf{x}^{(t)})_j) \text{ or } (\mathbf{x}^{(t)})_j \in \mathcal{N}_p((\mathbf{x}^{(t)})_i) \\ 0 & \text{otherwise} \end{cases}$$

where $(\mathbf{x}^{(t)})_i$ indicates the feature information of u_i, $\mathcal{N}_p((\mathbf{x}^{(t)})_i)$ represents p-nearest neighbors of $(\mathbf{x}^{(t)})_i$. Then feature information can be modeled by minimizing the following term:

$$\begin{aligned} &\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{A}_{ij}^{(t)}||(\mathbf{X}^{(t)}\mathbf{W}^{(t)})_i - (\mathbf{X}^{(t)}\mathbf{W}^{(t)})_j||_2^2 \\ &= Tr((\mathbf{X}^{(t)}\mathbf{W}^{(t)})^T(\mathbf{D}^{(t)} - \mathbf{A}^{(t)})(\mathbf{X}^{(t)}\mathbf{W}^{(t)})) \\ &= Tr((\mathbf{X}^{(t)}\mathbf{W}^{(t)})^T\mathbf{L}^{(t)}(\mathbf{X}^{(t)}\mathbf{W}^{(t)})), \end{aligned} \tag{3}$$

where $\mathbf{D}^{(t)} \in \mathbb{R}^{n\times n}$ is a diagonal matrix with $\mathbf{D}_{ii}^{(t)} = \sum_{j=1}^{n}\mathbf{A}_{ij}^{(t)}$, $\mathbf{L}^{(t)} = \mathbf{D}^{(t)} - \mathbf{A}^{(t)}$ is the Laplacian matrix. Since the Laplacian matrix in Eq. (3) is positive-semidefinite, Eq. (3) can also be written as:

$$\begin{aligned} &Tr((\mathbf{X}^{(t)}\mathbf{W}^{(t)})^T\mathbf{L}^{(t)}(\mathbf{X}^{(t)}\mathbf{W}^{(t)})) \\ &= ||(\mathbf{X}^{(t)}\mathbf{W}^{(t)})^T(\mathbf{L}^{(t)})^{\frac{1}{2}}||_F^2. \end{aligned} \tag{4}$$

The optimization formulation, which integrates feature information, is defined as:

$$\begin{aligned} \min_{\mathbf{W}^{(t)}} \mathcal{J}(\mathbf{W}^{(t)}) = &\frac{1}{2}||\mathbf{X}^{(t)}\mathbf{W}^{(t)} - \mathbf{\Pi}||_F^2 + \alpha\sum_{i=1}^{k}\|(\mathbf{w}^{(t)})^i\|_1 \\ &+ \frac{\beta}{2}||\mathbf{W}^{(t)}||_F^2 + \frac{\gamma}{2}||(\mathbf{X}^{(t)}\mathbf{W}^{(t)})^T(\mathbf{L}^{(t)})^{\frac{1}{2}}||_F^2, \end{aligned} \tag{5}$$

where γ is the regularization parameter to balance link information and feature information.

3.3 Streaming Feature Selection Framework

The objective function in Eq. (5) at time step t is parameterized by a transformation matrix $\mathbf{W}^{(t)}$. It can be further decomposed into a series of k sub-problems which correspond to k social latent factors:

$$\begin{aligned} \min_{(\mathbf{w}^{(t)})^i} \mathcal{J}((\mathbf{w}^{(t)})^i) = &\frac{1}{2}||\mathbf{X}^{(t)}(\mathbf{w}^{(t)})^i - \boldsymbol{\pi}^i||_2^2 + \alpha\|(\mathbf{w}^{(t)})^i\|_1 \\ &+ \frac{\beta}{2}||(\mathbf{w}^{(t)})^i||_2^2 + \frac{\gamma}{2}||(\mathbf{X}^{(t)}(\mathbf{w}^{(t)})^i)^T(\mathbf{L}^{(t)})^{\frac{1}{2}}||_2^2, \end{aligned} \tag{6}$$

where $i = 1, ..., k$. By solving each sub-problem in Eq. (6), we can select a subset of features at time t. Next we introduce how to efficiently perform feature selection when a new feature f_{t+1} is generated at a new time step $t+1$. Following common steps of supervised streaming feature selection [33], the proposed framework will test: (1) whether we should select the new feature; and (2) whether we should discard some existing features.

3.3.1 Testing New Features

It can be observed from Eq. (6) that at time step $t+1$, incorporating a new feature feature f_{t+1} involves adding a new non-zero weight value $(\mathbf{w}^{(t+1)})_{t+1}^i$ to the model, which incurs a penalty increasing $\alpha\|(\mathbf{w}^{(t+1)})_{t+1}^i\|_1$ on the ℓ_1 regularization term. The addition of the new feature f_{t+1} reduces the overall objective function value in Eq. (6) only when the overall reduction from the first, third, and forth term outweighs the increase of ℓ_1 penalty $\alpha\|(\mathbf{w}^{(t)})_{t+1}^i\|_1$.

Motivated by [26, 27], we adopt a stagewise way to check newly arrived features. Let $\mathcal{J}((\mathbf{w}^{(t+1)})^i)$ denotes the objective function of Eq. (6) at time step $t+1$:

$$\begin{aligned} \min_{(\mathbf{w}^{(t+1)})^i} \mathcal{J}((\mathbf{w}^{(t+1)})^i) = &\frac{1}{2}||\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i||_2^2 \\ &+ \alpha\|(\mathbf{w}^{(t+1)})^i\|_1 + \frac{\beta}{2}||(\mathbf{w}^{(t+1)})^i||_2^2 \\ &+ \frac{\gamma}{2}||(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i)^T(\mathbf{L}^{(t+1)})^{\frac{1}{2}}||_2^2, \end{aligned} \tag{7}$$

then the derivative of $\mathcal{J}((\mathbf{w}^{(t+1)})^i)$ with respect to $(\mathbf{w}^{(t+1)})_{t+1}^i$ is as follows:

$$\begin{aligned} &\frac{\partial\mathcal{J}((\mathbf{w}^{(t+1)})^i)}{\partial(\mathbf{w}^{(t+1)})_{t+1}^i} \\ &= [(\mathbf{X}^{(t+1)})^T(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i) + \beta(\mathbf{w}^{(t+1)})^i \\ &+ \gamma(\mathbf{X}^{(t+1)})^T\mathbf{L}^{(t+1)}\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i]_{t+1} + \alpha\text{sign}((\mathbf{w}^{(t+1)})_{t+1}^i) \\ &= [(\mathbf{X}^{(t+1)})^T(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i) + \beta(\mathbf{w}^{(t+1)})^i \\ &+ \gamma(\mathbf{X}^{(t+1)})^T\mathbf{L}^{(t+1)}\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i]_{t+1} \pm \alpha. \end{aligned} \tag{8}$$

In Eq. (8), the derivative of ℓ_1-norm term $\alpha\|(\mathbf{w}^{(t+1)})^i\|_1$ w.r.t $(\mathbf{w}^{(t+1)})_{t+1}^i$ is not smooth. Here we discuss the sign of the derivative, i.e., $\text{sign}(\mathbf{w}^{(t+1)})_{t+1}^i$. When the new feature f_{t+1} arrives, we first set its feature coefficient $(\mathbf{w}^{(t+1)})_{t+1}^i$ to be zero and add it to the model, if:

$$\begin{aligned} &[(\mathbf{X}^{(t+1)})^T(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i) + \beta(\mathbf{w}^{(t+1)})^i \\ &+ \gamma(\mathbf{X}^{(t+1)})^T\mathbf{L}^{(t+1)}\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i]_{t+1} - \alpha > 0, \end{aligned} \tag{9}$$

it is easy to verify that:

$$\frac{\partial\mathcal{J}((\mathbf{w}^{(t+1)})^i)}{\partial(\mathbf{w}^{(t+1)})_{t+1}^i} > 0. \tag{10}$$

In order to reduce the objective function value $\mathcal{J}((\mathbf{w}^{(t+1)})^i)$, we need to slightly reduce the value of $(\mathbf{w}^{(t+1)})_{t+1}^i$ to make it negative, and then the sign of $(\mathbf{w}^{(t+1)})_{t+1}^i$ will be negative. For the same reason, if:

$$\begin{aligned} &[(\mathbf{X}^{(t+1)})^T(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i) + \beta(\mathbf{w}^{(t+1)})^i \\ &+ \gamma(\mathbf{X}^{(t+1)})^T\mathbf{L}^{(t+1)}\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i]_{t+1} + \alpha < 0, \end{aligned} \tag{11}$$

then:

$$\frac{\partial\mathcal{J}((\mathbf{w}^{(t+1)})^i)}{\partial(\mathbf{w}^{(t+1)})_{t+1}^i} < 0, \tag{12}$$

the sign of $(\mathbf{w}^{(t+1)})_{t+1}^i$ will be positive. If both of previous conditions are not satisfied, it is impossible to reduce the objective function value $\mathcal{J}((\mathbf{w}^{(t+1)})^i)$ by making $(\mathbf{w}^{(t+1)})_{t+1}^i$ as a small disturbance around 0. In other words, for the new feature f_{t+1}, we need to check:

$$\begin{aligned} &|[(\mathbf{X}^{(t+1)})^T(\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i - \boldsymbol{\pi}^i) + \beta(\mathbf{w}^{(t+1)})^i \\ &+ \gamma(\mathbf{X}^{(t+1)})^T\mathbf{L}^{(t+1)}\mathbf{X}^{(t+1)}(\mathbf{w}^{(t+1)})^i]_{t+1}| > \alpha. \end{aligned} \tag{13}$$

As the condition in Eq. (13) is satisfied, it indicates that the addition of the new feature f_{t+1} will reduce the objective function value $\mathcal{J}((\mathbf{w}^{(t+1)})^i)$, therefore the new feature is included in the model described in Eq. (7).

3.3.2 Testing Existing Features

In social media, when new features are continuously being generated, they may take place of some existing features since new features can better reflect the interests of users, etc. Old features become outdated as a result, therefore, in the proposed unsupervised streaming feature selection framework USFS, we also investigate if it is necessary to remove any existing selected features.

After a new feature is accepted and added to the model, we optimize Eq. (7) with respect to existing feature weights, the optimization may force some feature weights to be zero. If the feature weight obtains a zero value, it indicates that the existence of the feature is not likely to reduce the objective function value and the feature can be removed. Here we discuss how to solve the optimization problem in Eq. (7). The objective function in Eq. (7) is convex and the gradient with respect to $(\mathbf{w}^{(t+1)})^i_{t+1}$ can be easily obtained as Eq. (8), then a global optimum solution can be achieved. We choose to use a Broyden-Fletcher-Goldfarb-Shanno (BFGS) quasi-newton method [3] to solve the optimization problem. Unlike traditional Newton's method which requires the calculation of second derivatives (the Hessian), BFGS only needs the gradient of the objective function to be computed at each iteration. Therefore, it is more efficient than Newton's methods especially when Hessian evaluation is slow.

The minimization problem in Eq. (7) can be generalized to the following form:

$$\min f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n. \tag{14}$$

At each iteration, the optimal solution $\mathbf{x}$ is updated as:

$$\mathbf{x}_{m+1} = \mathbf{x}_m - \delta_m \mathbf{H}_m \mathbf{g}_m, \tag{15}$$

where $\mathbf{H}_m = \mathbf{B}_m^{-1}$, $\mathbf{B}_m$ is an approximation to the Hessian matrix ($\mathbf{B}_m \approx [\nabla^2 f(\mathbf{x}_m)]$), $\mathbf{g}_m = \nabla f(\mathbf{x}_m)$ is the gradient and δ_m is the step size that can be determined by line search. Let the vectors $\mathbf{s}_m$ and $\mathbf{c}_m$ be:

$$\mathbf{s}_m = \mathbf{x}_{m+1} - \mathbf{x}_m, \ \mathbf{c}_m = \mathbf{g}_{m+1} - \mathbf{g}_m, \tag{16}$$

the next Hessian approximation has to meet the secant equation:

$$\mathbf{B}_{m+1}\mathbf{s}_m = \mathbf{c}_m. \tag{17}$$

By pre-multiplying the secant equation $\mathbf{s}_m^T$ at both sides, we obtain the curvature condition:

$$\underbrace{\mathbf{s}_m^T \mathbf{B}_{m+1} \mathbf{s}_m}_{>0} = \mathbf{s}_m^T \mathbf{c}_m > 0. \tag{18}$$

If the curvature condition is satisfied, $\mathbf{B}_{m+1}$ in the secant equation has at least one solution, which can be updated by the following way:

$$\mathbf{B}_{m+1} = \mathbf{B}_m + \frac{\mathbf{c}_m \mathbf{c}_m^T}{\mathbf{c}_m^T \mathbf{s}_m} - \frac{\mathbf{B}_m \mathbf{s}_m \mathbf{s}_m^T \mathbf{B}_m}{\mathbf{s}_m^T \mathbf{B}_m \mathbf{s}_m}. \tag{19}$$

Its inverse, i.e., $\mathbf{H}_{m+1}$, can be updated efficiently by Sherman-Morrison formula [10]:

$$\mathbf{H}_{m+1} = \mathbf{H}_m - \frac{\mathbf{s}_m \mathbf{c}_m^T \mathbf{H}_m + \mathbf{H}_m \mathbf{c}_m \mathbf{s}_m^T}{\mathbf{s}_m^T \mathbf{c}_m} + (1 + \frac{\mathbf{c}_m^T \mathbf{H}_m \mathbf{c}_m}{\mathbf{s}_m^T \mathbf{c}_m}) \frac{\mathbf{s}_m \mathbf{s}_m^T}{\mathbf{s}_m^T \mathbf{c}_m}. \tag{20}$$

With these, the BFGS algorithm to solve Eq. (7) is illustrated in Algorithm 1.

Algorithm 1 BFGS to optimize Eq. (7)

Input: Starting point $\mathbf{x}_0$, convergence threshold ϵ, initial inverse Hessian approximation $\mathbf{H}_0$
Output: Optimal solution $\mathbf{x}^*$

1: $m \leftarrow 0$
2: $\mathbf{g}_m = \nabla f(\mathbf{x}_m)$
3: **while** $||\mathbf{g}_m|| > \epsilon$ **do**
4: Obtain a direction $\mathbf{p}_m = -\mathbf{H}_m \mathbf{g}_m$
5: Compute $\mathbf{x}_{m+1} = \mathbf{x}_m + \delta_m \mathbf{p}_m$, where δ_m is chosen by line search to meet curvature condition
6: $\mathbf{g}_{m+1} = \nabla f(\mathbf{x}_{m+1})$
7: $\mathbf{s}_m = \mathbf{x}_{m+1} - \mathbf{x}_m$
8: $\mathbf{c}_m = \mathbf{g}_{m+1} - \mathbf{g}_m$
9: $\mathbf{H}_{m+1} = \mathbf{H}_m - \frac{\mathbf{s}_m \mathbf{c}_m^T \mathbf{H}_m + \mathbf{H}_m \mathbf{c}_m \mathbf{s}_m^T}{\mathbf{s}_m^T \mathbf{c}_m} + (1 + \frac{\mathbf{c}_m^T \mathbf{H}_m \mathbf{c}_m}{\mathbf{s}_m^T \mathbf{c}_m}) \frac{\mathbf{s}_m \mathbf{s}_m^T}{\mathbf{s}_m^T \mathbf{c}_m}$
10: $m \leftarrow m + 1$
11: **end while**
12: **return** $\mathbf{x}_m$

3.3.3 Feature Selection by USFS

By solving all k sub-problems at time step $t+1$, we obtain the sparse coefficient matrix $\mathbf{W} = [(\mathbf{w}^{(t+1)})^1, ..., (\mathbf{w}^{(t+1)})^k]$. Since we solve each sub-problem separately, the number of non-zero weights in each $(\mathbf{w}^{(t+1)})^i (i = 1, ..., k)$ is not necessarily to be the same. For each feature f_j, if any of the k corresponding feature weight coefficients $(\mathbf{w}^{(t+1)})^i_j (i = 1, 2, ..., k)$ is nonzero, the feature is included in the final model, otherwise the feature is not selected. If f_j is selected, its feature score at time step $t+1$ is defined as:

$$FScore(j)^{(t+1)} = \max((\mathbf{w}^{(t+1)})^1_j, ..., (\mathbf{w}^{(t+1)})^k_j) \tag{21}$$

The selected features are then sorted according to their feature scores in a descending order, the higher the feature score, the more important the feature is.

The pseudo code of the proposed unsupervised streaming feature selection algorithm for social media data is illustrated in Algorithm 2. It efficiently performs unsupervised feature selection when a new feature f_{t+1} arrives. In line 1, we obtain the social latent factor matrix $\mathbf{\Pi}$ using the link information $\mathbf{M}$. The algorithm to check new feature and existing features is illustrated in lines 2-8. More specifically, for each sub-problem, we first check the gradient condition, this step decides whether we accept the new feature (line 3). If the condition is satisfied (line 4), the new feature is included in the model (line 5) and the model is re-optimized with respect to all existing feature weights (line 6). At last, when the new feature is included in the model, it updates the Laplacian matrix (line 10), calculates the feature scores, and updates the selected feature set (lines 11-12).

3.4 Time Complexity Analysis

3.4.1 Time Complexity for All Streaming Features

The mixed membership stochastic model to extract social latent factors has a time complexity of $O(n^2k^2)$. Assuming the total number of streaming features is t and the number of obtained features is s, the time complexity of updating

Algorithm 2 Unsupervised streaming feature selection framework (USFS)

Input: New feature f_{t+1} at time $t+1$, feature weight matrix $\mathbf{W}^{(t)}$ at previous time step t, link information $\mathbf{M}$, parameters α, β, γ, number of social latent factors k, number of nearest neighbors p
Output: Selected feature subset $\mathcal{S}^{(t+1)}$ at time step $t+1$

```
Obtain social latent factors Π from M
for each social latent factor π^l (l = 1, ..., k) do
    compute gradient g for f_{t+1} according to Eq. (8)
    if abs(g) > α then
        add feature f_{t+1} to the model
        optimize the model via BFGS in Algorithm 1
    end if
end for
if feature f_{t+1} is accepted then
    update Laplacian matrix L^{(t+1)}
    obtain feature scores according to Eq. (21)
    sort features by scores and update S^{(t+1)}
end if
return S^{(t+1)}
```

Laplacian matrix is bounded by $O(n^2st)$. At each time step, we check the gradient condition in Eq. (13). The time complexity upper bound of the gradient checking over all t time steps is $O(n^2kst)$. Since we optimize the model in Eq. (7) when the new feature is accepted, the total time of optimization with BGFS is $O(n^2s^2t)$ in the worst case when the selected s features are the latest arrived s features.

Overall, the total time complexity of the proposed USFS is $O(n^2k^2) + O(n^2st) + O(n^2kst) + O(n^2s^2t)$. Since $k \ll t$ and $s \ll t$, the upper bound of the overall time complexity is $O(n^2s^2t)$. However, it only provides an upper bound, in real-world applications, the time complexity could be much lower than this upper bound. We will empirically show the efficiency of the proposed framework in the experiments.

3.4.2 Time Complexity for An Individual Feature

For the newly generated feature, suppose there are already s features in the model, if its previous feature is added in the model, the time complexity of gradient test is $O(n^2ks)$, otherwise the time complexity is only $O(n)$. To test existing features via BFGS, the time complexity is $O(n^2s^2)$.

4. EXPERIMENTS

In this section, we conduct experiments to evaluate the performance of the proposed framework USFS. In particular, we evaluate the following: 1) How is the quality of selected features by USFS compared with the state-of-the-art unsupervised feature selection algorithms? 2) How efficient is the proposed USFS framework? Before introducing the details of experiments, we first introduce the datasets and experimental settings.

4.1 Datasets

We use two real-world social media datasets BlogCatalog and Flickr for experimental evaluation. Some statistics of the datasets are listed in Table 1.

BlogCatalog: BlogCatalog[3] is a social blog directory which manages bloggers and their blogs. Bloggers are associated with sets of tags, which provide feature information. Users in blogcatalog follow each other which form the social link information. Bloggers can also register their blogs under predefined categories, which are used as ground truth for validation in our work.

	BlogCatalog	Flickr
# of Users	5,196	7,575
# of Features	8,189	12,047
# of Links	171,743	239,738
# of Ave Degree	66.11	63.30
# of Classes	6	9

Table 1: Detailed information of datasets.

Flickr: Flickr[4] is an image hosting and sharing website, the key features in flickr are tags, user can specify the list of tags to reflect their interests. Similar to blogcatalog, users in flickr can interact with others. Photos are organized under prespecified categories, which are used as the ground truth.

4.2 Experimental Settings

Following a standard way to assess unsupervised feature selection [4, 18, 29, 35], we use the clustering performance to evaluate the quality of selected features. Two commonly used clustering performance evaluation metrics, i.e., *accuracy* (ACC) and *normalized mutual information* (NMI) [4] are used in this paper.

To the best of our knowledge, we are the first to study streaming feature selection in social media. To investigate the effectiveness and the efficiency of the proposed framework, we choose the following state-of-the-art unsupervised feature selection algorithms as baseline methods:

- **LapScore**: Laplacian score [16] evaluates feature importance by its ability to preserve the local manifold structure of data.
- **SPEC**: Features are selected by spectral analysis [36] and SPEC can be considered as an extension of Laplacian score method.
- **NDFS**: Nonnegative Discriminative Unsupervised Feature Selection [18] which selects features via a joint nonnegative spectral analysis as well as a $\ell_{2,1}$-norm regularization.
- **LUFS**: LUFS utilizes both content information and link information to perform feature selection in an unsupervised scenario [29].

For LapScore, NDFS and USFS, we follow previous work [4, 16] to specify the number of neighborhood size to be 5 to construct the Laplacian matrix on the data instances. NDFS and LUFS have different regularization parameters, we set these regularization parameters according to the suggestions from the original papers. For USFS, we set the number of social latent factors as the number of clusters. There are three important regularization parameters α, β and γ in USFS. α controls the sparsity of the model, β is the parameter for elastic net which controls the robustness of the model, and γ balances the contribution of the link information and feature information. In the experiments, we empirically set $\alpha = 10$,

[3]http://www.blogcatalog.com/

[4]https://www.flickr.com/

$\beta = 0.1$, $\gamma = 0.1$ and more details about the effects of these parameters on the proposed framework will be discussed in Section 4.5.

All experiments are conducted on a machine with 16GB RAM and Intel Core i7-4770 CPU.

4.3 Quality of Selected Features

Following streaming feature selection settings that assume features arrive one at a time [37], we divide all features into 9 groups where we choose the first $\{20\%, ..., 90\%, 100\%\}$ as streaming features. In each group, we perform feature selection with traditional unsupervised feature selection algorithm as well as the proposed USFS algorithm. We record how many features USFS selects and specify the same number as that of selected features by traditional unsupervised feature selection algorithms for a fair comparison.

After obtaining the feature selection results, K-means clustering is performed based on the chosen features. We repeat the K-means algorithm 20 times and report average results because K-means may converge to local minima. The clustering results are evaluated by both *accuracy* (ACC) and *normalized mutual information* (NMI). The higher the ACC and NMI values are, the better feature selection performance is. The comparison results are shown in Table 2 and Table 3 for BlogCatalog and Flickr, respectively. Note that the number in parentheses in the table indicates the number of selected features determined by USFS. We make the following observations:

- USFS tends to accept new features at the very beginning, then it becomes increasingly difficult for newly generated features to alternate previous decisions since existing features already provide us enough information. For example, no new features are accepted anymore after the number of selected features reach 275 and 670 in BlogCatalog and Flickr, respectively.
- USFS consistently outperforms all baseline methods on both datasets with significant performance gain in most cases. The reason is that traditional unsupervised feature selection algorithms are based on the *i.i.d.* assumption which is invalid in linked social media data. USFS takes advantage of link information to guide the unsupervised streaming feature selection. It can also be observed that when feature information is scarce (for example 20%), link information could better complement feature information for feature selection. We also perform pair-wise wilcoxon signed-rank test [7] between USFS and other baseline methods on different proportions of streaming features and the test results show that USFS is significantly better (with both 0.01 and 0.05 significance level).
- For baseline methods, clustering performance gradually decreases when features are continuously generated. While for USFS, the clustering performance is relatively more stable when the proportion of streaming features varies from 20% to 100%. The number of selected features by USFS is also very stable, which varies from 236 to 275 in BlogCatalog and 562 to 670 in Flickr, respectively. It demonstrates the effectiveness of streaming feature selection, with a large amount of streaming features, we can only dynamically maintain a small set of relevant features without deteriorating the performance.

4.4 Efficiency Performance Comparison

To evaluate the efficiency of the proposed USFS algorithm, we compare the running time of different methods in Figure 3. As LapScore, SPEC, NDFS and LUFS are not designed for dealing with streaming features, we rerun the feature selection process at each time step. For both datasets, we set the cumulative running time threshold to be around 10^4 seconds since all methods except USFS take more than 50 hours. As can be observed, the proposed USFS algorithm is significantly faster than other baseline methods, the average processing time for each feature in BlogCatalog and Flickr is only 0.62 seconds and 1.37 seconds, respectively.

We also record the cumulative running time of USFS when the cumulative running time of other methods arrive the threshold (10^4 seconds), the results show that in BlogCatalog, USFS is $7\times$, $20\times$, $29\times$, $76\times$ faster than LapScore, LUFS, NDFS, SPEC, respectively; in Flickr, USFS is $5\times$, $11\times$, $20\times$, $75\times$ faster than LapScore, LUFS, NDFS, SPEC, respectively. The difference is becoming larger as the curve of USFS in Figure 3 is getting more smooth when streaming features continuously arrive.

4.5 Effects of Parameters

As discussed in Section 3, USFS has four important parameters: the number of social latent factors k, and parameters α, β, and γ in Eq. (6). To investigate the effects of these parameters, we vary one parameter each time and fix the other three to see how the parameter affects the feature selection performance in terms of clustering with different number of selected features. We only perform the parameter study on BlogCatalog dataset to save space since we have similar observations on the Flickr dataset.

First, we vary the number of social latent factors k from 5 to 10 while fix the other three parameters ($\alpha = 10$, $\beta = 0.1$, $\gamma = 0.1$). The clustering performance in terms of Accuracy and NMI is illustrated in Figure 4. The clustering performance is the best when the number of social latent factors is close to the number of clusters, which is 6 in BlogCatalog.

To assess the effect of parameter α which controls the model sparseness, we vary α as $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ while fix $k = 6$, $\beta = 0.1$, $\gamma = 0.1$, performance variance between α and number of selected features is presented in Figure 5. With the increase of α, the clustering performance rises rapidly and then keeps stable between the range of 10 to 1000. A high value of α indicates that it is not easy for new features to pass the gradient test in Eq. (13), thus the accepted features are more relevant and meaningful.

We study the effect of parameter β which makes the model more robust. Similar to the setting of α, β is also in the range of $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ and $k = 6$, $\alpha = 10$, $\gamma = 0.1$. The results are shown in Figure 6. We can see that clustering performance is much more sensitive to the number of selected features than to β. The performance is relatively higher when β is between 0.1 and 10.

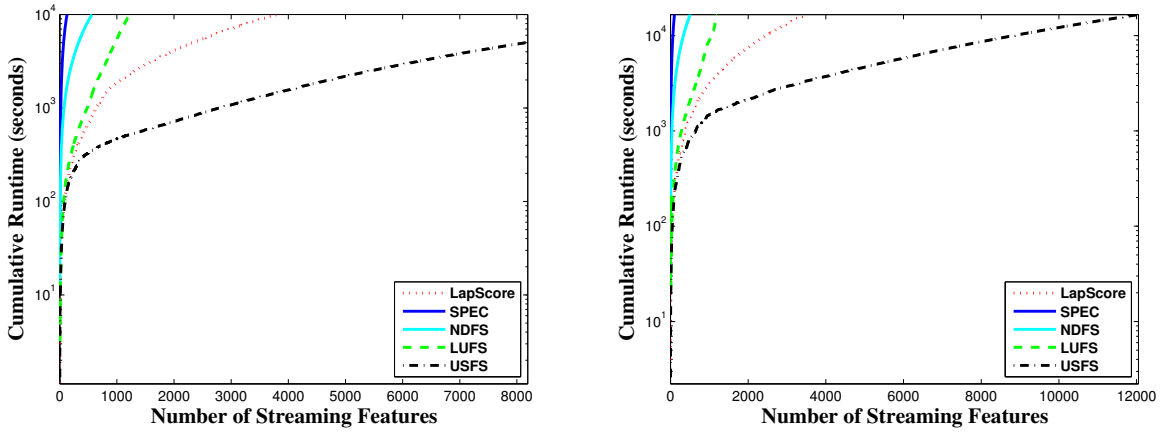
At last, we evaluate the trade-off between link information and feature information by varying γ in $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ while fix $k = 6$, $\alpha = 10$, $\beta = 0.1$. The results are presented in Figure 7. As shown in the figure, in most cases, the clustering performance first increases, reaches its peak and then it gradually decreases. The best performance achieves when γ is around 0.1. These observations suggest the importance of both link information and feature information in unsupervised streaming feature selection.

	Accuracy								
	20%(259)	30%(266)	40%(270)	50%(271)	60%(272)	70%(272)	80%(274)	90%(275)	100%(275)
LapScore	37.36	25.60	22.88	23.00	23.36	26.96	26.33	26.73	26.06
SPEC	30.10	29.50	24.79	21.48	18.88	19.63	18.34	18.30	18.01
NDFS	35.89	25.37	23.67	26.44	25.42	26.00	23.69	23.85	23.73
LUFS	24.65	24.11	22.27	22.84	22.50	20.71	21.61	20.71	20.48
USFS	**40.65**	**39.61**	**40.57**	**40.61**	**40.67**	**40.67**	**40.78**	**40.84**	**40.84**
	NMI								
	20%(259)	30%(266)	40%(270)	50%(271)	60%(272)	70%(272)	80%(274)	90%(275)	100%(275)
LapScore	0.1451	0.0600	0.0474	0.0510	0.0507	0.0743	0.0675	0.0793	0.0652
SPEC	0.0606	0.0765	0.0397	0.0143	0.0051	0.0098	0.0032	0.0029	0.0019
NDFS	0.1475	0.1250	0.1193	0.1092	0.1234	0.1006	0.1125	0.1130	0.1150
LUFS	0.0674	0.0533	0.0465	0.0490	0.0462	0.0492	0.0462	0.0345	0.0287
USFS	**0.2028**	**0.1861**	**0.2028**	**0.2026**	**0.2042**	**0.2042**	**0.2059**	**0.2072**	**0.2072**

Table 2: Clustering results with different feature selection algorithms on BlogCatalog dataset.

	Accuracy								
	20%(645)	30%(666)	40%(670)	50%(670)	60%(670)	70%(670)	80%(670)	90%(670)	100%(670)
LapScore	25.06	19.30	21.27	17.52	15.27	13.58	13.53	12.73	12.07
SPEC	25.52	20.26	17.50	15.46	13.53	14.11	13.94	13.52	13.07
NDFS	22.30	**29.50**	26.79	25.29	25.64	28.01	25.97	**29.08**	20.45
LUFS	27.13	22.11	19.19	24.00	24.79	19.97	18.22	19.24	23.99
USFS	**27.22**	**29.50**	**28.37**	**28.37**	**28.37**	**28.37**	**28.37**	28.37	**28.37**
	NMI								
	20%(645)	30%(666)	40%(670)	50%(670)	60%(670)	70%(670)	80%(670)	90%(670)	100%(670)
LapScore	0.1072	0.0629	0.0786	0.0521	0.0308	0.0143	0.0172	0.0100	0.0040
SPEC	0.0854	0.0546	0.0328	0.0245	0.0117	0.0152	0.0118	0.0109	0.0083
NDFS	0.0676	0.1260	0.1073	0.0876	0.0853	0.1207	0.1236	0.1152	0.0663
LUFS	0.1129	0.0958	0.0550	0.1015	0.1023	0.0602	0.0535	0.0524	0.0913
USFS	**0.1235**	**0.1368**	**0.1262**	**0.1262**	**0.1262**	**0.1262**	**0.1262**	**0.1262**	**0.1262**

Table 3: Clustering results with different feature selection algorithms on Flickr dataset.



(a) BlogCatalog (b) Flickr

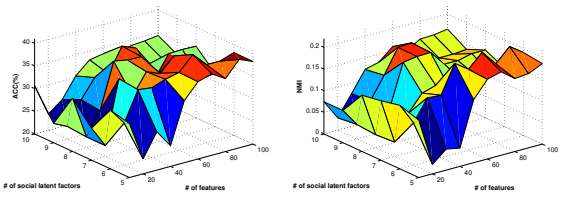
Figure 3: The cumulative runtime on BlogCatalog dataset and Flickr dataset. The Y axes are in log scale.

5. RELATED WORK

In this section, we first briefly review some related work in traditional feature selection, then we introduce existing work on supervised streaming feature selection.

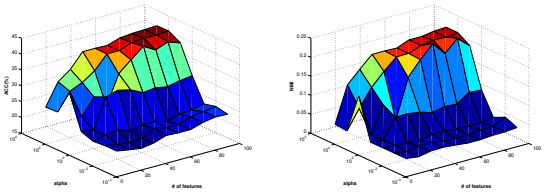
5.1 Traditional Feature Selection

Feature selection algorithms can be broadly grouped into three categories: filter methods, wrapper methods as well as embedded methods. Filter methods are independent of any learning algorithms and are thereby very efficient, they rely on some data characteristics such as distance, consistency, dependency, information, and correlation to measure the strength of each feature individually [13, 16, 25, 28]. Wrapper methods use the prediction power of a predefined learning algorithm to evaluate the quality of selected features. They are inevitably computational expensive since the search space grows exponentially with the number of features [9]. Embedded methods is a tradeoff between these two models which combines feature selection and model con-



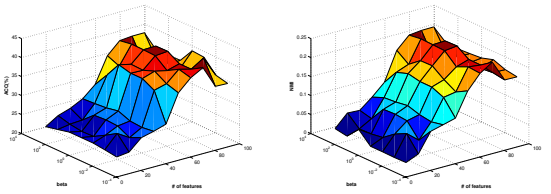
(a) Accuracy (b) NMI

Figure 4: Effect of # social latent factors.



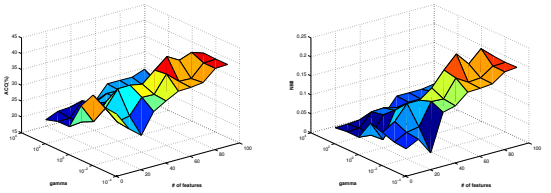
(a) Accuracy (b) NMI

Figure 5: Effect of α.



(a) Accuracy (b) NMI

Figure 6: Effect of β.



(a) Accuracy (b) NMI

Figure 7: Effect of γ.

struction [4, 15, 23]. Therefore, they are usually comparably efficient to filters and are comparably accurate to wrappers.

According to the availability of labels, feature selection methods consist of supervised methods and unsupervised methods. Supervised methods take advantage of the discriminative information encoded in class labels to select the subset of features that are able to distinguish instances from different classes [8, 23, 24]. Since real-world data is usually unlabeled and collecting labeled data is particular expensive requiring both time and effort, unsupervised feature selection receives more and more attention recently. Due to the lack of label information, unsupervised feature selection algorithms exploit different criteria to define the relevance of features such as data similarity [4, 16, 36] and local discriminative information [18, 35].

5.2 Streaming Feature Selection

Previous mentioned methods can only handle static dataset without considering the dynamic properties of features. Sometimes, it is difficult to obtain the whole feature space. Streaming feature selection, provides a solution to this problem. The first attempt to perform streaming feature selection is credited to Perkins and Theiler [27]. They adopted a stagewise gradient descent technique to dynamically update selected feature set. Alpha-investing calculates the p-value of new feature in a regression model to determine if it should be included. In Alpha-investing [37], once a feature is added, it will never be discarded in the future. OSFS [33] finds an optimal subset of features based on online feature relevance and feature redundancy analysis. At each time step, new candidate feature is accepted if it is strongly or weakly relevant to existing features; then existing but redundant features will be removed by redundancy analysis. In contrast to traditional streaming feature selection, Wang et al. [32] studied the problem of group streaming feature selection in which features continuously arrive in groups, the setting is more realistic in real-world applications. Guo et al. [14] proposed a node classification algorithm for streaming network. They consider the changes of network structure and node contents for node classification via feature selection techniques. All previous mentioned methods, focus on supervised scenarios, unsupervised streaming feature selection, however, is not well studied yet.

It should be noted that streaming feature selection differs from the task of feature selection/extraction on data streams, such as [1, 5, 6, 11, 34], in which feature sets are predefined while data are generated dynamically.

6. CONCLUSION AND FUTURE WORK

The prevalence of unlabeled, high-dimensional and high-velocity social media data presents new challenges for feature selection. Meanwhile, the unique characteristics of link information in social media bring about new opportunities as we might be able to leverage them to dynamically select relevant features efficiently. In this work, we study a novel problem, unsupervised streaming feature selection for social media data. In particular, we investigate how to take advantage of link information to perform unsupervised streaming feature selection. Also, we propose a stagewise algorithm to solve the optimization problem at each time step. Therefore, the model has the power to decide whether to add a newly arrived feature and whether to remove existing features promptly. Theoretical time complexity analysis and empirical experimental results on real-world social media datasets demonstrate that the proposed USFS framework works effectively and efficiently.

Future work can be focused on two aspects: First, in this work, we assume that the link information is relative stable compared with dynamic feature information. However, sometimes, the network structure is also continuously changing, how to perform streaming feature selection on dynamic network is a challenging problem and is unsolved yet. Second, in social media, we have features from multiple sources, like image features, text features, video features, etc. We will investigate how to fuse heterogeneous feature sources in the streaming feature selection task in the future.

Acknowledgments

This material is, in part, supported by National Science Foundation (NSF) under grant number IIS-1217466.

7. REFERENCES

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for projected clustering of high dimensional data streams. In *VLDB*, pages 852–863. VLDB Endowment, 2004.

[2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. In *NIPS*, pages 33–40, 2009.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] D. Cai, C. Zhang, and X. He. Unsupervised feature selection for multi-cluster data. In *KDD*, pages 333–342. ACM, 2010.

[5] X. Chen and K. S. Candan. Gi-nmf: Group incremental non-negative matrix factorization on data streams. In *CIKM*, pages 1119–1128. ACM, 2014.

[6] X. Chen and K. S. Candan. Lwi-svd: low-rank, windowed, incremental singular value decompositions on time-evolving data sets. In *KDD*, pages 987–996. ACM, 2014.

[7] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[8] C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.

[9] J. G. Dy and C. E. Brodley. Feature subset selection and order identification for unsupervised learning. In *ICML*, pages 247–254, 2000.

[10] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[11] L. Gong, J. Zeng, and S. Zhang. Text stream clustering algorithm based on adaptive feature selection. *Expert Systems with Applications*, 38(3):1393–1399, 2011.

[12] P. K. Gopalan, S. Gerrish, M. Freedman, D. M. Blei, and D. M. Mimno. Scalable inference of overlapping communities. In *NIPS*, pages 2249–2257, 2012.

[13] Q. Gu, Z. Li, and J. Han. Generalized fisher score for feature selection. In *UAI*, pages 266–273, 2012.

[14] T. Guo, X. Zhu, J. Pei, and C. Zhang. Snoc: streaming network node classification. In *ICDM*, pages 150–159. IEEE, 2014.

[15] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

[16] X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *NIPS*, pages 507–514, 2005.

[17] G. H. John, R. Kohavi, K. Pfleger, et al. Irrelevant features and the subset selection problem. In *ICML*, 1994.

[18] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu. Unsupervised feature selection using nonnegative spectral analysis. In *AAAI*, pages 1026–1032, 2012.

[19] H. Liu, E.-P. Lim, H. W. Lauw, M.-T. Le, A. Sun, J. Srivastava, and Y. Kim. Predicting trusts among users of online communities: an epinions case study. In *EC*, pages 310–319. ACM, 2008.

[20] H. Liu and H. Motoda. *Computational Methods of Feature Selection*. CRC Press, 2007.

[21] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, pages 415–444, 2001.

[22] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[23] F. Nie, H. Huang, X. Cai, and C. H. Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *NIPS*, pages 1813–1821, 2010.

[24] F. Nie, S. Xiang, Y. Jia, C. Zhang, and S. Yan. Trace ratio criterion for feature selection. In *AAAI*, pages 671–676, 2008.

[25] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[26] S. Perkins, K. Lacker, and J. Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *The Journal of Machine Learning Research*, 3:1333–1356, 2003.

[27] S. Perkins and J. Theiler. Online feature selection using grafting. In *ICML*, pages 592–599, 2003.

[28] M. Robnik-Šikonja and I. Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning*, 53(1-2):23–69, 2003.

[29] J. Tang and H. Liu. Unsupervised feature selection for linked social media data. In *KDD*, pages 904–912. ACM, 2012.

[30] L. Tang and H. Liu. Relational learning via latent social dimensions. In *KDD*, pages 817–826. ACM, 2009.

[31] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, pages 267–288, 1996.

[32] J. Wang, Z.-Q. Zhao, X. Hu, Y.-M. Cheung, M. Wang, and X. Wu. Online group feature selection. In *IJCAI*, pages 1757–1763. AAAI Press, 2013.

[33] X. Wu, K. Yu, H. Wang, and W. Ding. Online streaming feature selection. In *ICML*, pages 1159–1166, 2010.

[34] W. Yang, H. Xu, and A. Theorem. Streaming sparse principal component analysis. In *ICML*, pages 494–503, 2015.

[35] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou. l2, 1-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI*, pages 1589–1594, 2011.

[36] Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *ICML*, pages 1151–1157. ACM, 2007.

[37] J. Zhou, D. Foster, R. Stine, and L. Ungar. Streaming feature selection using alpha-investing. In *KDD*, pages 384–393. ACM, 2005.

[38] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67(2):301–320, 2005.