NED: Niche Detection in User Content Consumption Data

Ekta Gujral*
egujr001@ucr.edu
UC Riverside

Leonardo Neves
lneves@snap.com
Snap Inc.

Evangelos Papalexakis
epapalex@cs.ucr.edu
UC Riverside

Neil Shah
nshah@snap.com
Snap Inc

ABSTRACT

Explainable machine learning methods have attracted increased interest in recent years. In this work, we pose and study the *niche detection* problem, which imposes an explainable lens on the classical problem of co-clustering interactions across two modes. In the niche detection problem, our goal is to identify *niches*, or co-clusters with node-attribute oriented explanations. Niche detection is applicable to many social content consumption scenarios, where an end goal is to describe and distill high-level insights about user-content associations: not only that certain users like certain types of content, but rather the *types* of users and content, explained via node attributes. Some examples are an e-commerce platform with who-buys-what interactions and user and product attributes, or a mobile call platform with who-calls-whom interactions and user attributes. Discovering and characterizing niches has powerful implications for user behavior understanding, as well as marketing and targeted content production. Unlike prior works, ours focuses on the intersection of explainable methods and co-clustering. First, we formalize the niche detection problem and discuss preliminaries. Next, we design an end-to-end framework, NED, which operates in two steps: *discovering* co-clusters of user behaviors based on interaction densities, and *explaining* them using attributes of involved nodes. Finally, we show experimental results on several public datasets, as well as a large-scale industrial dataset from Snapchat, demonstrating that NED improves in both co-clustering (≈ 20% accuracy) and explanation-related objectives (≈ 12% average precision) compared to state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Feature selection**; **Non-negative matrix factorization**.

KEYWORDS

Co-clustering; Feature selection; Mutual Information; Explainability

ACM Reference Format:
Ekta Gujral, Leonardo Neves, Evangelos Papalexakis, and Neil Shah. 2021. NED: Niche Detection in User Content Consumption Data. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3459637.3482455

*This work was done when the author was on internship at Snap Inc.

This work is licensed under a Creative Commons Attribution International 4.0 License.

CIKM '21, November 1–5, 2021, Virtual Event, Australia.
© 2021 Copyright is held by the owner/author(s).
ACM ISBN 978-1-4503-8446-9/21/11.
https://doi.org/10.1145/3459637.3482455

1 INTRODUCTION

Developing features that attract and appeal to customers is critical to companies, as it determines their success and revenue [22]. Recently, platforms that create, serve, and curate content such as Snapchat, Netflix, and Youtube use data-driven approaches to attract and maintain a large and diverse userbase. This approach has successfully promoted mass engagement and user traction on popular content via recommendation systems: the larger the number of users that interact with a piece of content, the more that piece of content tends to be promoted. While this model is successful, engineering the process of content creation to generate this level of appeal by automating the process of identifying market *niches*, is quite challenging.

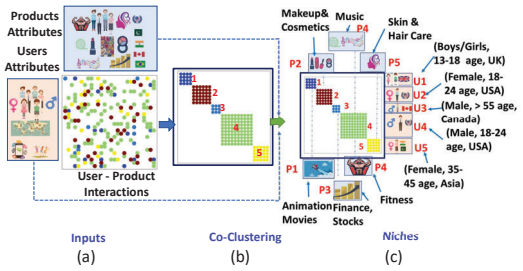


Figure 1: Our niche detection framework, NED: (a) takes as inputs user-content interactions, user attributes, and content attributes, (b) mines coherent co-clusters from interaction data, and (c) outputs niches, or co-clusters imbued with concise, user and content attribute-oriented explanations.

Specifically, in creating content, it is beneficial to understand the attributes that make a particular type of content (e.g food-related videos) attractive to specific markets that might be under served by the platform (e.g. 18-24 year old women in Australia). Thus, to improve targeting and to create and promote content that will likely better retain, engage, and satisfy target audiences, we propose a method for self-explaining *niche detection* in user content consumption data. Our work characterizes niches as outstanding co-clusters in user-content interaction graph data, imbued with user and content-oriented attributed co-cluster explanations which designate audience and content types. At its heart, our work addresses the scenario: *Consider that we have a rich user-liked-content interaction graph, and nodal attributes describing the users (e.g. user profile, demographics information, device types) and content (e.g. creator profile, category), how can we explain and make sense of it?*

Our work lies at the intersection of co-clustering and explainable machine learning, though the problem we aim to solve is one that other works do not. Co-clustering is a method that aims to simultaneously cluster both users and content (interchangeably, products) simultaneously so that users and content can be organized into homogeneous blocks. This approach is helpful to partially

answer our research question. In the literature, many co-clustering methods [3, 19, 26] have been proposed. For instance, [3, 12] take information theoretic approaches to derive co-clusters by maximizing preserved mutual information and entropy between users and items. More recently, [62, 63] utilize deep autoencoders to generate low-dimensional representations for users and products, and employ Gaussian Mixture Models to infer the co-clusters. Several works have also employed Non-Negative Matrix Tri-Factorization (NMTF) [21, 45, 52], turning the co-clustering problem into a matrix approximation problem [25]. Although these methods can be utilized to identify outstanding co-clusters, they are inherently limited in their capacity to *explain* the interactions. This is primarily because they cannot leverage user or item attributes to explain the discovered co-clusters, and thus have difficulty summarizing interactions between diverse user and item groups concisely. Explainability needs are paramount for the task we aim to solve in practice: ultimately, any discovered data-driven insights must be acted upon by creators to dictate a content creation strategy. Moreover, while explainability has become a focal point of recent works in machine learning [17, 29], there is no prior work on imbuing co-clusters with explanations, which is an important facet of the niche detection problem we propose in this work.

Summarily, addressing the niche detection problem has clearly high value for content strategists and platforms, but it poses several notable challenges. Firstly, it requires powerful and performant, large-scale co-clustering, to discover coherent and outstanding statistical regularities in user-content interaction data. Secondly, it requires a component to leverage nodal attributes available on users and content to concisely explain the associations between users and content which a co-cluster is characterized by, in a fashion interpretable to humans who must act as decision-makers on the basis of the explanations. The output of such a method may deliver informative insights from crude user-content interaction data, such as "female users generally enjoy skin-care related content more than others" or "18-24 male users enjoy bodybuilding content more than others," which can be acted upon and guide creators and strategists. More importantly than discovering popular or globally intuitive patterns, the methodology of a successful niche detection framework offers the promise of enabling the discovery of smaller, underserved and non-apparent niches which can inform highly localized and tailored content solutions for a subgroup. In this work, we formally pose the niche detection problem, and propose a framework, NED, to tackle it by carefully designing the two components: Figure 1 illustrates the high-level process. We further conduct extensive experiments demonstrating NED's success in discovering coherent niches, and outperformance of alternative approaches. Our contributions are:

- **Novel Problem Formulation**: We formally pose the niche detection problem for user behavior modeling. The problem is guided by two sub-problems: discovering co-clusters of user behaviors based on interaction densities, and explaining them using attributes of involved nodes via feature learning.
- **Novel Algorithm**: We propose an intuitive, simple, efficient, and effective method, NED, a niche detection framework, which proposes mutual information-inspired variant of NMTF for co-clustering, and a mutual information-inspired solution for explainable feature selection. To our knowledge, NED is the first approach at explainable co-clustering. Moreover, to evaluate niche quality, we propose a compression-based score to bridge the detected co-clusters and the feature explanations.
- **Extensive Experiments**: We evaluate NED on multiple synthetic and publicly available real-world datasets, as well as a private, large-scale sparse dataset with $500K$ users, 2500 products and $> 5M$ interactions from Snapchat, showing $\approx$ 14% accuracy improvement against state-of-the-art co-cluster approaches and $\approx$ 20% average precision improvement in explanation quality.

Our implementation is available at link[1].

2 PROBLEM FORMULATION

Table 1 contains the symbols used throughout the paper. Before we conceptualize the niche detection problem that our work tackles, we define certain terms necessary to set up the problem and formally define the problem statement.

Symbols	Definition
$\mathbf{X}$, $\mathbf{x}$, x	a matrix, column vector and scalar
$\mathbf{A}, \mathbf{B}, \mathbf{S}$	user cluster, product cluster and summary matrix
$\mathbf{U}_f, \mathbf{P}_f$	user attributes matrix, product attributes matrix
$\mathbf{U}_{P_f}, \mathbf{P}_{U_f}$	user to product attributes matrix, product to user attributes matrix
I, J, F_1, F_2	#users, #products, #user features, #product features,
M, N	No of user clusters, No of product clusters

Table 1: Table of symbols and their descriptions.

2.1 Problem Context

We assume input data that consists of:

- A set of users and products (interchangeably, contents) represented by $\mathcal{U} = \{U_1, \ldots, U_I\}$ and $\mathcal{P} = \{P_1, \ldots, P_J\}$ respectively. The relationship between entities consists of: user-product interactions containing tuple of the form (U_i, P_j, x_{ij}), where $U_i \in \mathcal{U}$ and $P_i \in \mathcal{P}$ and $x_{ij} \in \{0, 1\}$, and arranged in the form of binary matrix $\mathbf{X} \in \mathbb{R}^{I \times J}$, and zero entries indicate absent interactions.
- A set of user features, arranged in a binary matrix $\mathbf{U}_f \in \mathbb{R}^{I \times F_1}$, where F_1 represents total number of user features, and zero entries indicate absent features.
- A set of product (or content) features stored in a binary matrix $\mathbf{P}_f \in \mathbb{R}^{J \times F_2}$, where F_2 represents total number of product features, and zero entries indicate absent features.

The eventual goal in solving the niche detection problem is the capacity to discover co-clusters with user and product attribute-oriented explanations. The problem can be decomposed into two subgoals: first, identifying quality co-clusters, and second, explaining the co-clusters via careful node feature selection.

2.2 Problem Statement

Next, we introduce a few basic definitions necessary to define our novel niche detection problem.

Definition 1 (Co-cluster). *Formally, given a $I \times J$ data matrix $\mathbf{X}$, a co-clustering can be defined by two maps ρ and γ, which groups*

[1] http://www.cs.ucr.edu/~egujr001/ucr/madlab/src/NED.zip

users (or rows) and products (or columns) of $\mathbf{X}$ *into* M *and* N *disjoint or hard clusters respectively. Specifically,*

$$\rho : \{U_1, U_2, ..., U_I\} \rightarrow \{\hat{U}_1, \hat{U}_2, ..., \hat{U}_M\} \quad (1)$$

$$\gamma : \{P_1, P_2, ..., P_J\} \rightarrow \{\hat{P}_1, \hat{P}_2, ..., \hat{P}_N\} \quad (2)$$

where $\rho(U) = \hat{U}$ *denotes that user* U *is in user cluster* $\hat{U}$*, and* $\gamma(P) = \hat{P}$ *denotes that product* P *is in product cluster* $\hat{P}$*. A co-cluster is an interaction block defined by* $\{\hat{U}_m, \hat{P}_n\}$ *for some* $m \leq M, n \leq N$*.*

Definition 2 (Co-cluster Explanation). *A co-cluster explanation predicates the existence of binary feature matrices for users and products,* $\mathbf{U}_f$*, and* $\mathbf{P}_f$ *respectively. Let* $sub(\mathbf{U}_f)$ *and* $sub(\mathbf{P}_f)$ *respectively denote some subset of the* F_1 *user and* F_2 *product features, without loss of generality. An explanation for a co-cluster is indicated by a suitable pair* $(sub(\mathbf{U}_f), sub(\mathbf{P}_f))$*, which are associated with the co-cluster.*

Definition 3 (Niche). *A niche is the pairing of a co-cluster with a co-cluster explanation. Formally, a niche is indicated by*

$$\{\hat{U}_m, \hat{P}_n, sub(\mathbf{U}_f), sub(\mathbf{P}_f)\} \quad (3)$$

where $\hat{U}_m$ *is* m^{th} *user cluster,* $\hat{P}_n$ *is the* n^{th} *product cluster, and* $sub(\mathbf{U}_f)$ *and* $sub(\mathbf{P}_f)$ *are subsets of user and product features.*

We now have all the necessary definitions to formally define our problem. Hence, we pose the following:

Given (a) a set of users $\mathcal{U}$ and products $\mathcal{P}$ with user-product interactions $\mathbf{X}$, (b) user-feature relationships $\mathbf{U_f}$, and (c) product-feature relationships $\mathbf{P_f}$;
Design a framework to identify one or more coherent niches of the form in Equ. 3.

3 NED FOR NICHE DETECTION

Our proposed approach relies on two successive steps. First, the *co-clustering step* that co-clusters the user-product interaction data matrix and second, the *explaining step*, which focuses on learning the set of features that suitably characterize high-quality co-clusters discovered in the previous step. A two-step method is beneficial for two reasons: first, the *driving* features of the generative process may be missing in the observed data in lieu of strong correlates, in which case we may not want to try and infer a misleading process directly from these correlates; its sufficient in our case to try to identify strong correlative, instead of non-causal relationships. Second, with co-clustering independent of the features, we avoid *missed cluster* scenarios where a joint generative process may not identify a co-cluster simply because it is composed of a mix of features that the process is not sufficiently capable of capturing. By decoupling clustering and explanation, we prioritize recall on observed interaction clusters, acknowledging that some may be hard to adequately explain, but exist nonetheless. Although any co-clustering algorithm can be used in solving the niche detection problem, we propose a variant of NMTF guided by mutual information as one suitable solution. In fact, through extensive experiments (see Section 4.4), we show that our proposed method is not only intuitive, but achieves state-of-the-art co-clustering performance compared to previously proposed methods in literature.

3.1 Step 1: Co-Clustering

We frame the co-clustering problem (Defn. 1) in an NMTF-inspired formulation: we seek a decomposition of the user-product matrix $\mathbf{X} \in \mathbb{R}^{I \times J}$ into three low dimensional non-negative [36, 52] latent factor matrices i.e. $\mathbf{A} \in \mathbb{R}_+^{I \times M}$ is user-clustering matrix, $\mathbf{B} \in \mathbb{R}_+^{J \times N}$ is product-clustering matrix and $\mathbf{S} \in \mathbb{R}_+^{M \times N}$ provides the summary of $\mathbf{X}$ due to co-clustering. The values M and N represent the number of user and product clusters, respectively ($M << \min(I, J)$ and $N << \min(I, J)$). The optimization problem can be represented as:

$$\mathcal{L}(\mathbf{X}, \mathbf{A}, \mathbf{B}, \mathbf{S}) = \min_{\mathbf{A},\mathbf{B},\mathbf{S}} ||\mathbf{X} - \mathbf{ASB}^T||_2^F \quad s.t. \quad \mathbf{A} \geq 0, \quad \mathbf{B} \geq 0, \quad \mathbf{S} \geq 0 \quad (4)$$

In this way, users and products are clustered simultaneously while satisfying constraints, keeping a good low-rank approximation.

3.1.1 Factor Inference. Here, we derive an alternating optimization algorithm that infer the latent factor matrices from the user-product interaction $\mathbf{X}$. The Equ. 4 can re-written as:

$$\begin{aligned} \mathcal{L} &= \frac{1}{2} Tr((\mathbf{X} - \mathbf{ASB}^T)(\mathbf{X} - \mathbf{ASB}^T)^T) \\ &= \frac{1}{2} Tr(\mathbf{XX}^T - 2\mathbf{XBS}^T\mathbf{A}^T + \mathbf{ASB}^T\mathbf{BS}^T\mathbf{A}^T) \end{aligned} \quad (5)$$

Next, as in the bilateral k-means algorithm [24], we derive iterative update rules for $\mathbf{A}$ and $\mathbf{B}$ under the non-negativity constraints. The update rule of $\mathbf{A}$ is given as:

$$\mathbf{A}_{im} = \begin{cases} 1 & m = \text{argmax}_i(\hat{\mathbf{A}}(i,:)), \quad \hat{\mathbf{A}}(i,:) = [\mathbf{XBS}^T](i,:) \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

There is only one element equal to 1 at m^{th} column and the rest are zeros in each i^{th} row of $\mathbf{A}$. Similarly, $\mathbf{B}$ can be updated as:

$$\mathbf{B}_{jn} = \begin{cases} 1 & n = \text{argmax}_j(\hat{\mathbf{B}}(j,:)), \quad \hat{\mathbf{B}}(j,:) = [\mathbf{X}^T\mathbf{AS}](j,:) \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

As we impose constraints on user ($\mathbf{A}$) and product ($\mathbf{B}$) latent factors, the summary matrix $\mathbf{S}$ could be noisy since it is not optimized with any given criterion. It could represent an unclear structure (due to data noise, and high overlap among the categories represented by the clusters) where either there is no correlation between clusters, or every cluster is associated with other clusters. In order to mitigate the above issue, we compute the summary matrix $\mathbf{S}$ as positive point-wise mutual information (PPMI) matrix, which can extract clearer co-clusters and exploits its background knowledge for further convergence of the algorithm. The PMI is a theoretical measure of information widespread used to measure the association between pairs of results that arise from discrete random variables. In the literature [44], it is shown that this measure is highly correlated to conditional probability and resembles human judgment. Mathematically, the PMI between two random variable u and v is given by:

$$PMI(u, v) = \log\left(\frac{p(u, v)}{p(u)p(v)}\right) \quad (8)$$

Thus, we compute $\mathbf{S} \in \mathbb{R}_+^{M \times N}$ as follows:

$$\mathbf{S} = \log\left(\frac{\mathbf{A}^T\mathbf{XB}}{\sum_{j=1}^{J}\mathbf{A}^T\mathbf{X}\sum_{i=1}^{I}\mathbf{XB}}\right) \quad s.t. \quad \mathbf{S} \geq 0 \quad (9)$$

So, each element of $\mathbf{S}_{m,n}$ represents the PMI between a user cluster $\mathbf{A}_m$ and a product cluster $\mathbf{B}_n$. The rows of the matrix $\mathbf{A}^T\mathbf{X}$ are the basis vectors of the row space of $\mathbf{S}$ (i.e., the user centroids) and $\mathbf{B}$ the columns of the matrix $\mathbf{XB}$ are the basis vectors of the column space of $\mathbf{S}$ (i.e., the product centroids). Each positive value of the matrix has surely to be considered in the identification of co-cluster.

Why PMI?: The intuition behind computing $\mathbf{S}$ using PMI comes from the observation that sometimes, in the co-clustering of user-product data, a user cluster is associated with another product cluster that does not exist. This leads to the idea of an update rule based on co-occurrence between users and products for all cluster pairs in the given data. PMI is an information-theoretic approach that measures how often two clusters $(\mathbf{A}_m, \mathbf{B}_n)$ occur as compared with what we expect if they were independent. The numerator of PMI informs us how often we observed the two clusters together in user-product context consumption. The denominator informs us how often we would anticipate both to co-cluster, assuming they are independent clusters. Thus, the ratio provides us an estimation of how much more the two clusters co-cluster than we anticipate by chance. Our choice for PMI is encouraged by [58]. Using PMI, we encourage users of similar interactions with products to have closer representation in latent space. For example, when representing co-clusters, we can easily think of positive relationship (e.g. "Female" and "Nurse") but find it much harder to relate negative ones ("Female" and "Carpenter"). Therefore, we focus on positive point-wise mutual information. Next, Equ. (6) and (7) can be re-written as:

$$\hat{\mathbf{A}} = \mathbf{XB}\log\left(\frac{\mathbf{A}^T\mathbf{XB}}{\sum_{i=1}^{I}\mathbf{A}^T\mathbf{X}\sum_{i=1}^{I}\mathbf{XB}}\right)^T; \hat{\mathbf{B}} = \mathbf{X}^T\mathbf{A}\log\left(\frac{\mathbf{A}^T\mathbf{XB}}{\sum_{j=1}^{J}\mathbf{A}^T\mathbf{X}\sum_{i=1}^{I}\mathbf{XB}}\right) \tag{10}$$

Now, we assign each user/product to a single cluster by finding the cluster with maximum membership. This translates to finding the maximum column index for each row.

3.2 Step 2: Co-Cluster Explanation

After discovering outstanding co-clusters, we next aim to identify the user and product feature subsets that best explain the co-clusters. The process involves two steps: auxiliary feature matrix creation to infer the user preferences over product features to get insights about implicit user similarities, and feature selection using point wise mutual information to compute the association between features and user/product cluster centroids. The quest for similarities plays an important role in co-cluster analysis [59]. In quadripartite graphs (see Figure 2), one can derive several semantics on similarity by considering different paths in a graph. Upon deriving these, we have all information to learn important co-cluster features: the final suitability score for each feature is captured as a linear combination of both steps. We briefly explain each step as follow:

3.2.1 Auxiliary Feature Matrix. We have the user and product feature vectors ($\mathbf{U}_f \in \mathbb{R}^{I\times F_1}$ and $\mathbf{P}_f \in \mathbb{R}^{J\times F_2}$) which are independent and do not infer any user preferences for the features that appear in products and vice-versa. This could lead to undermining the user/product similarities for feature learning. To overcome this issue, we compute the user's proximity to a product feature through a meta-path as an indication of the user's possible "preference" towards the product feature and vice versa. The meta-path [59]

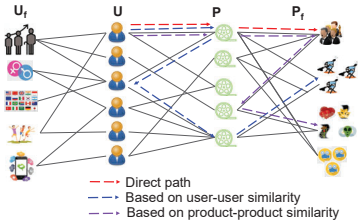


Figure 2: Quadripartite graph of the users $\mathbf{U} \in \mathbb{R}^I$, Product $\mathbf{P} \in \mathbb{R}^J$ and the user features $\mathbf{U}_f \in \mathbb{R}^{I\times F_1}$ and product features $\mathbf{P}_f \in \mathbb{R}^{J\times F_2}$. The resultant user auxiliary feature matrices is $\mathbf{U}_{p_f} \in \mathbb{R}^{I\times F_2}$.

is a powerful mechanism for a user to select an appropriate similarity semantics to learn from a set of examples of similar objects. Formally, a meta-path can be defined as:

Definition 4 (Meta-path). *A meta-path is a sequence of relations $\mathcal{R}$ between object types O, which defines a new composite relation between its starting type and ending type. It is denoted in the form of $O_1 \xrightarrow{\mathcal{R}_1} O_2 \xrightarrow{\mathcal{R}_2} \ldots \xrightarrow{\mathcal{R}_{m-1}} O_m$, which defines a composite relation between $\mathcal{R}_1 \circ \mathcal{R}_2 \circ \cdots \circ \mathcal{R}_m$ between types O_1 and O_m, where $\circ$ denotes the composition operator on relation. For example, a meta-path (Figure 2, purple path) user (male) $\rightarrow$ product (movie) $\rightarrow$ feature (action) $\rightarrow$ product (movie) $\rightarrow$ feature (comedy) (denoted as UPFPF) indicates user preferences based on content similarities.*

The process of auxiliary feature matrix creation is adapted from [31, 59]. In [31], movie preferences are learned by leveraging user similarities defined through different types of meta paths or relations to successfully design a new movie. Here, the auxiliary feature matrix helps to leverage both explicit features and user/product similarities via a graph-theoretic approach. For user auxiliary feature matrix as shown in Figure 2, the red path $\mathbf{U}_{p_f}^{UPF}$ (i.e., starting from a user and ending on a product feature via a product) finds the preferences for the product features for each user based on its interactions. The blue path $\mathbf{U}_{p_f}^{UPUPF}$ finds user preferences for the product features based on user-user similarity, and the purple path $\mathbf{U}_{p_f}^{UPFPF}$ finds user preferences based on product-product (content) similarity.

For the final weighted matrix $\mathbf{U}_{p_f} \in \mathbb{R}^{I\times F_2}$, which represents user $\mathbf{U}$ preference over product features $\mathbf{P}_f$ is a linear combination of the weighted $\mathbf{U}_{p_f}$ over the three predefined meta-path types:

$$\mathbf{U}_{p_f} = \alpha\mathbf{U}_{p_f}^{UPF} + \beta\mathbf{U}_{p_f}^{UPUPF} + \gamma\mathbf{U}_{p_f}^{UPFPF} \tag{11}$$

This linear combination helps to smooth the information in case of sparse direct user-product preferences. Similarly, we compute weighted auxiliary matrix for product features as:

$$\mathbf{P}_{u_f} = \alpha\mathbf{P}_{u_f}^{PUF} + \beta\mathbf{P}_{u_f}^{PUPUF} + \gamma\mathbf{P}_{u_f}^{PUFUF} \tag{12}$$

where $\alpha, \beta, \gamma \in \mathbb{R}^+$ are combination parameters satisfying criterion $\alpha+\beta+\gamma = 1$. We set $\alpha = 0.5$, $\beta = 0.25$ and $\gamma = 0.25$ in both cases. We give higher importance to direct user/product preferences (α) and lower importance to 4-step path (β, γ) because it could 'obscure' the individual preferences by depending on 'similar' users/products. Section 4.6 shows sensitivity analysis for these parameters.

3.2.2 Feature Selection. For feature selection, we propose a PMI-based approach that leverages the association between user and product cluster centroids. We compute information about how often we observed the certain features for each user cluster using $\mathbf{A}$ and $\mathbf{U}_f$ as $\frac{(\mathbf{A}^T\mathbf{U}_f)_{mf}}{\sum_{m=1}^{M}\sum_{\hat{f}=1}^{F_1}(\mathbf{A}^T\mathbf{U}_f)_{m\hat{f}}} \in \mathbb{R}^{M\times F_1}$. Next, we compute information about their association while independently drawn as $p(\mathbf{U}_f) = \frac{\sum_{\hat{f}=1}^{F_1} U_f}{\sum_{m=1}^{M}\sum_{\hat{f}=1}^{F_1}(\mathbf{A}^T\mathbf{U}_f)_{m\hat{f}}} \in \mathbb{R}^{1\times F_1}$ and $p(\mathbf{A}) = \frac{\sum_{m=1}^{M} A}{\sum_{m=1}^{M}\sum_{\hat{f}=1}^{F_1}(\mathbf{A}^T\mathbf{U}_f)_{m\hat{f}}} \in \mathbb{R}^{1\times M}$. Finally, we compute PMI relation as:

$$\mathbf{U}^1_{PMI} = \log\frac{p(\mathbf{A}^T\mathbf{U}_f)}{p(\mathbf{A})^T p(\mathbf{U}_f)} = \frac{\mathbf{A}^T\mathbf{U}_f * \sum_{m=1}^{M}\sum_{\hat{f}=1}^{F_1}(\mathbf{A}^T\mathbf{U}_f)_{m\hat{f}}}{\sum_{m=1}^{M}\mathbf{A}^T * \sum_{\hat{f}=1}^{F_1}\mathbf{U}_f} \tag{13}$$

where $\mathbf{U}^1_{PMI} \in \mathbb{R}^{M\times F_1}$. Now, we compute PMI relation for user auxiliary features matrix $\mathbf{U}_{p_f}$ as:

$$\mathbf{U}^2_{PMI} = \log\frac{p(\mathbf{A}^T\mathbf{U}_{p_f})}{p(\mathbf{A})^T p(\mathbf{U}_{p_f})} \in \mathbb{R}^{M\times F_2} \tag{14}$$

Similarly, we compute both PMIs ($\mathbf{P}^1_{PMI} \in \mathbb{R}^{N\times F_2}$ and $\mathbf{P}^2_{PMI} \in \mathbb{R}^{N\times F_1}$) for product features also. Due to space limitations and symmetry, we do not include derivations for them.

To select the most relevant user and product features for the niche, we linearly combine the PMIs associated with co-clustering (i.e. via summary matrix $\mathbf{S}$), attribute matrices (via. Equ. 13), and auxiliary matrices (via. Equ. 14) as following:

$$\textbf{For users:}\quad \mathbf{e}^u = \mathbf{U}^1_{PMI} + \mathbf{S} * \mathbf{P}^2_{PMI} \in \mathbb{R}^{M\times F_1} \tag{15}$$

$$\textbf{For products:}\quad \mathbf{e}^p = \mathbf{P}^1_{PMI} + \mathbf{S}^T * \mathbf{U}^2_{PMI} \in \mathbb{R}^{N\times F_2} \tag{16}$$

Finally, we choose the top N highest values for each cluster (or each row) from $\mathbf{e}^u$ (Equ. 15) and $\mathbf{e}^p$ (Equ. 16) to explain the niche.

4 EMPIRICAL EVALUATION

In this section, we aim to answer the following research questions:

- **RQ1 Accuracy**: Can NED outperform state-of-the-art alternatives at effectively capturing co-clusters?
- **RQ2 Explainability**: Can NED help learn meaningful explanations of co-clusters, and thus better niches?
- **RQ3 Scalability** How efficient and scalable is NED with respect to the size of the input graphs?

We discuss these after detailing our experimental setup.

4.1 Datasets and Experiment Setup

The details of the data used for experiments are given in Table 2.

4.1.1 Synthetic Data. In order to fully control and evaluate the niches in our experiments, we generate synthetic data i.e. user-product engagement graph $\mathbf{X} \in \mathbb{R}^{I\times J}$, user attributes $\mathbf{U}_f \in \mathbb{R}^{I\times F_1}$ (e.g. age, gender, country, app engagement etc.) and product attributes $\mathbf{P}_f \in \mathbb{R}^{J\times F_2}$ (e.g. name, publisher name, country, category,subcategory etc.) as discussed in Supplementary Material.

4.1.2 Real Data. We employ several real-world public datasets from different domains: *Cora* is publications dataset, *WebKB4* consists of classified web page information, *MovieLens* has data has 6000 users and 4000 movie rating information, *News20* is a collection of approximately 20,000 newsgroup documents, *Caltech* [32] is an image dataset. We also use private *Snapchat* dataset containing full-duration views between users and public feed contents from the Snapchat platform; we assume full views to indicate persisted interest in the consumed content, as in [27, 34].

Dataset	#users	#products	#features (users, products)	#clusters (users, products)
Syn-I	$10K$	$1K$	(22, 43)	(14, 20)
Syn-II	$50K$	$5K$	(22, 55)	(70, 35)
Syn-III	$500K$	$10K$	(22, 63)	(140, 50)
Syn-IV	$1M$	$50K$	(22, 86)	(280, 70)
Cora	$3K$	$1.5K$	–	(7, –)
WebKB4	$4K$	$1K$	–	(4, –)
MovieLens	$6K$	$4K$	(25, 23)	(–, 20)
News20	$19K$	$61K$	–	(20, –)
Caltech	$2K$	300	–	(3, –)
Snapchat	$500K$	$2.5K$	(22, 238)	(–, –)

Table 2: Details for the datasets. "-" indicates unknown/unavailable public information.

4.2 Baseline Methods

The two major components of NED are in (a) the discovering of coherent co-clusters, and (b) their concise explanation via nodal attributes. Thus, we conduct experiments with two categories of baseline to evaluate each contribution separately.

4.2.1 Co-clustering. : To demonstrate the superior performance of the proposed algorithm for data representation, we compare NED with closely related methods, which are listed as follows:

- **NEO-CC** [61]: A non-exhaustive overlapping co-clustering method based on the minimum sum-squared residue objective.
- **DeepCC** [63]: A deep autoencoder method which employs a variant of GMM to infer cluster assignments.
- **CoClusInfo** [50]: An information-theoretic approach which uses mutual information to define its objective function.
- **FNMTF** [10]: A fast, NMTF method based on projected gradients, coordinate descent, and alternating least squares optimization.
- **WC-NMTF** [52]: A word co-occurrence NMTF method that uses mutual information for co-clustering the word and documents.
- **SNCC** [37]: a sparse neighbor constrained co-clustering method for learning category consistency.

Note that in [52], **WC-NMTF** method performed better than original NMF [64], orthogonal NMF [65], projective NMF [66], graph regularized NMF [4], NMTF [36], orthogonal NMTF [13] and graph regularized NMTF [54]. Similarly, in [37], paper evaluated performance of **SNCC** against K-means [35], NMF [64], SNMF [14], graph regularized NMF [4], DNMTF [54], and DLLC [60]. Therefore, to avoid the repetitive comparison, we chose to compare our proposed method's performance with **SNCC** and **WC-NMTF**.

4.2.2 Explainability. Although we could not find any explicit co-clustering explainability baselines, we adapted the recently proposed **LightGBM** [29] for our explainability baseline. It is a boosting decision tree-based method that employs feature bundling to deal with a large number of features. To use it as our baseline, we fed co-clustering outcomes from the above-discussed method as labels for the user and product clusters along with user/product features data matrices to discern feature importance per co-cluster. We also compared our method with recently proposed method **BMGUFS** [2] which is rigorous approximation algorithms for block model

guided unsupervised feature selection and helps in finding high-quality features for cluster explanation.

4.3 Evaluation Measures

4.3.1 Co-clustering. We evaluate NED and the baselines for co-clustering using three criteria: ***Normalized Mutual Information (NMI)*** [5], ***Accuracy*** [5] and ***CPU time (sec)***.

4.3.2 Explainability. We evaluate NED and the baselines for explainability using three criteria namely **Average Precision**, **Stability score** [46] and our proposed **Compression Score**. The MDL based compression score is given by:

$$\begin{aligned} Score = {} & \log^* r_m + \log^* c_n + \log^* uf_m + \log^* pf_n + E(B_{mn}) \\ & - c_n \log_2 \frac{c_n}{J} + \log_2(r_m c_n + 1) - uf_m \log_2 \frac{uf_m}{F_1} + E(B_{mm}) \\ & + \log_2(r_m uf_m + 1) - pf_n \log_2 \frac{pf_n}{F_2} + \log_2(c_n pf_n + 1) + E(B_{nn}) \end{aligned} \tag{17}$$

where $\log^*$ is the universal code length for integers [49], r_m is m^{th} the row cluster encoding bits, c_n is n^{th} the column cluster encoding bits, uf_m is the m^{th} feature cluster encoding bits corresponding to the m^{th} row cluster, pf_n is n^{th} feature cluster encoding bits corresponding to the n^{th} column cluster. $E(B_{mn})$, $E(B_{mm})$ and $E(B_{nn})$ are the number of bits required to encode block of user-product, user-feature and product-feature, respectively. The lower the value, the better is the compression.

4.4 Quantitative Analysis

In this section, we provide quantitative evaluation and analysis of our method for co-clustering and explainability on synthetic and real-world datasets. Recommendation is undoubtedly one key task for businesses. However, explainable and thematic engagement is useful for both individual content creators (influencers, etc.) and companies (Snapchat, Netflix, etc.) who aim to attract certain audiences with original content. The most concrete and evaluateable components are discovering and describing the co-clusters and explanations found, upon which these creators can leverage findings. Thus, our quantitative evaluation is focused around these points, and we also show associated qualitative end-to-end findings on real data.

4.4.1 Co-clustering Performance. To answer the first experimental question, we report co-clustering performance of all methods in Table 3 and 4. We cap the allowed run-time of all methods to 24 hours, indicating unfinished/non-converged results as "-".

Synthetic Data: As we can see in Table 3, NED achieves the best performance for all synthetic datasets. As expected, NED significantly out-performs three related NMTF-based methods, SNCC, FNMTF and WC-NMTF, due to its use of mutual information via the summary matrix as described in Section 3. Moreover, NED outperforms CoClusInfo, NEO-CC and the neural method DeepCC, in both accuracy and NMI, surpassing them with substantial accuracy improvements (average $\approx$ 14%). We visualize the co-clustering ability of NED and second highest performing baseline (CoClusInfo) on an additional small synthetic data (SmallSYN). The SmallSYN data has 500 users, 500 features and 7 co-clusters, and is shown in Figure 3(a). The shuffled data fed into CoClusInfo and NED is shown in Figure 3(b). Subsequently, the users and products are rearranged to show discovered co-clusters as shown in Figure 3(d), according to which NED detects the co-clusters precisely (100% accuracy). CoClusInfo's results (achieving 91% accuracy) are shown in Figure 3(c). Note that the inferred co-cluster sequences in Figure 3(c-d) are not the same as in Figure 3(a) – this is because the cluster assignment for the same user/product clusters may be arbitrarily permuted during inference.

Real Data: Table 4 presents NED 's performance compared to state-of-the-art techniques on real datasets, using the NMI, accuracy and CPU Time (in seconds). For each dataset, we list the scores that correspond to a specific cluster type. For all real datasets except Movielens data, only user labels are available. For Movielens data, product labels are inferred from movie genre types. NED is the only method that shows consistently high performance on all different kinds of datasets, indicating its flexibility. Due to space limitations, here, we provide analysis of Movielens data below and analysis of other real datasets are provided in Supplementary Material.

Movielens Data: We observe that NED is able to detect 15 clusters for movies. This is reasonable outcome for this data, as there are overlapping movie categories; for example, "Toy Story (1995)" can be categorized in Animation as well as Comedy. FNMTF detected only 5 – 6 clusters as shown in Figure 4. CoClusInfo, WC-NMTF, SNCC and DeepCC achieved better performance, but still lower than NED.

We also visualize the co-clustering results on the Movielens dataset, to show a visual representation of improved co-clustering performance. We rearrange the original data matrix (Figure 4(a)) according to the user and product cluster assignments to show the co-clustering result. We observe that the co-clusters for NED (Figure 4(b)) are more salient compared to those for baselines FNMTF, WC-NMTF and DeepCC (Figures 4(c-e)). Snapchat dataset is very interesting and detailed analysis is provided in Section 4.7.

4.4.2 Explainability Evaluation. To answer the second experimental question, we first analyze the explanations derived from NED, and compare the results with explanations from baseline methods. Table 5 presents NED's performance over LightGBM explanations on 2 synthetic and 2 real datasets, using the stability score, average precision, CPU time in sec. and our proposed compression score.

Setup: First, we obtain co-clusters using NED and other very closely related two matrix tri-factorization based methods, i.e FNMTF and WC-NMTF. Note that baselines CoclusInfo, DeepCC and NEO-CC provide only user and product cluster factor matrices but do not provide any summary matrix **S**, which is required for our computations (See Equ. 15 and 16). Next, co-clustering results are fed into explainability components for both LightGBM, BMGUFS and NED. For NED, first, we create auxiliary matrices for both users and products (see Section 3.2.1), and then compute PMI using co-clustering outcomes (see Section 3.2.2). We then select the top-5 user and product features per co-cluster to explain it.

Results: We observe that NED drastically outperforms the baselines for all the datasets. For synthetic data SYN-I, each user cluster is created with combination of age and gender. Similarly, product clusters are focused on gender dominated viewership. For example, in our simulated data, women associate highly with "makeup & cosmetic, weight loss, and pop music" and men associate highly "card games, driving and racing games, and body building." Our results suggest that LightGBM and BMGUFS are not able to explain

Dataset	Cluster	Metric	NEO-CC	DeepCC	CoClusInfo	FNMTF	WC-NMTF	SNCC	NED
I	User	NMI	0.672 ± 0.052	0.771 ± 0.031	0.879 ± 0.021	0.601 ± 0.023	0.824 ± 0.022	0.831 ± 0.016	**0.948 ± 0.017**
		Accuracy	0.532 ± 0.029	0.597 ± 0.034	0.781 ± 0.049	0.703 ± 0.021	0.695 ± 0.019	0.780 ± 0.021	**0.865 ± 0.045**
	Product	NMI	0.772 ± 0.075	0.736 ± 0.002	0.886 ± 0.001	0.851 ± 0.043	0.801 ± 0.032	0.854 ± 0.161	**0.949 ± 0.096**
		Accuracy	0.776 ± 0.105	0.651 ± 0.005	0.789 ± 0.057	0.719 ± 0.001	0.683 ± 0.045	0.741 ± 0.122	**0.865 ± 0.032**
II	User	NMI	0.793 ± 0.062	0.769 ± 0.047	0.901 ± 0.008	0.883 ± 0.041	0.746 ± 0.052	0.902 ± 0.021	**0.960 ± 0.011**
		Accuracy	0.682 ± 0.062	0.568 ± 0.075	0.794 ± 0.052	0.703 ± 0.028	0.683 ± 0.119	0.761 ± 0.061	**0.867 ± 0.038**
	Product	NMI	0.673 ± 0.024	0.743 ± 0.064	0.924 ± 0.064	0.904 ± 0.058	0.839 ± 0.072	0.911 ± 0.141	**0.947 ± 0.014**
		Accuracy	0.540 ± 0.086	0.694 ± 0.109	0.799 ± 0.034	0.834 ± 0.058	0.788 ± 0.001	0.801 ± 0.014	**0.853 ± 0.042**
III	User	NMI	[-]	[-]	0.720 ± 0.011	0.913 ± 0.046	0.763 ± 0.023	0.921 ± 0.046	**0.973 ± 0.006**
		Accuracy			0.694 ± 0.017	0.523 ± 0.034	0.492 ± 0.002	0.632 ± 0.021	**0.882 ± 0.023**
	Product	NMI			0.840 ± 0.001	0.842 ± 0.074	0.763 ± 0.028	0.856 ± 0.121	**0.933 ± 0.019**
		Accuracy			0.640 ± 0.068	0.653 ± 0.104	0.535 ± 0.112	0.716 ± 0.214	**0.799 ± 0.023**
IV	User	NMI	[-]	[-]	0.879 ± 0.021	0.879 ± 0.021	[-]	0.881 ± 0.012	**0.959 ± 0.017**
		Accuracy			0.534 ± 0.011	0.495 ± 0.133		0.563 ± 0.112	**0.587 ± 0.072**
	Product	NMI			0.947 ± 0.001	0.893 ± 0.129		0.891 ± 0.012	**0.971 ± 0.007**
		Accuracy			0.832 ± 0.064	0.793 ± 0.101		0.801 ± 0.102	**0.902 ± 0.032**

Table 3: Experimental results for NMI and Accuracy for synthetic data. Boldface indicates the best results.

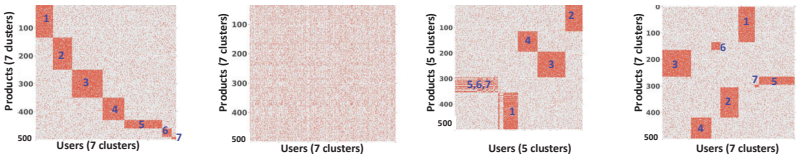


Figure 3: The co-clustering result of NED on synthetic data. Left to right: (a) original synthetic data with 7 users and 7 product clusters, (b) shuffled synthetic data, (c) the second-best performing baseline (CoClusInfo) result (91% accurate), and (d) NED's result (100% accurate).

Dataset	Cluster	Metric	NEO-CC	DeepCC	CoClusInfo	FNMTF	WC-NMTF	SNCC	NED
Cora	User	NMI	0.034 ± 0.004	0.003 ± 0.001	0.152 ± 0.006	0.152 ± 0.011	0.173 ± 0.002	0.112 ± 0.001	**0.181 ± 0.023**
		Accuracy	0.206 ± 0.034	0.2943 ± 0.045	0.375 ± 0.032	0.373 ± 0.101	0.314 ± 0.133	0.213 ± 0.022	**0.399 ± 0.034**
		Time (sec)	1488.8 ± 126.6	1481.4 ± 256.3	6.6 ± 1.23	4.5 ± 1.2	121.3 ± 12.5	**4.1 ± 0.34**	7.9 ± 0.7
WebKB4	User	NMI	0.136 ± 0.001	0.379 ± 0.084	0.383 ± 0.095	0.255 ± 0.054	0.241 ± 0.058	0.149 ± 0.003	**0.489 ± 0.043**
		Accuracy	0.374 ± 0.075	0.568 ± 0.026	0.653 ± 0.082	0.549 ± 0.102	0.503 ± 0.121	0.461 ± 0.101	**0.752 ± 0.029**
		Time (sec)	4089.7 ± 118.8	1719.9 ± 493.5	4.7 ± 1.6	3.7 ± 0.63	286.3 ± 2.2	**3.2 ± 0.21**	3.8 ± 0.6
MovieLens	Product	NMI	0.356 ± 0.020	0.636 ± 0.102	0.742 ± 0.112	0.394 ± 0.021	0.689 ± 0.039	0.472 ± 0.102	**0.783 ± 0.124**
		Accuracy	0.413 ± 0.020	0.550 ± 0.192	0.649 ± 0.102	0.382 ± 0.048	0.592 ± 0.124	0.564 ± 0.116	**0.683 ± 0.017**
		Time (sec)	8402.3 ± 363.5	2067.78 ± 34.5	103.98 ± 3.6	34.78 ± 6.9	673.67 ± 5.7	**29.43 ± 6.4**	41.55 ± 3.6
News20	User	NMI	[-]	0.448 ± 0.018	0.499 ± 0.091	0.149 ± 0.031	0.392 ± 0.019	0.334 ± 0.016	**0.541 ± 0.019**
		Accuracy		0.452 ± 0.075	0.433 ± 0.012	0.108 ± 0.093	0.329 ± 0.121	0.316 ± 0.021	**0.477 ± 0.029**
		Time (sec)		4028.3 ± 42.6	114.4 ± 4.1	97.4 ± 12.3	1143.1 ± 41.6	**92.6 ± 10.4**	111.1 ± 16.3
Caltech	User	NMI	0.324 ± 0.086	0.636 ± 0.087	0.753 ± 0.059	0.211 ± 0.101	0.593 ± 0.109	0.462 ± 0.011	**0.756 ± 0.019**
		Accuracy	0.543 ± 0.001	0.778 ± 0.022	0.899 ± 0.012	0.531 ± 0.111	0.835 ± 0.291	0.791 ± 0.091	**0.911 ± 0.102**
		Time (sec)	2649.6 ± 132.4	281.4 ± 13.3	2.4 ± 0.72	3.5 ± 0.41	103.5 ± 17.4	**2.9 ± 0.21**	3.9 ± 0.24

Table 4: Experimental results for NMI, Accuracy and CPU Time in seconds for real data. The boldface means the best results.

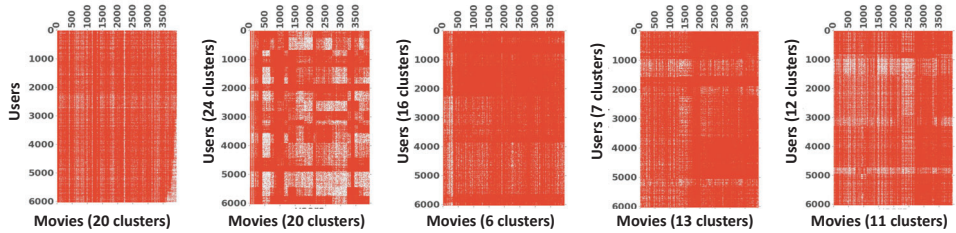


Figure 4: Visualized co-clustering result of NED on the Movielens data. From left to right: (a) Original data, (b) co-cluster detected by NED, (c) FNMTF, (d) WC-NMTF, and (e) DeepCC. NED evidently produces the most coherent co-clustering structure.

co-clusters well with the correct attributes. This is likely, partially because it trains a classifier using only co-cluster/cluster outcomes and does not leverage implicit similarities between users and products. Conversely, NED successfully captures these similarities in auxiliary matrices and is thus able to explain these relationships for each co-cluster. Similarly, NED is able to provide more complex explanations for SYN-II in which product clusters do not have any specific gender based dominance.

For Movielens, NED discovered an interesting co-cluster in which salesmen and programmers strongly associated with adventure movies, and in another co-cluster lawyers strongly associated with drama and fantasy movies. Interestingly, all co-clustering baselines with combination of BMGUFS for explanation underperformed in our experiments, compared to the solution adopted by LightGBM and NED. NED consistently compressed the discovered co-clusters quite well, compared to LightGBM, suggesting that we can describe the co-clusters more concisely with better-quality attribute/feature explanations. Overall, NED outperforms state-of-the-art approaches by at least ≈ 15% stability and ≈ 20% average precision improvement. Also, NED achieves at least 5% compression bits and 20% runtime reduction (see Table 5).

4.5 Scalability

Finally, to answer the third question, we experimentally study the runtime of NED with respect to the input size on real graph. For runtime measurements, we use a large private viewer-publisher

Data	Cluster	Score	FNMTF			WC-NMTF			NED		
			LightGBM	BMGUFS	NED	LightGBM	BMGUFS	NED	LightGBM	BMGUFS	NED
S-I	U	a	0.306 ± 0.04	0.294 ± 0.01	0.445 ± 0.06	0.381 ± 0.06	0.264 ± 0.02	0.447 ± 0.07	0.501 ± 0.06	0.416 ± 0.08	**0.596 ± 0.06**
		b	0.119 ± 0.08	0.115 ± 0.06	0.134 ± 0.01	0.112 ± 0.03	0.201 ± 0.09	0.289 ± 0.02	0.148 ± 0.01	0.296 ± 0.16	**0.496±0.07**
	P	a	0.288 ± 0.03	0.276 ± 0.04	0.301 ± 0.02	0.129 ± 0.04	0.131 ± 0.02	0.132 ± 0.01	0.116 ± 0.09	0.264 ± 0.12	**0.306 ± 0.04**
		b	0.218 ± 0.01	0.201 ± 0.06	0.311 ± 0.02	0.109 ± 0.06	0.146 ± 0.141	0.231 ± 0.03	0.274 ± 0.01	0.336 ± 0.17	**0.412 ± 0.02**
	B	c	43.34 ± 5.2	264.43 ± 21.64	8.45 ± 4.2	58.01 ± 4.3	124.6 ± 6.4	10.23 ± 3.1	28.26 ± 2.6	119.6 ± 10.4	**5.86 ± 1.9**
		d	518.55 ± 0.55	598.65 ± 0.84	499.69 ± 0.4	659.9 ± 0.16	652.6 ± 0.26	636.4 ± 0.07	340.26 ± 0.06	387.4 ± 0.04	**307.5± 0.05**
S-II	U	a	0.241 ± 0.05	0.224 ± 0.07	0.356 ± 0.1	0.202 ± 0.02	0.200 ± 0.06	0.331 ± 0.02	0.485 ± 0.07	0.483 ± 0.02	**0.8481± 0.02**
		b	0.105 ± 0.01	0.106 ± 0.03	0.321 ± 0.09	0.09 ± 0.01	0.121 ± 0.04	0.298 ± 0.04	0.185 ± 0.08	0.321 ± 0.02	**0.403 ± 0.02**
	P	a	0.637 ± 0.11	0.592 ± 0.09	0.812 ± 0.13	0.071 ± 0.01	0.062 ± 0.01	0.035 ± 0.01	0.585 ± 0.15	0.576 ± 0.14	**0.836 ± 0.06**
		b	0.121 ± 0.06	0.122 ± 0.08	0.224 ± 0.01	0.101 ± 0.03	0.102 ± 0.04	0.101 ± 0.02	0.100 ± 0.04	0.101 ± 0.03	**0.390± 0.05**
	B	c	1123.69 ± 12.7	1202.1.7 ± 0.04	102.454 ± 11.9	1229.82 ± 32.8	1364.7 ± 10.4	116.42 ± 18.4	1178.24 ± 8.9	962.4 ± 10.4	**64.1 ± 4.8**
		d	6294.58 ± 0.08	6746.24 ± 0.02	5928.72 ± 0.01	9761.52 ± 0.1	9842 ± 0.04	9285.6 ± 0.02	4590 ± 0.09	4656.8 ± 0.02	**3920.18 ± 0.07**
ML	U	a	0.197 ± 0.01	0.210 ± 0.01	0.126 ± 0.06	0.136 ± 0.04	0.141 ± 0.02	0.1679 ± 0.05	0.129 ± 0.08	0.196 ± 0.02	**0.215 ± 0.04**
		b	0.124 ± 0.02	0.132 ± 0.11	0.246 ± 0.05	0.136 ± 0.01	0.142 ± 0.06	0.389 ± 0.08	0.197 ± 0.01	0.271 ± 0.06	**0.466 ± 0.11**
	P	a	0.009 ± 0.07	0.101 ± 0.01	0.101 ± 0.02	0.129 ± 0.09	0.162 ± 0.03	0.196 ± 0.06	0.212 ± 0.02	0.108 ± 0.01	**0.228 v 0.06**
		b	0.246 ± 0.12	0.294 ± 0.02	0.301 ± 0.02	0.312 ± 0.01	0.264 ± 0.01	0.358 ± 0.03	0.326 ± 0.09	0.113 ± 0.05	**0.424 ± 0.03**
	B	c	38.6 ± 7.4	124.6 ± 2.5	30.5 ± 1.4	35.1 ± 2.9	112.2 ± 3.6	28.4 ± 6.4	21.24 ± 1.4	98.7 ± 3.3	**14.30 ± 2.9**
		d	749.29 ± 0.42	760.4 ± 0.01	730.95 ± 0.19	788.08 ± 0.12	796.6 ± 0.04	760.02 ± 0.18	696.17 ± 0.49	683.6 ± 0.01	**671.7 ± 0.32**
SC	U	a	0.289 ± 0.04	0.122 ± 0.04	0.305 ± 0.02	0.292 ± 0.03	0.161 ± 0.01	0.321 ± 0.04	0.329 ± 0.04	0.231 ± 0.02	**0.351± 0.08**
		b	0.374 ± 0.07	0.101 ± 0.04	0.518 ± 0.11	0.281 ± 0.03	0.009 ± 0.01	0.521 ± 0.04	0.277 ± 0.04	0.201 ± 0.01	**0.527 ± 0.05**
	P	a	0.298 ± 0.01	0.106 ± 0.07	0.344 ± 0.09	0.312 ± 0.01	0.112 ± 0.01	0.241 ± 0.09	0.311 ± 0.09	0.261 ± 0.09	**0.351 ± 0.01**
		b	0.178 ± 0.03	0.113 ± 0.02	0.236 ± 0.01	0.201 ± 0.02	0.121 ± 0.04	0.241 ± 0.10	0.200 ± 0.12	0.102 ± 0.01	**0.257 ± 0.03**
	B	c	1498.4 ± 12.5	2642.6 ± 23.7	149.5 ± 38.8	1298.5 ± 34.9	4364.36 ± 12.4	246 ± 13.9	1256.37 ± 56.2	3641.78 ± 34.7	**107.15 ±11.3**
		d	60457.47 ± 1.21	61362.2 ± 0.04	54538.27 ± 1.14	58694.03 ± 3.21	58961.7 ± 0.01	56022.87 ± 3.42	52889 ± 0.45	53316.8 ± 0.14	**47765.7 ± 0.34**

Table 5: Experimental results for explainability evaluation. The boldface means the best results. The 'a' represents Stability, 'b' represents Avg. Precision, 'c' represents CPU Time (sec) and 'd' represents Compression (Kb). Here 'U' = User, 'P' = Product and 'B' = Both.

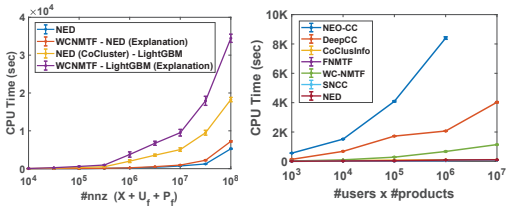


Figure 5: (a) Total running time (averaged over 10 runs) of NED versus the total number of non-zeros for Snapchat data (b) Co-clustering running time for synthetic data.

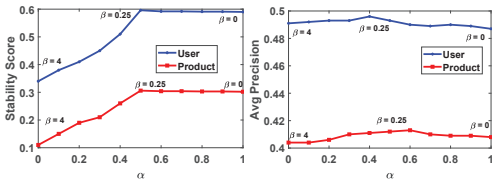


Figure 6: Sensitivity of interpolation parameters for co-cluster explanation; values of $\alpha = 0.5$ (interpolating direct path with indirect metapath matrices) produce best explanation results.

Beauty & Lifestyle	Family & Hobbies	Music
Cosmetics	Parenting & child care	Urban hip-hop
Women Fashion	Home Improvement	Electronic Dance Music
Women Lifestyle	Shopping	Pop Music
Spa beauty	Gardening	Concerts
Nail Art	Decor Design	Classical Opera

Table 6: Illustration of niches discovered by NED.

Snapchat data with 5 million viewer and 7500 publisher. To generate real graphs of growing size, we increasingly sample the Snapchat user-product user engagement matrix rows and report NED runtime averaged over 10 runs in Figure 5. We can observe that the run-time of NED scales approximately linearly; notably, NED can handle interaction matrices with many millions of interactions in mere minutes. We show only linear scaling for Snapchat dataset because (a) it is the largest dataset, and (b) the scaling trends are consistent with other datasets.

4.6 Parameter Sensitivity Analysis

We evaluate the sensitivity of the interpolation parameters α, β and γ in Equ. 11-12 which describe involvement of each meta-path matrix. We learn stability and average precision score for different combinations of $\{\alpha, \beta, \gamma\}$ on synthetic data SYN-I. Here, we kept $\beta = \gamma$ to give equal importance to 4-step meta path. Figure 6(left) shows that NED performs better when higher importance is given to direct paths i.e $\alpha \geq 0.5$. Beyond $\alpha = 0.5$, there is no significant change in performance. Moreover, Figure 6 (right) shows that average precision achieves optimal values around $\alpha = 0.5$, suggesting that incorporating indirect interactions via meta-paths beyond just direct paths does contribute to improved performance. Hence, we choose $\alpha = 0.5, \beta = \gamma = 0.25$ in our experiments.

4.7 NED at Work

We use NED on a private viewer-publisher interaction dataset from Snapchat, consisting of $500K$ viewers, 2500 publishers, and 5 million viewer-publisher interactions (here, viewers correspond to users, and publishers to products, for consistency with our prior discussion). The dataset is naturally unlabeled, making NMI and accuracy analysis infeasible; therefore, we resort to qualitative discussion. In this dataset, each viewer is described by 22 associated features (age, gender, country, etc.) and each publisher has 238 features (e.g. publisher demographics, publishing category etc.). We ran NED with the maximum number of user clusters $M = 100$ and product clusters $N = 25$. For explainability evaluation, we use top $N = 5$ highest feature values computed with Equ. 15 and 16. NED finds viewer-publisher co-clusters of various sizes in Snapchat data. Table 6 provide three major niches based on high PMI values from the summary matrix S.

The viewers clusters labeled as '1' in Figure 7 mostly belong to groups of young women (age between $13 - 20$) associated with publisher who publish content regarding 'Beauty & Lifestyles' category. We observe that viewers in this dense group have similar content consumption. The viewers cluster '2' contains viewers with connections across many publishers clusters like 'Home & Family' and 'Hobbies & Interests'. All of the viewers are between two age groups $25 - 34$ and > 35. This niche does not have any gender dominance in viewer cluster but we observe that most of the content creators or publishers are men. Niche represented by '3' is the most

prominent: this is a dense group of male viewers, > 70% of them are from North America and all of them are between age group 21 − 24, highly associated with content related to 'World Music', published by group of male publishers between age group 25 − 34. Most of the publishers are from Europe. Overall as shown in Table 5, NED outperforms state-of-the-art approaches by ≈ 6% stability and ≈ 4% average precision improvement. Also, NED achieves at least 10% compression bits and 25% run time reduction (See Table 5) for Snapchat dataset. All in all, by using NED, we are able to understand and explain the user content consumption graph in a completely unsupervised fashion.

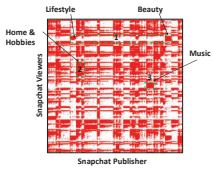


Figure 7: NED on Snapchat data finds clusters of viewers/publishers with similar attributes coherence.

5 RELATED WORK

Co-clustering: Most prior clustering literature has focused on one-sided clustering algorithms like k-means and its parameter-free variants [16, 33], spectral [23, 68], and probabilistic clustering [48, 67]. Our problem deals with simultaneous clustering of rows and columns, known as bi(dimensional)-, co-, or block clustering [9, 43] that can be categorized into following three main categories: information-theoretic, decomposition-based, and neural methods, which we discuss below.

The paper [12] introduced a co-clustering that utilizes a lossy coding scheme to co-cluster a two dimensional joint probability distribution, and it requires the number of clusters as input. [18] proposes a parameter-free and a fast information-theoretic agglomerative co-clustering method. [61] proposed a co-clustering method that allow rows and column clusters to overlap with each other. [8] developed a hierarchical structure for rows and columns with a minimum number of leaf clusters to realize co-clustering. Most recently, [50] proposed improvements to [12].

[21] proposed two regularization terms to use the geometric structure data of data graph and feature graph separately when using semi-NMTF decomposition. [10, 11, 30, 37] proposed a fast approach to constrain factor matrices to be cluster indicator matrices. [45, 55] proposed a co-clustering method from the perspective of dynamical synchronization. Most recently, [52] proposed a word co-occurrence NMTF method that leverages mutual information for co-clustering words and documents.

The paper [62] used the deep auto-encoder to map data to a low-dimensional space and then minimize the KL difference between cluster assignments and an auxiliary distribution distribution. Most recently, [63] proposed a deep co-clustering model, DeepCC, which used a deep auto-encoder to generate low-dimensional representations for rows and columns and used GMM framework for cluster assignment prediction.

Explainable Machine Learning In the last decade, explainable machine learning has gained considerable attention. Prior work shows that the ability of intelligent systems to justify their choices is important for their successful use; when users do not understand the decisions of an intelligent system, they become cynical and unwilling to use it, despite improved performance [20, 28]. Several works aim to explain complex predictive models with simple rule-based explanations; rule-based explanations [6, 15, 41, 42] and deep learning based explanations [47] have been a popular approach to explain black-box models. However, these methods are often tailored to the specifics of the model which is being explained.

Recently, another line of work has focused on explaining predictions of complex models in terms of the importance of features in the classification. [57] proposed an ablation-style approach which removes all possible subsets of features and evaluates changes in predictions. However, such combinatorial approaches are computationally expensive. [40] improved the computations described in [57] and proposed efficient method to interpret model prediction using weights on features, representing their relative contribution on the prediction using Shapley values, and effectively applying to any downstream classification model. Tree-based ensemble methods [39] such as random forests [38] and gradient boosted trees [1, 29] achieve state-of-the-art performance in many domains. They have a successful history of use in machine learning, and new high-performance implementations are an active area of research [7, 29, 51]. Such models often outperform standard deep models [56] on datasets where features are individually meaningful and do not have strong temporal or spatial structures [7, 53]. Most recently, [29] proposed LightGBM and [2] proposed BMGUFS to enhance the performance of tree-based models and NMTF model.

6 CONCLUSION

In this work, we tackle the problem of discovering market niches for strategic content creation to satisfy diverse audience groups. We pose the niche detection problem as one which involves discovering coherent co-clusters in user-product (content) interaction graph data, as well as explaining the co-clusters using nodal attributes on user and product nodes. To our knowledge, ours is the first work which tackles such an explainable co-clustering problem. We proposed NED, the first niche detection framework for finding and explaining co-clusters in attributed interaction graphs. NED utilizes principles from mutual information to (a) propose and solve a non-negative matrix tri-factorization oriented objective to discover cohesive co-clusters, and (b) select important user and product features associated with these co-clusters using meta-path driven feature selection. In doing so, we find that NED successfully discovers niches in user-content consumption data, and demonstrates improvements over both state-of-the-art co-clustering approaches (≈ 14% accuracy) as well as candidate explanation approaches (≈ 20% average precision) on multiple simulated and real-world datasets. Finally, we show that NED provides an advantageous speed-quality tradeoff over alternative approaches, and scales effortlessly and discovers interesting insights on large-scale interaction data with over *60M* interactions, via a private dataset from Snapchat.

7 ACKNOWLEDGEMENTS

Research was supported in part by the National Science Foundation under CAREER grant no. IIS 2046086 and grant no. 1901379. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding parties.

REFERENCES

[1] Maria Athanasiou, Konstantina Sfrintzeri, Konstantia Zarkogianni, Anastasia C Thanopoulou, and Konstantina S Nikita. 2020. An explainable XGBoost-based approach towards assessing the risk of cardiovascular disease in patients with Type 2 Diabetes Mellitus. *arXiv preprint arXiv:2009.06629* (2020).

[2] Zilong Bai, Hoa Nguyen, and Ian Davidson. 2020. Block Model Guided Unsupervised Feature Selection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1201–1211.

[3] Arindam Banerjee, Inderjit Dhillon, Joydeep Ghosh, Srujana Merugu, and Dharmendra S Modha. 2007. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *Journal of Machine Learning Research* 8, Aug (2007), 1919–1986.

[4] Deng Cai, Xiaofei He, Jiawei Han, and Thomas S Huang. 2010. Graph regularized nonnegative matrix factorization for data representation. *IEEE transactions on pattern analysis and machine intelligence* 33, 8 (2010), 1548–1560.

[5] Deng Cai, Xiaofei He, Xiaoyun Wu, and Jiawei Han. 2008. Non-negative matrix factorization on manifold. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 63–72.

[6] Richard G Carter and John Levine. 2007. An investigation into tournament poker strategy using evolutionary algorithms. In *2007 IEEE Symposium on Computational Intelligence and Games*. IEEE, 117–124.

[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[8] Wei Cheng, Xiang Zhang, Feng Pan, and Wei Wang. 2016. HICC: an entropy splitting-based framework for hierarchical co-clustering. *Knowledge and Information Systems* 46, 2 (2016), 343–367.

[9] Yizong Cheng and George M Church. 2000. Biclustering of expression data.. In *Ismb*, Vol. 8–2000. 93–103.

[10] Andrej Čopar, Blaž Zupan, and Marinka Zitnik. 2019. Fast optimization of non-negative matrix tri-factorization. *PloS one* 14, 6 (2019), e0217994.

[11] Waldyr L de Freitas, Sarajane M Peres, Valdinei Freire, and Lucas Fernandes Brunialti. 2020. OvNMTF Algorithm: an Overlapping Non-Negative Matrix Tri-Factorization for Coclustering. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–8.

[12] Inderjit S Dhillon, Subramanyam Mallela, and Dharmendra S Modha. 2003. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 89–98.

[13] Chris Ding, Tao Li, Wei Peng, and Haesun Park. 2006. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. 126–135.

[14] Chris HQ Ding, Tao Li, and Michael I Jordan. 2008. Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence* 32, 1 (2008), 45–55.

[15] David R Duling. 2017. Use Machine Learning to Discover Your Rules. *Paper SAS579-2017* (2017).

[16] Maziar Moradi Fard, Thibaut Thonet, and Eric Gaussier. 2020. Deep k-means: Jointly clustering with k-means and learning representations. *Pattern Recognition Letters* 138 (2020), 185–192.

[17] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[18] Tiantian Gao and Leman Akoglu. 2014. Fast information-theoretic agglomerative co-clustering. In *Australasian Database Conference*. Springer, 147–159.

[19] Gérard Govaert and Mohamed Nadif. 2013. *Co-clustering: models, algorithms and applications*. John Wiley & Sons.

[20] Shirley Gregor and Izak Benbasat. 1999. Explanations from intelligent systems: Theoretical foundations and implications for practice. *MIS quarterly* (1999), 497–530.

[21] Quanquan Gu and Jie Zhou. 2009. Co-clustering on manifolds. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. 359–368.

[22] PMARCA GUIDE. 2007. The only thing that matters. https://pmarchive.com/guide_to_startups_part4.html. [Online].

[23] Ekta Gujral, Evangelos E Papalexakis, Georgios Theocharous, and Anup Rao. 2019. Hacd: Hierarchical Agglomerative Community Detection In Social Networks. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 1–6.

[24] Junwei Han, Kun Song, Feiping Nie, and Xuelong Li. 2017. Bilateral k-means algorithm for fast co-clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[25] Junwei Han, Kai Xiong, and Feiping Nie. 2017. Orthogonal and Nonnegative Graph Reconstruction for Large Scale Clustering.. In *IJCAI*. 1809–1815.

[26] John A Hartigan. 1972. Direct clustering of a data matrix. *Journal of the american statistical association* 67, 337 (1972), 123–129.

[27] Parisa Kaghazgaran, Maarten Bos, Leonardo Neves, and Neil Shah. 2020. Social Factors in Closed-Network Content Consumption. In *CIKM*.

[28] Ujwal Kayande, Arnaud De Bruyn, Gary L Lilien, Arvind Rangaswamy, and Gerrit H Van Bruggen. 2009. How incorporating feedback mechanisms in a DSS affects DSS evaluations. *Information Systems Research* 20, 4 (2009), 527–546.

[29] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. In *NeurIPS*. 3146–3154.

[30] Jungeun Kim, Jae-Gil Lee, Byung Suk Lee, and Jiajun Liu. 2020. Geosocial Co-Clustering: A Novel Framework for Geosocial Community Detection. *ACM Transactions on Intelligent Systems and Technology (TIST)* 11, 4 (2020), 1–26.

[31] Danai Koutra, Abhilash Dighe, Smriti Bhagat, Udi Weinsberg, Stratis Ioannidis, Christos Faloutsos, and Jean Bolot. 2017. Pnp: Fast path ensemble method for movie design. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1527–1536.

[32] Abhishek Kumar, Piyush Rai, and Hal Daume. 2011. Co-regularized multi-view spectral clustering. In *NeurIPS*. 1413–1421.

[33] Rashmi Kumari, MK Singh, R Jha, NK Singh, et al. 2016. Anomaly detection in network traffic using K-mean clustering. In *2016 3rd International Conference on Recent Advances in Information Technology (RAIT)*. IEEE, 387–393.

[34] Hemank Lamba and Neil Shah. 2019. Modeling dwell time engagement on visual multimedia. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1104–1113.

[35] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. 2003. The global k-means clustering algorithm. *Pattern recognition* 36, 2 (2003), 451–461.

[36] Bo Long, Zhongfei Zhang, and Philip S Yu. 2005. Co-clustering by block value decomposition. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. 635–640.

[37] Zhoumin Lu, Genggeng Liu, and Shiping Wang. 2020. Sparse neighbor constrained co-clustering via category consistency learning. *Knowledge-Based Systems* 201 (2020), 105987.

[38] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2019. Explainable AI for trees: From local explanations to global understanding. *arXiv preprint arXiv:1905.04610* (2019).

[39] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. 2020. From local explanations to global understanding with explainable AI for trees. *Nature machine intelligence* 2, 1 (2020), 2522–5839.

[40] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *NeurIPS*. 4765–4774.

[41] David Martens, Bart Baesens, Tony Van Gestel, and Jan Vanthienen. 2007. Comprehensible credit scoring models using rule extraction from support vector machines. *European journal of operational research* 183, 3 (2007), 1466–1476.

[42] David Martens and Foster Provost. 2014. Explaining data-driven document classifications. *Mis Quarterly* 38, 1 (2014), 73–100.

[43] Boris Mirkin. 2013. *Mathematical classification and clustering*. Vol. 11. Springer Science & Business Media.

[44] David Newman, Sarvnaz Karimi, and Lawrence Cavedon. 2009. External evaluation of topic models. In *in Australasian Doc. Comp. Symp., 2009*. Citeseer.

[45] Feiping Nie, Xiaoqian Wang, Cheng Deng, and Heng Huang. 2017. Learning a structured optimal bipartite graph for co-clustering. In *NeurIPS*. 4129–4138.

[46] Sarah Nogueira and Gavin Brown. 2016. Measuring the stability of feature selection. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 442–457.

[47] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[48] Tommaso Rigon, Amy H Herring, and David B Dunson. 2020. A generalized Bayes framework for probabilistic clustering. *arXiv preprint arXiv:2006.05451* (2020).

[49] Jorma Rissanen. 1983. A universal prior for integers and estimation by minimum description length. *The Annals of statistics* (1983), 416–431.

[50] François Role, Stanislas Morbieu, and Mohamed Nadif. 2018. Coclust: a python package for co-clustering. *PloS one* (2018).

[51] Steven J Rosansky, Donald R Hoover, Lisa King, and James Gibson. 1990. The association of blood pressure levels and change in renal function in hypertensive and nonhypertensive subjects. *Archives of internal medicine* 150, 10 (1990), 2073–2076.

[52] Aghiles Salah, Melissa Ailem, and Mohamed Nadif. 2018. Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[53] Prathyush Sambaturu, Aparna Gupta, Ian Davidson, SS Ravi, Anil Vullikanti, and Andrew Warren. 2020. Efficient Algorithms for Generating Provably Near-Optimal Cluster Descriptors for Explainability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 1636–1643.

[54] Fanhua Shang, LC Jiao, and Fei Wang. 2012. Graph dual regularization non-negative matrix factorization for co-clustering. *Pattern Recognition* 45, 6 (2012), 2237–2250.

[55] Junming Shao, Chongming Gao, Wei Zeng, Jingkuan Song, and Qinli Yang. 2017. Synchronization-inspired co-clustering and its application to gene expression

data. In *2017 IEEE Int. Conf. on Data Mining (ICDM)*. IEEE, 1075–1080.
[56] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. *arXiv preprint arXiv:1704.02685* (2017).
[57] Erik Strumbelj and Igor Kononenko. 2010. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research* 11 (2010), 1–18.
[58] Qi Su, Kun Xiang, Houfeng Wang, Bin Sun, and Shiwen Yu. 2006. Using pointwise mutual information to identify implicit features in customer reviews. In *International Conference on Computer Processing of Oriental Languages*. Springer, 22–30.
[59] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. 2011. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. *Proceedings of the VLDB Endowment* 4, 11 (2011), 992–1003.
[60] Shiping Wang and Wenzhong Guo. 2017. Robust co-clustering via dual local learning and high-order matrix factorization. *Knowledge-Based Systems* 138 (2017), 176–187.
[61] Joyce Jiyoung Whang and Inderjit S. Dhillon. 2017. Non-Exhaustive, Overlapping Co-Clustering. In *Proceedings of the 26th ACM Conference on Information and Knowledge Management (CIKM)*. 2367–2370.
[62] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*. 478–487.
[63] Dongkuan Xu, Wei Cheng, Bo Zong, Jingchao Ni, Dongjin Song, Wenchao Yu, Yuncong Chen, Haifeng Chen, and Xiang Zhang. 2019. Deep co-clustering. In *Proceedings of the 2019 SIAM SDM*. SIAM, 414–422.
[64] Wei Xu, Xin Liu, and Yihong Gong. 2003. Document clustering based on nonnegative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. 267–273.
[65] Jiho Yoo and Seungjin Choi. 2010. Orthogonal nonnegative matrix tri-factorization for co-clustering: Multiplicative updates on stiefel manifolds. *Information processing & management* 46, 5 (2010), 559–570.
[66] Zhijian Yuan, Zhirong Yang, and Erkki Oja. 2009. Projective nonnegative matrix factorization: Sparseness, orthogonality, and clustering. *Neural Process. Lett* (2009), 11–13.
[67] Peng Zhang and Kun She. 2020. A Novel Hierarchical Clustering Approach Based on Universal Gravitation. *Mathematical Problems in Engineering* 2020 (2020).
[68] Xiaofeng Zhu, Yonghua Zhu, and Wei Zheng. 2020. Spectral rotation for deep one-step clustering. *Pattern Recognition* 105 (2020), 107175.