

Article

Fund Similarity: A Use of Bipartite Graphs

Ren-Raw Chen *, Liangbingyan Luo, Yihui Wang and Xiaohu Zhang

Gabelli School of Business Administration, Fordham University, New York, NY 10023, USA; lluo25@fordham.edu (L.L.); ywang151@fordham.edu (Y.W.); xzhang25@fordham.edu (X.Z.)

* Correspondence: rchen@fordham.edu

Abstract

Fund similarity is important for investors when constructing diversified portfolios. Because mutual funds do not always adhere closely to their stated investment policies, investors may unintentionally hold funds with overlapping exposures, leading to reduced diversification and instead causing “diworsification”, which is an investment term for when too much complexity leads to worse results. As a result, various quantitative methods have been proposed in the literature to investigate fund similarity, primarily using portfolio holdings. Recently, machine learning tools such as clustering and graph theory have been introduced to capture fund similarity. This paper builds on this literature by applying bipartite graphs and node2vec embeddings to a comprehensive dataset that covers 3874 funds over a nearly 6-year period. Our empirical evidence suggests that, bipartiteness is not preserved for non-index (active) funds. Furthermore, while graph embeddings yield higher similarity scores than holding-based measures, they do not necessarily outperform holding-based similarity in explaining returns. These findings suggest that graph-based embeddings capture structural relationships among funds that are distinct from direct portfolio overlap but are not sufficient substitutes when similarity is evaluated solely through returns. As a result, we recommend a more comprehensive similarity measure that includes important risk metrics such as volatility risk, liquidity risk, and systemic risk.

Keywords: bipartite graph; fund similarity; Jaccard index; diworsification

1. Introduction

When investing in mutual funds, one must watch out for “diworsification”, which is an investment term for when too much complexity leads to worse results. Mutual funds often differ in name, stated objective, or mandate, yet hold similar positions. As a result, investors may believe they are diversified by investing in multiple mutual funds when, in fact, they are exposed to the same risk factors (Sensoy [1]). That is, they acquire too many funds that are too similar and, as a result, lose the benefits of diversification. See Sakakibara et al. [2], for example, for detailed discussions.

Similarity also has important implications for risk management and market stability. When funds hold similar assets, correlated investor flows can force simultaneous buying or selling, amplifying liquidity risk and price dislocations (Nanda and Wei [3]; Delpini et al. [4]). In addition, similar funds face more intensive competition in the market (Wahal and Wang [5]).

Consequently, understanding whether two funds are truly similar is crucial for investors. Various services can be found online that enable investors to find similarities between funds. For example, a famed rating agency, Morningstar, provides



Academic Editor: Yong Gao

Received: 14 November 2025

Revised: 2 January 2026

Accepted: 7 January 2026

Published: 13 January 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

a tool to help investors find similar funds (See https://awgmain.morningstar.com/webhelp/tools/Find_Similar_Funds.htm, accessed on 5 January 2026. There are numerous other examples, such as <https://thefundoo.com/Tools/PortfolioOverlap> and <https://primeinvestor.in/mutual-funds-overlap/>, accessed on 5 January 2026).

Recently, as opposed to simply comparing the holdings of funds to determine similarity, machine learning techniques have been adopted, one of which is graph theory (in particular, bipartite graphs). The bipartite graph is believed to have first been used by Delpini et al. [4] and then extended (using node2vec) by Satone, Desai, and Mehta [6]. Graphs are good for studying fund similarities in that (1) they provide a good description of a network, and (2) funds and their invested assets form a natural bipartite graph.

A bipartite graph is a type of graph that has vertices that are separated into two groups. Vertices (nodes) are not connected within a group but only across groups. As a result, it is natural that funds are in one group and their invested assets are in the other group. In this paper, we follow Satone, Desai, and Mehta [6], who use a bipartite graph to analyze fund similarities. In their work, Satone, Desai, and Mehta convert the bipartite graph into a corpus using node2vec (Grover and Leskovec [7]), which is an NLP (natural language processing) technique. A brief introduction to node2vec is given in an Appendix A. Simply speaking, node2vec takes a graph and treats each vertex (node) as a word, and a path from one vertex (node) to other vertices as a sentence. By randomly traveling many times, node2vec forms a corpus. Such a corpus is then fed to NLP to process the similarities.

There are two key methods used in NLP in our analysis: (1) embedding and (2) dimension reduction. Embedding, or word embedding, or graph embedding, is a technique used to transform a word (or a sequence of words) into a vector of numbers. These numbers can then be recognized by computers (computer language). The numbers are created in such a way that they can mimic natural language. Put differently, similar sentences in natural language are translated to similar embeddings. That is, via embedding, computers can communicate with humans.

The second piece of this technology is dimension reduction. We used t-SNE for this. A vector generated by embedding has a long string of numbers (e.g., 1536 for OpenAI, 300 for Google), which represents a high number of dimensions. These high-dimensional vectors are hard to presented in a viewable manner (i.e., in two or three dimensions). Hence, mapping is necessary. Dimension reduction in NLP is a wide topic, and we chose t-SNE to reduce our results to two dimensions.

The main finding of Satone, Desai, and Mehta [6] is their confirmation of the utility of bipartite graphs in analyzing fund similarity. However, their study has two important limitations. First, their analysis focuses exclusively on index funds, which suffer from substantial overlaps of asset holdings, hence causing a tautological result in their bipartite graphs (details are presented in our empirical section later). Secondly, their data only cover a short period of time—the first quarter of 2020. Following their line of work, we summarize our contributions to the literature as follows:

- We adopt a much larger sample of data than Satone, Desai, and Mehta [6] by including a large collection of funds (3874 funds) over a longer time period (June 2018 to March 2024). Through this method, we cannot validate their empirical findings.
- Unlike Satone, Desai, and Mehta [6], we avoid index funds, resulting in a rejection of their hypothesis that similarity is significant in a bipartite graph.
- The new performance result indicates that bipartite graphs are not necessarily better than plain portfolio weights at explaining similarity.
- Finally, a broader inclusion of features, not just returns, is necessary in studying fund similarity. Systemic risk, for example, has shown to impact performance during a crisis, for which Chen and Zhang [8] have discovered that graphs are useful.

The paper is organized as follows. We first provide a short literature review of fund similarity. Then, the entire process of using node2vec, which includes doc2vec, Jaccard index, v-measure, and t-SNE, is introduced. Experienced modelers who are familiar with these techniques can skip this section. Section 4 is the main section of this paper, presenting the main empirical findings. Finally, we conclude.

2. Literature Review of Fund Similarities and Bipartite Graph

2.1. Fund Similarity

Although the interest in fund similarities is prevalent, the academic literature is quite limited. The literature is broadly divided into two distinct categories: network and performance. The former only focuses on how funds are structured and why, while the latter also investigates how the performances of funds are connected to their similarities. In this paper, we use a large dataset to address both issues.

Razafitombo [9] studies 551 equity funds from the Reuters–Lipper database. The data period covers 211 investment funds from 2000 to 2007. To estimate the degree of similarity between 15 performance measures. He concludes that similarities matter in fund performance.

Sakakibara et al. [2] classify Japanese mutual funds using a network stretched edge between funds with weights for the number of common stocks in the top 10 stocks in each mutual fund. Then they cluster the funds using k-means and spectral clustering methods. Their network result is a one-time snapshot and hence is unable to provide specific performance implications of the network.

Nanda and Wei [3] use fund similarities to study funds' exposure to liquidity risk. They find strong evidence that mutual funds mitigate such liquidity risk exposure by reducing portfolio overlap with peer funds when their flows become more correlated. Reducing similarity mitigates the risk of concurrent selling (or buying) of the same stocks by peer funds. They use quarterly portfolio holdings for U.S. equity mutual funds during 1993–2015 from Thomson Reuters for portfolio weights.

Delpini et al. [4] study fund similarities in the financial crisis. They treat funds as a bipartite network. They find that portfolios became less similar during the crisis. They only study three reference quarters: 2006Q2, 2008Q2, and 2010Q2. The bipartite network of funds provides a “degree” which is defined as the number of different assets held at a given moment in time. They also perform simulations of shocks on the bipartite graph in order to test network fragility.

Satone, Desai, and Mehta [6] use funds in the first quarter of 2020 to draw a bipartite graph. Then they calculate “bipartiteness” by a v-measure, which is a combination of fund completeness and homogeneity. They then use the v-measure to test how much bipartiteness is preserved.

McLemore et al. [10] study fund similarities by looking at patent distributions. Using CRSP-matched Google patent data, which includes patent filings from 1926 to 2017, they calculate fund similarities based on how funds own patents (via firms). They conclude that those funds that own a substantial number of patents have superior monthly returns (by 282 basis points).

Finally, Desai et al. [11] argue that fund labeling is often mistaken. Hence, using descriptive labeling to classify funds is inaccurate, causing investors not receive what they think they should. They use random forest to find how two funds are similar (i.e., in close proximity in a tree), which is then used to identify mis-categorization. Analyzing 1167 funds with 373 features from MorningStar, they find that there is significant mis-categorization of MorningStar classification. However, as in Sakakibara et al. [2], their

result is a one-time snapshot and cannot provide the implications of the consequences of mis-categorization.

2.2. Applications of Bipartite Graph

The literature on bipartite graphs is voluminous. Here, a brief review is based on the top rankings from Google Scholar.

While much of the fund similarity literature relies on holdings overlap or performance co-movement, recent work emphasizes the value of representing funds and assets as networks. In such settings, similarity is determined not only by direct overlap but also by a fund's position in the broader investment network. Bipartite fund–asset graphs provide a natural representation of this structure and have been used to study diversification, contagion, and systemic risk in financial markets (Delpini et al. [4]; Chen and Zhang, [8]). Our study builds on this insight by evaluating whether graph-based similarity measures offer incremental information relative to traditional holdings-based benchmarks.

As its name implies, bipartite graphs are best for matching problems. For example, Tanimoto et al. [12] give various matching examples using bipartite graphs. Bipartite graphs can be used in other areas, such as (The list is provided by geeksforgeeks. See their website: <https://www.geeksforgeeks.org/maths/real-life-applications-of-bipartite-graph>, accessed on 5 January 2026):

- Recommendation systems (see Maier and Simovici [13])
- Social Networks (see Zhu, Su, and Kong [14])
- Resource Allocation (see Wang, Wang, and Chen [15])
- Economic Markets (see Fernandez, Galeano, and Hidalgo [16])
- Information Retrieval (see Cao et al. [17])
- Transportation Networks (see Anderson and Farooq [18])
- Customer–Product Purchases (see Zhang et al. [19])
- Author–Paper Networks (see Morris and Yen [20])
- Music Playlist Generation (see Zhang, Zhao, and Zhao [21])
- Drug–Target Interaction Networks (see Manoochehri and Nourani [22])

While there are numerous applications of bipartite graphs as mentioned above, using it in fund similarity is new. As reviewed in the previous sub-section, Delpini et al. [4] were the first to apply bipartite graphs on fund similarity and extended by Satone, Desai, and Mehta [6]. Our contribution to this line of research is to apply a comprehensive dataset that covers all funds over a long period of time.

3. A Sketch of Bipartite Graph and NLP Tools (node2vec, Jaccard Index, v-Measure, and t-SNE)

In this section, we briefly describe the algorithms behind node2vec/word2vec, similarity/Jaccard index, v-measure (for testing bipartiteness), and t-SNE. Those who are already familiar with these topics can skip this section entirely.

3.1. Building a Bipartite Graph

A bipartite graph is a graph whose vertices are divided into two disjoint sets, where there are no edges connecting vertices within each set (i.e., edges can only exist for vertices in different sets). A sample of a bipartite graph is given (taken from Wikipedia) below in Figure 1 (The example is taken from https://en.wikipedia.org/wiki/Bipartite_graph, accessed on 13 November 2025). Panel (a) of Figure 1 is clearly seen as a bipartite graph in that one set of vertices (numerals) is on one side, where no edges exist, while the other set of vertices (alphabets) is on the other side, where no edges exist. The only edges can be found between the vertices of different sets.

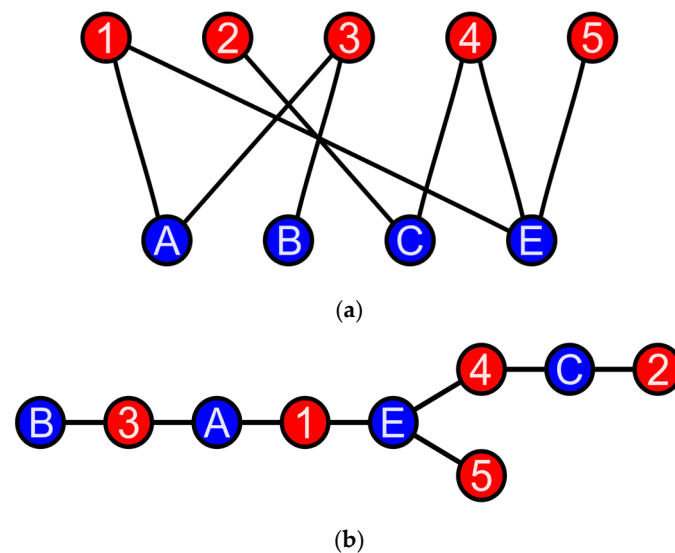


Figure 1. An Example of a bipartite graph (taken from Wikipedia). (a) An easily interpretable bipartite graph. (b) The shown chain graph is also a bipartite graph (isomorphic to (a)). Source: Wikipedia.

We should also note that Panel (b) is also a bipartite graph, identical to Panel (a). It is called an isomorphic graph as compared to the graph in Panel (a). In graph theory, isomorphic graphs are regarded as identical graphs, as their vertices and edges are identically defined. That is, how to draw a graph on a two-dimensional plane is irrelevant as long as the relationships of the vertices are kept. Finally, we note that if every vertex in one set is connected to every vertex in the other set, then it is called a complete bipartite graph. Figure 1 is clearly not a complete bipartite graph.

When we study fund similarity, we study their similarity through their holdings. This is a bipartite graph. The bipartite graph is easy to draw because we know the holdings of each fund and the weights (edges) of the assets held by the fund.

3.2. node2vec

The key methodology used in this paper to study fund similarities is node2vec. node2vec is an approach used to turn a graph into a corpus. Then, we can embed the graph built by the node2vec method. node2vec takes a network graph and calculates the embeddings of the vertices. Often, the outcome is also presented as a graph (but without edges) (It is a vector representation, i.e., embedding.), as shown in Figure 2.

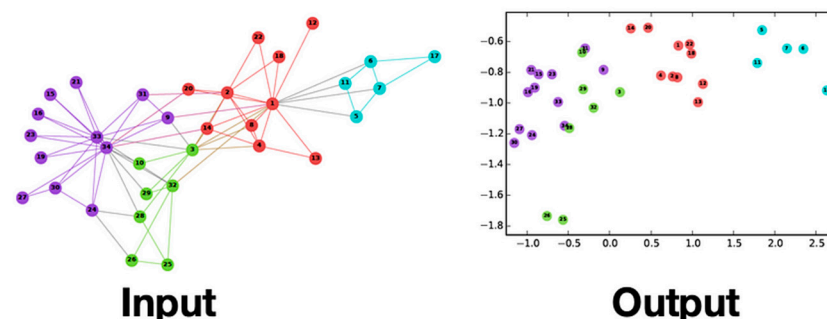


Figure 2. An Example of node2vec. A node2vec is a technique that takes a set of nodes and edges to form a path. The path is then recognized as a sentence that can be fed into an NLP process to generate a text embedding. Finally, the embedding can be used to generate a graph of a desired purpose (as shown in the right panel). Colors in the above graphs represent different clusters. Source: <https://medium.com/nlplanet/two-minutes-nlp-using-word2vec-to-learn-node-embeddings-on-graphs-ebece432a4b6>, accessed on 13 November 2025.

In this example, a graph on the left presents a network of several vertices (nodes), which represent entities of interest. Edges in the graph do not have any particular meaning (i.e., not properly quantified or unweighted) other than linking various vertices. As a result, from the graph itself, it is hard to know if two vertices are similar or not (which cannot be determined by the distance between them). `node2vec` fills the void and produces the embedding output on the right. Now the positions of the vertices are properly quantified. Both x- and y-axes provide a coordinate of a vertex. Hence, now, the distance between any two vertices has meaning, representing whether any two vertices are similar (if they are close to each other) or not (if they are far from each other) (Please see Grover and Leskovec [7] for further details.).

`node2vec`, simply speaking, mimics `word2vec` or `doc2vec` in natural language processing (NLP). It turns a word, a sentence, a document, or even a corpus into a vector of numbers, which is known as embedding. `node2vec` turns a series of vertices (nodes) into a vector of numbers, which is known as node embedding. It can be interpreted as each vertex (node) being treated as a word, and a series of nodes can be regarded as a sentence (or a document, or a corpus). A detailed example is provided in the Appendix A.

To use `node2vec`, we need to set hyperparameter values. These hyperparameters include the following:

- d : the number of dimensions in the graph
- r : the number of random walks to be generated from each vertex in the graph
- ℓ : the length (number of hops) of each random walk from each vertex in the graph
- p : the probability that a random walk will return to a vertex it has already visited previously in the graph
- q : the probability that a random walk will explore an unexplored part of the graph

3.3. Cosine Similarity and Jaccard Index

Cosine similarity is defined as

$$\chi = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

and the Jaccard index is a statistic used for gauging the similarity and diversity of sample sets. It is defined, in general, by taking the ratio of two sizes (areas or volumes), the intersection size, divided by the union size, also called intersection over union (IoU).

In our empirical work, we calculate two sets of similarity scores. One is based upon the asset holdings of the bipartite graph. Each fund has a vector of holdings (weights) of the assets (if the assets are not held by a fund, then the weights are zero). These weights are then used to calculate similarity scores. The other is based on the `node2vec` embeddings of the bipartite graph. Note that an embedding is a vector of numbers that can be used to calculate similarity scores.

The next step is to select, for each fund, its most similar funds. For example, say, for fund A, its top five most similar funds are fund B, C, D, E, and F using asset weights, and B, C, F, G, and K using `node2vec` embedding. To compute the Jaccard index, we need the union and intersection of the two sets:

- $\{B, C, D, E, F\} \cap \{B, C, F, G, K\} = \{B, C, F\}$
- $\{B, C, D, E, F\} \cup \{B, C, F, G, K\} = \{B, C, D, E, F, G, K\}$

The Jaccard index is a ratio of the number of elements in the intersection (three in this example) over the number of elements in the union (seven in this example); thus, the result is $\frac{3}{7}$.

As we can see, when the Jaccard index is one, the two sets are identical, and when it is zero, there is no overlap. Hence, this is a test to see if the two classification methods generate similar or different fund similarities. This is important, as evidence has shown that using asset holdings can lead to incorrect diversification (Satone, Desai, and Mehta [6] have tried using the top 5, 10, 20, and 50 most similar funds for each fund as a robustness test.).

The purpose here is to demonstrate that using a bipartite graph captures more structural similarities, not just how many assets overlap. It considers the whole network and finds similar funds, even with fewer nodes in common. We can use this approach for demonstration as well.

3.4. v-Measure

An important task is to verify whether funds and assets follow a bipartite graph. For this purpose, Satone, Desai, and Mehta [6] suggest a v-measure, which is defined as follows:

$$v = \frac{(1 + \beta)hc}{\beta h + c} \quad (1)$$

where

$$c = \begin{cases} 1 & \text{if } H(C, K) = 0 \\ 1 - \frac{H(K|C)}{H(K)} & \text{else} \end{cases} \quad (2)$$

and (C is class and K is cluster)

$$h = \begin{cases} 1 & \text{if } H(C, K) = 0 \\ 1 - \frac{H(C|K)}{H(C)} & \text{else} \end{cases} \quad (3)$$

Here, h represents “homogeneity”, c represents “completeness”, and β represents a weight between them. Homogeneity measures how many classes are in a given cluster. A purely homogeneous value of one occurs when the cluster contains only one class. Completeness measures how well a cluster captures elements of the class. When $\beta > 1$, completeness is weighed more than homogeneity, and when $\beta < 1$, vice versa. In an extreme case when both homogeneity and completeness are perfect (a value of one), then $v = \frac{1+\beta}{\beta+1} = 1$.

The parameter β in Equation (1) reveals an inclination towards either homogeneity ($\beta < 1$) or completeness ($\beta > 1$). Since the number of labels in a bipartite graph is two, clearly, the completeness will decrease as K increases. We follow Satone, Desai, and Mehta [6] and focus on maximizing homogeneity (systematically weighting homogeneity significantly more than completeness in the computation of the v-measure) and set $\beta = 0.01$. For example, in Satone, Desai, and Mehta, the k-means analysis yields five clusters, three of which can be identified as fund clusters and the remaining two as asset clusters. As a result, their v-measure is close to one, and hence the bipartiteness is validated.

Note that v-measure is part of the standard Python 1.8.0. library scikit-learn that can be easily found as follows: `v_measure_score(original_labels, cluster_labels, beta = 0.01)`. In `original_labels`, use 0 for assets and 1 for funds, and `cluster_labels` is the cluster number from k-means. Following prior studies, we chose $\beta = 0.01$, which places greater weight on homogeneity. This choice also reflects our objective of assessing whether fund and asset nodes remain distinguishable in embedding space, particularly in a heterogeneous and sparsely connected network of active mutual funds. Alternative values of β yield qualitatively similar conclusions.

Although it is not reported here to conserve space, we tried different dates (selected from 270+ fund dates). In addition, we also tested selecting the top 100 funds (ordered

by market value). For each round, we experimented with k-means for k from 2 to 10. The results were similar, and the v-measure ranged from 0.0+ to 0.2+.

Next, we use k-means to cluster embedded nodes to separate funds and assets nodes. Ideally, no matter the magnitude of k (number of clusters), each cluster should only contain one type of node.

Since v-measure is used in conjunction with k-means clustering, the optimal value k (which is set to be 2~10) is determined by the largest v-measure.

Optimization results are evaluated by the v-measure, which captures the mismatch between the clustering from k-means and the ground truth (asset and fund). The target of hyperparameter tuning is to maximize v-measure. The purpose is to demonstrate that the bipartiteness characteristic is preserved in the embedding.

In this paper, we use v-measure in conjunction with node2vec. Note that node2vec captures local neighborhoods. In a densely connected graph (index funds share more common holdings), the model would consistently capture the cross-partition patterns; the bipartiteness is preserved in embedded space. Two groups are positioned on two sides. In more sparsely connected graphs, it becomes harder to capture the structural patterns. Funds are surrounded by their holdings in the embedded space (more inseparable).

3.5. t-SNE

t-SNE (The description here is a tutorial material offered by datacamp (<https://www.datacamp.com/tutorial/introduction-t-sne>, accessed on 13 November 2025).), or t-distributed Stochastic Neighbor Embedding, is an unsupervised non-linear dimensionality reduction technique for data exploration and visualizing high-dimensional data. Non-linear dimensionality reduction means that the algorithm allows us to separate data that cannot be separated by a straight line. As a result, one can view t-SNE as a non-linear counterpart of PCA (principal component analysis), which is a linear dimension-reduction technique. Note that PCA is more than just a dimension reduction tool. It also provides a set of orthogonal dimensions for further data processing. However, t-SNE is just a dimension reduction tool. For example, the t-SNE result is sensitive to parameter settings such as “perplexity”, “# of iterations”, and “random state”. See van Der Maaten and Hinton [23] for further details.

t-SNE models a point that is selected as a neighbor of another point in both higher and lower dimensions. It starts by calculating a pair-wise similarity between all data points in the high-dimensional space using a Gaussian kernel. The points that are far apart have a lower probability of being picked than the points that are close together.

The algorithm then tries to map higher-dimensional data points onto a lower-dimensional space while preserving the pairwise similarities. It achieves this by minimizing the divergence between the original high-dimensional and lower-dimensional probability distributions. The algorithm uses gradient descent to minimize the divergence. The lower-dimensional embedding is optimized to a stable state.

In t-SNE, the main calculation is based on probabilities rather than distances. When two points are closer in distance, it is converted to a higher probability. When data points are mapped from the high-dimension to the lower dimension, the algorithm starts by randomly scattering points in the 2-D space, and then performs optimization to make the probability distribution as similar as possible to the original space. It performs a convergence calculation, such that for each data point, the k neighbors are those with the top- k probabilities. Mirroring or rotating the plot does not change this structure. The calculation tries to reach a balance between every pair of data points. That is why it preserves local structure rather than global structure. We select the default setting in the package scikit-learn, which is a widely accepted baseline that provides a balanced con-

figuration. A snapshot of the function is given below (The full explanation can be found in <https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html>, accessed on 5 January 2026):

```
class sklearn.manifold.TSNE(n_components=2, *, perplexity=30.0,
    early_exaggeration=12.0, learning_rate='auto', max_iter=1000,
    n_iter_without_progress=300, min_grad_norm=1e-07, metric='euclidean',
    metric_params=None, init='pca', verbose=0, random_state=None,
    method='barnes_hut', angle=0.5, n_jobs=None)
```

4. Empirical Work

4.1. Data

The data used in our study are the CRSP Mutual Funds available on WRDS. CRSP retains historical records for funds that are no longer in existence, thereby minimizing the survivorship bias. The selection criteria are listed below:

1. Fund data are available for the period of June 2018 to March 2024.
2. NAV is equal to or greater than \$10 billion.
3. Lipper Asset code indicates an equity fund: the percentage of investment in equity is equal to or greater than 85%.
4. It is a retail fund, not an institutional fund.
5. It is an active fund.

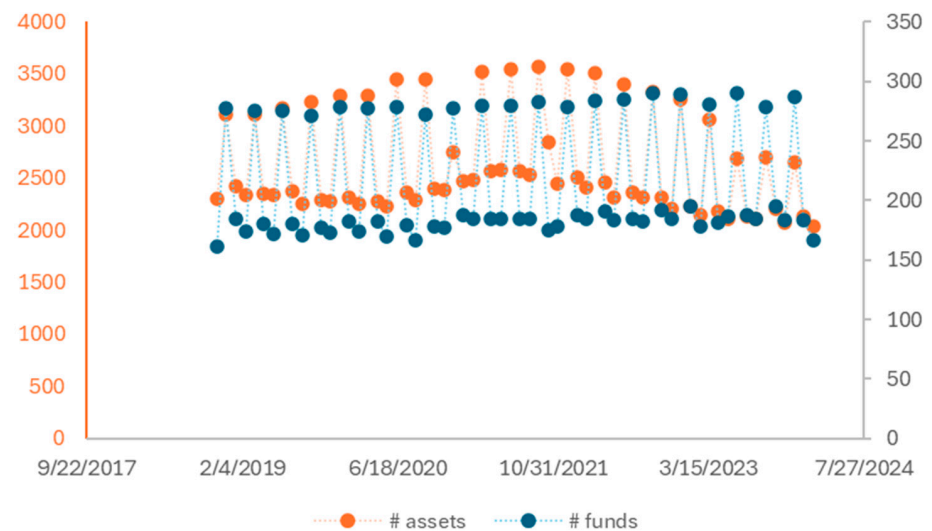
We exclude (given that we are trying to analyze the commonality of holdings across different funds) index funds, which, by design, hold the same position if they are tracking the same index. We also exclude industry-specific funds, such as real estate, precious metal, utility funds, and other similar funds. Finally, we limit ourselves to domestic funds (i.e., excluding international funds) (The index funds are filtered based on the index flag in the dataset; others are not filtered). Our sample consists of 3874 funds, representing the field of domestic large-cap active equity funds.

We download the holdings of all the funds in our sample from WRDS. We have a total of 2201 portfolios and 12,208 assets (There are funds that contain non-stock assets, as the filtering threshold is set to $\geq 85\%$ equity).

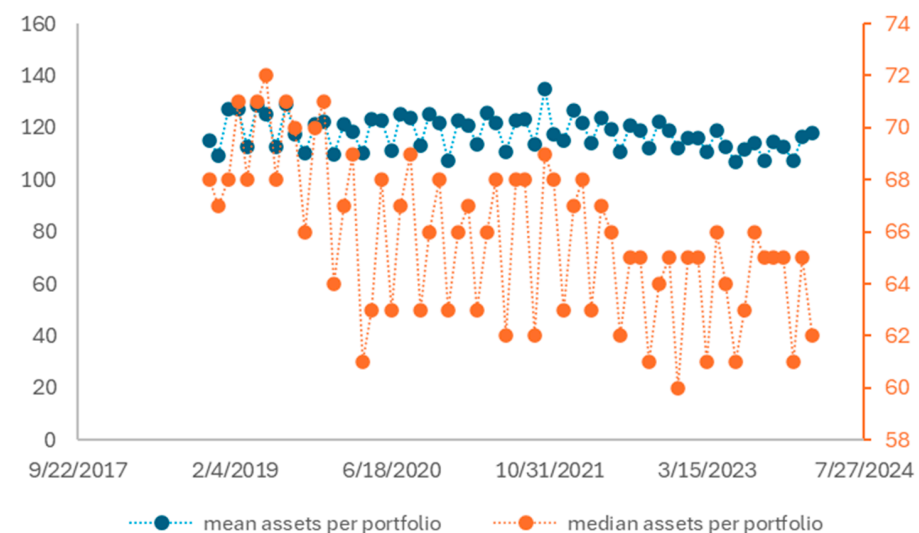
The number of funds and portfolios is summarized in Table 1 and plotted in Figure 3. Ignoring the first five months and the last month (very few observations), on average, we have 214 portfolios and 2573 assets monthly.

Table 1. Summary statistics of fund holding data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

(A) all					
(all)		64	fund		asset
number of months			64		64
sum			13,665		164,659
average			214		2573
(B) split into high and low					
(high)	fund	asset	(low)	fund	asset
number of months	46	46	number of months	18	18
sum	8630	107,817	sum	5035	56,842
average	188	2344	average	280	3158



(A) Raw data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets)



(B) Raw data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets)

Figure 3. Data summary. (A) Number of assets and number of funds over time. Note: The grand average number of funds is 213, and the average number of assets is 2622. (B) Average and median of assets per portfolio over time. Note: The average mean of the assets is 118, and the median of the assets is 66 per portfolio. This indicates that there are a few very large funds that dominate the market. “#” is “number”.

Figure 3 demonstrates that, except for the first five months and the last month, the number of funds and the number of assets are fairly stable over time. But they oscillate every few months. We suspect that this is due to the fact that the holding position data are not updated monthly for every fund. Some contain complete monthly updates, while others update quarterly, and some have gaps in certain months. Since the analysis is focusing on a cross-sectional snapshot, not a time series, we did not manually fill the gaps.

Clearly, we can see two distinctive levels. The higher level has an average of 280 funds and 3158 assets, while the lower level has an average of 188 funds and 2344 assets. In the next section, a bipartite graph is calibrated to the data. Also, we perform v-measure with k-means clustering to classify funds/assets. We then use the results to test if our data follow a bipartite graph.

4.2. Testing Hypotheses

Following Satone, Desai, and Mehta [6], the primary objectives of this paper are twofold. First, we examine whether the fund–asset holdings data can be meaningfully represented as a bipartite graph. Second, conditional on such a representation, we assess whether the bipartite structure is consistent with the clustering analysis.

To guide the empirical analysis, we consider the following hypotheses, which are evaluated through observable patterns in the data rather than formal statistical tests.

Hypothesis 1 (Bipartite Structure): *If fund–asset relationships conform to a bipartite structure, embeddings derived from the bipartite graph should preserve a clear separation between fund nodes and asset nodes. Empirically, support for this hypothesis would be reflected in relatively high v-measure values and clustering outcomes in which fund and asset nodes form distinct groups.*

Hypothesis 2 (Embedding Consistency): *If the bipartite structure is preserved in the embedded space, clustering applied to node embeddings should yield only two well-separated clusters corresponding to funds and assets. A lack of such separation would indicate that the bipartite structure is not well preserved by the embedding.*

Hypothesis 3 (Economic Relevance of Similarity): *If graph-based embeddings capture economically meaningful similarities among funds, then similarity measures derived from embeddings should exhibit systematic relationships with observable fund characteristics, including return co-movement and risk-based metrics.*

It is important to note that these hypotheses are not formulated in the classical statistical inference framework. For graph-based statistics such as the v-measure, no known asymptotic sampling distribution exists, which precludes conventional hypothesis testing with controlled Type I error rates. Accordingly, the hypotheses are evaluated using comparative and robustness-based evidence, consistent with standard practice in the graph learning and machine learning literature.

In a standard statistical inference framework, a null hypothesis is specified and evaluated using a Type I error control. In such settings, rejection of the null hypothesis at a given confidence level is meaningful, whereas acceptance is less informative in the absence of power or Type II error. In contrast, the hypotheses examined here—following Satone, Desai, and Mehta [6]—are intended to assess whether empirical evidence supports the presence of bipartite structure, rather than to conduct formal hypothesis testing. While this approach lacks classical statistical rigor, it is common in machine learning and graph-based analyses, where test statistics often lack known sampling distributions. Consequently, inference typically relies on robustness checks and systematic exploration across a wide range of parameter configurations, as implemented in our empirical analysis.

In addition, scalability is an important consideration when evaluating graph-based methods. Although computational complexity in graph algorithms is difficult to characterize using simple polynomial order notation due to its dependence on graph topology, size, and sparsity, scalability can be assessed empirically by applying the method to large networks with many nodes and edges. Graph embedding methods such as node2vec are particularly attractive in this regard because they transform network structure into numerical vector representations, enabling efficient downstream analysis using standard machine learning tools. Compared to graph-based measures such as centrality or community detection—which assign discrete labels or scalar values to nodes and are less amenable to pairwise comparison—node2vec provides a flexible, continuous representation that facilitates similarity measurement and integration with conventional statistical and machine learning techniques.

4.3. Bipartite Graph

Given that at any given time (month), there are over 200 funds holding over 2500 assets, we can only select a small sample to demonstrate a bipartite graph in this study. Figure 4 contains a small sample from March 2020. The left is a set of funds (by their IDs), and the right is the assets they own. For example, fund #1023335 holds Chevron, Merck, Procter & Gamble, Johnson & Johnson, Exxon Mobil, JP Morgan Chase, United Health, Abbvie, and Broadcom. Hence, this is a well-diversified fund that invests in pharmaceuticals, energy, finance, health, and high-tech.

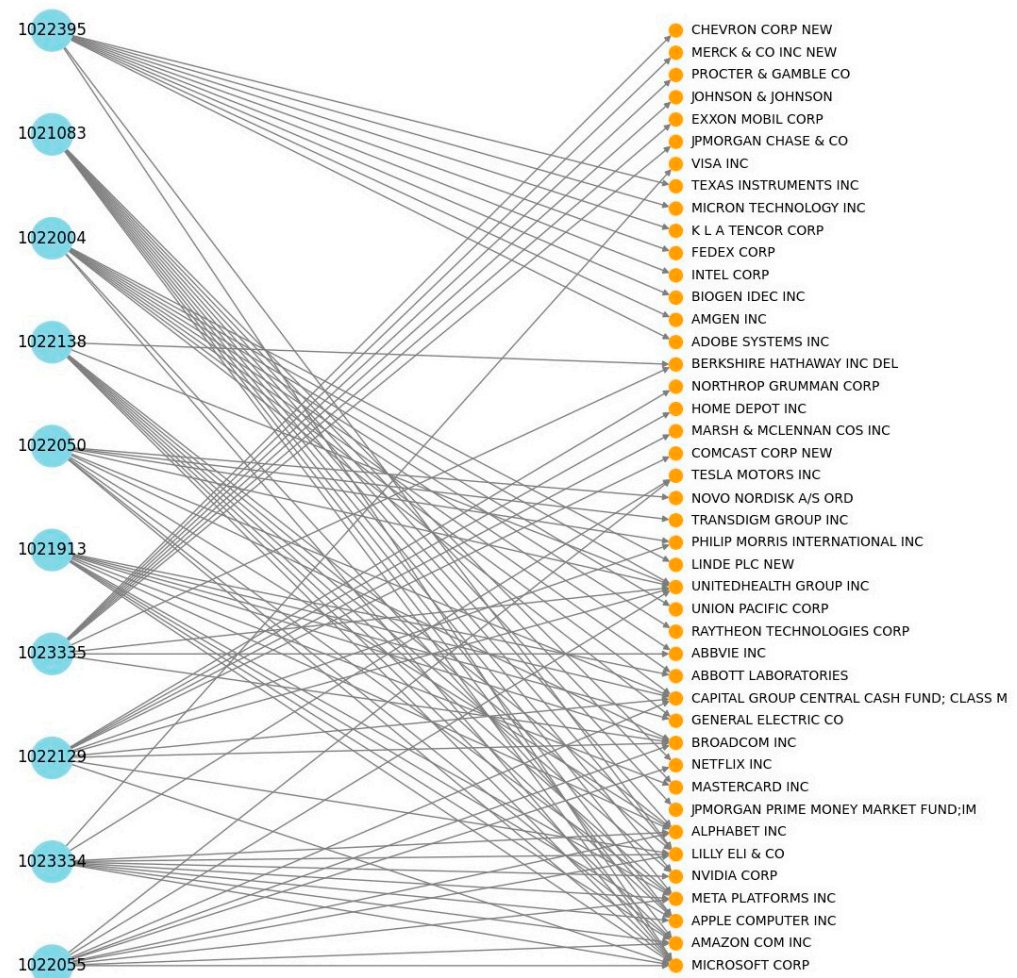


Figure 4. A bipartite graph of March 2020 with selected funds and assets.

We also see in Figure 4 that an asset is held by multiple funds. Microsoft, for example, is held by almost every fund.

Figure 4 illustrates a structure analogous to that shown in Figure 1. Funds form one set of nodes and assets form the other, with no connections within each set. Edges exist only between funds and assets, indicating asset holdings by funds. Because holdings are directional—from funds to assets—the resulting bipartite graph is directed. However, as the focus of our analysis is on network structure and similarity rather than edge directionality, we treat the graph as undirected in the empirical analysis.

4.4. *v*-Measure Tests and *k*-Means Clustering

As described in Section 3.4, node2vec requires the specification of five hyperparameters: the embedding dimension d , the length of each random walk l , the number of random walks per node r , the probability of returning to previously visited node p , and the proba-

bility of entering an unexplored part of the network q . For a given set of hyperparameter values, node embeddings are constructed and evaluated using clustering-based diagnostics.

To assess whether bipartite structure is preserved in the embedded space, we evaluate clustering outcomes using the v-measure, with higher values indicating stronger alignment between clusters and the fund–asset partition. Because no principled criterion exists for selecting a single optimal hyperparameter configuration in this setting, we conduct a systematic search over a range of commonly used parameter values and identify configurations that maximize the v-measure.

For each embedding specification, k-means clustering is applied to the embedded nodes, and the number of clusters is selected to maximize the corresponding v-measure. The resulting v-measure values across representative parameter combinations are reported in Table 2.

Table 2. v-measure under various settings. (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

d	l	r	p	q	Opt k	v-Measure
8	128	128	0.1	5	8.6	0.0494
8	128	128	5	0.1	8.8	0.1053
8	64	64	0.1	5	8.7	0.0317
8	64	64	5	0.1	8.5	0.0774
16	128	128	0.1	5	8.5	0.0848
16	128	128	5	0.1	8.5	0.1353
16	64	64	0.1	5	8.5	0.0559
16	64	64	5	0.1	8.3	0.0925
32	128	128	0.1	5	8.4	0.1284
32	128	128	5	0.1	8.5	0.1591
32	64	64	0.1	5	8.2	0.0851
32	64	64	5	0.1	8.5	0.1092

Table 2 summarizes representative results obtained under different node2vec parameter configurations. Specifically, we vary the embedding dimension $d \in \{8, 16, 32\}$, walk length $l \in \{64, 128\}$, number of walks $r \in \{64, 128\}$, return parameter $p \in \{0.1, 0.5\}$, and in–out parameter $q \in \{0.1, 0.5\}$. This yields a total of 48 parameter combinations for each month. For brevity, Table 2 reports a subset of 12 representative cases.

The final two columns of Table 2 report, for each parameter configuration, the number of clusters k that maximizes the v-measure, along with the corresponding average v-measure computed over time. Across specifications, the optimal number of clusters is relatively stable, with an average k of approximately 8.5. In contrast, v-measure values remain uniformly low, ranging from 0.0317 under the weakest-performing specification to 0.1591 under the strongest-performing specification. These findings indicate that, unlike index funds, active mutual funds do not exhibit a bipartite structure that is cleanly preserved by node2vec embeddings.

Figure 5 presents time-series plots of v-measure values for two representative parameter configurations. The left panel corresponds to the specification yielding the highest average v-measure (0.1591), while the right panel corresponds to the second-highest specification (0.1335). Although v-measures exhibit moderate temporal variation—ranging from below 0.1 to approximately 0.25—they remain low throughout the sample period. Moreover, the second-best specification exhibits substantially greater variability than the best-performing configuration.

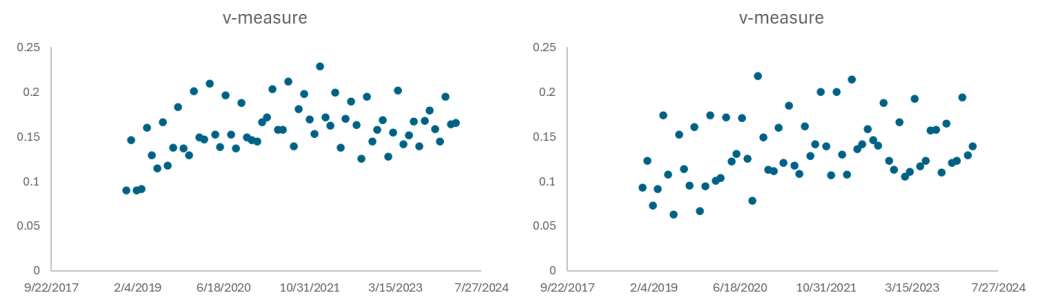


Figure 5. Time series plots of v-measure. The left panel assumes the following parameter values: $d = 32$; $l = 128$; $r = 128$; $p = 5$; and $q = 0.1$; while the right panel assumes the following parameter values: $d = 16$; $l = 128$; $r = 128$; $p = 5$; $q = 0.1$; $v = 0.1353$. Raw data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

For the remainder of the analysis, we adopt the parameter configuration that maximizes the v-measure. This choice ensures that subsequent results are evaluated under the most favorable conditions for preserving bipartite structure in the embedded space. The values used are $d = 16$, $\ell = 128$, $r = 128$, $p = 0.1$, and $q = 0.5$.

4.5. Clustering Results

Low v-measures indicate that our funds and assets do not exhibit a clear bipartite graph. To visualize this result, we plot the December 2019 data (funds and assets) in a two-dimensional diagram under selected parameter settings, as shown in Figure 6. Figure 6 presents a series of k -means clustering results for funds and assets under different parameter configurations, which are reported at the top of each panel.

The right panels of Figure 6 present two-dimensional cluster visualizations obtained via t-SNE. The number of clusters is set to nine (zero~eight). As argued in Satone, Desai, and Mehta [6], this visualization provides a test of bipartiteness. If the vertices exhibit a clear bipartite structure, clustering should yield two distinct groups corresponding to funds and assets. However, none of the plots display such a two-cluster separation. This observation is consistent with the result of the previous section, in which v-measures are uniformly low.

Following Satone, Desai, and Mehta [6], we re-color the same embeddings using two labels, for funds and assets, and display them in the left panels of Figure 6. The left and right panels are identical in terms of node locations. The orange dots represent funds, and the blue dots represent assets. If bipartiteness holds, funds and assets will occupy distinct regions of the embedding space (e.g., funds in one quadrant and assets in another), or at a minimum, the two groups will be clearly separable.

This pattern is not observed in Figure 6. For example, Panel (A) clearly demonstrates that the orange dots (funds) are mingled with the blue dots (assets), corresponding to a low v-measure of 0.0638. Even in Panel D, where the v-measure is the highest at 0.1564, the orange dots are clustered together, although they are surrounded by blue dots. While they are largely separated, they are not positioned in two different areas (e.g., upper right vs. lower left).

Our results indicate that bipartiteness is not preserved in the embedding space. This is because we eliminate index funds and include broader types of funds. Index funds, by construction, hold highly overlapping portfolios, which mechanically yield high v-measures and apparent bipartiteness. In contrast, active funds—where portfolio overlap is less pronounced—do not exhibit a clear bipartite pattern. Accordingly, our findings suggest that the bipartiteness documented in Satone, Desai, and Mehta [6] is primarily driven by the inclusion of index funds rather than by a general property of mutual fund holdings networks.

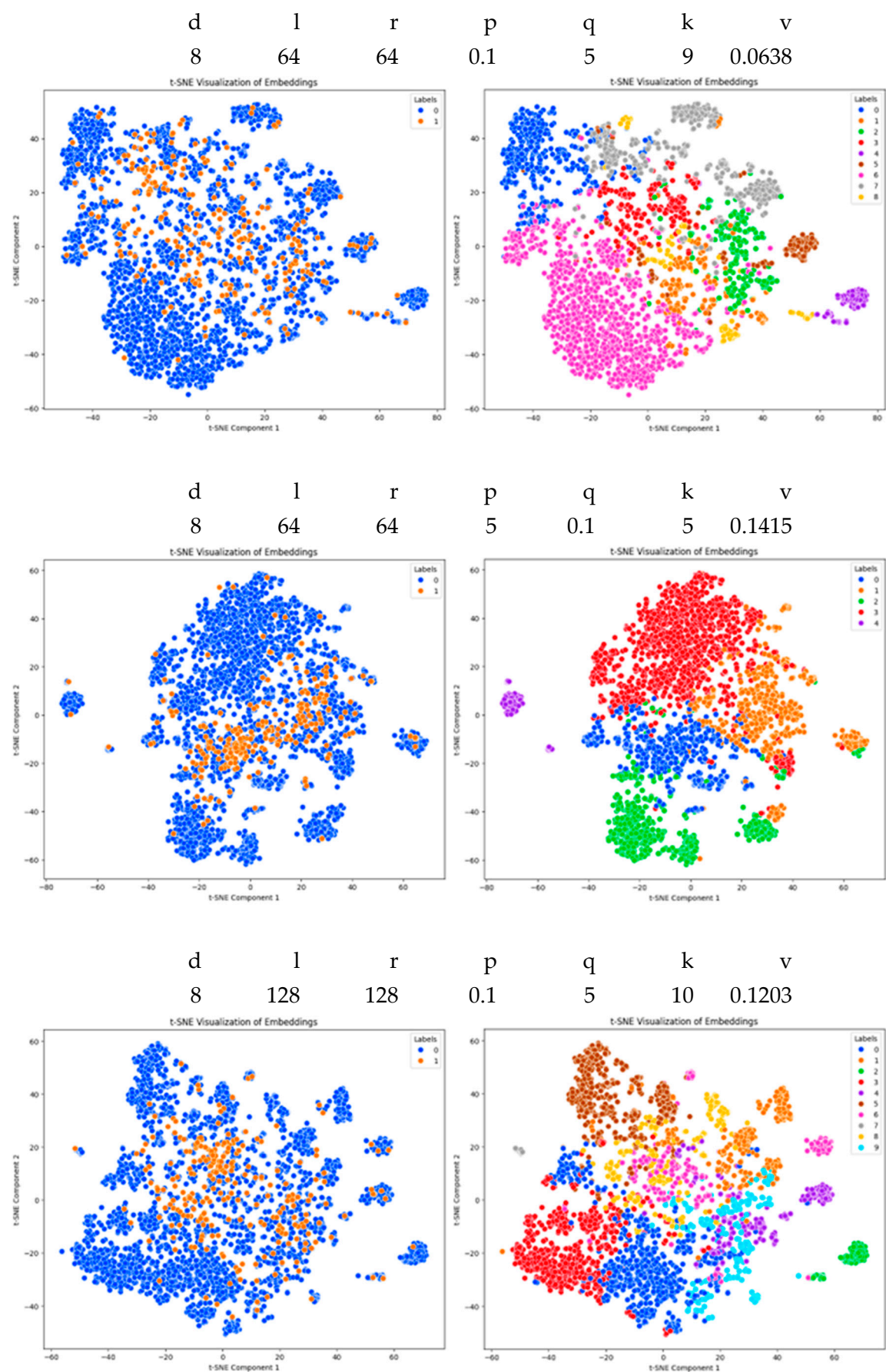


Figure 6. Cont.

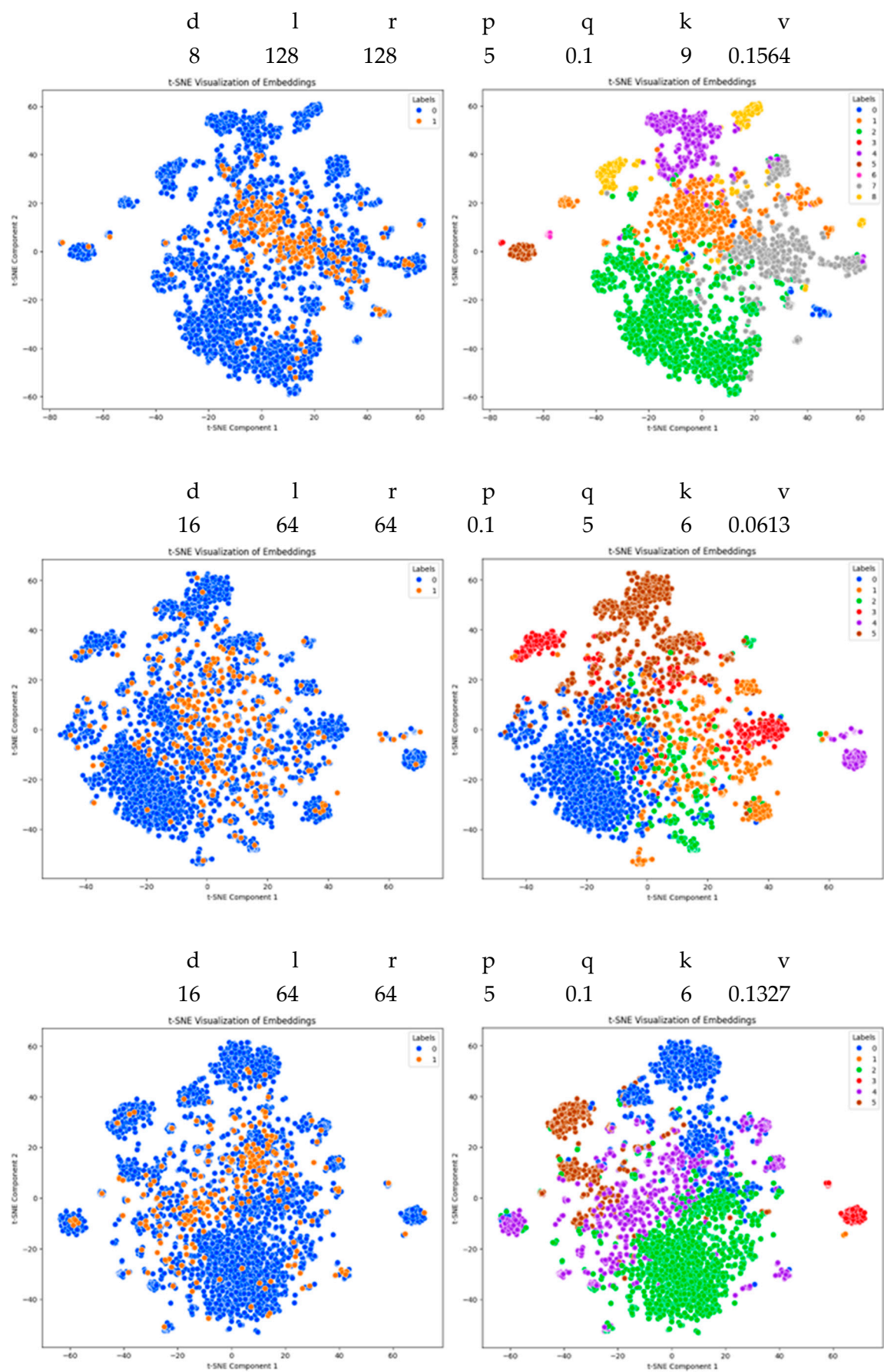


Figure 6. Cont.

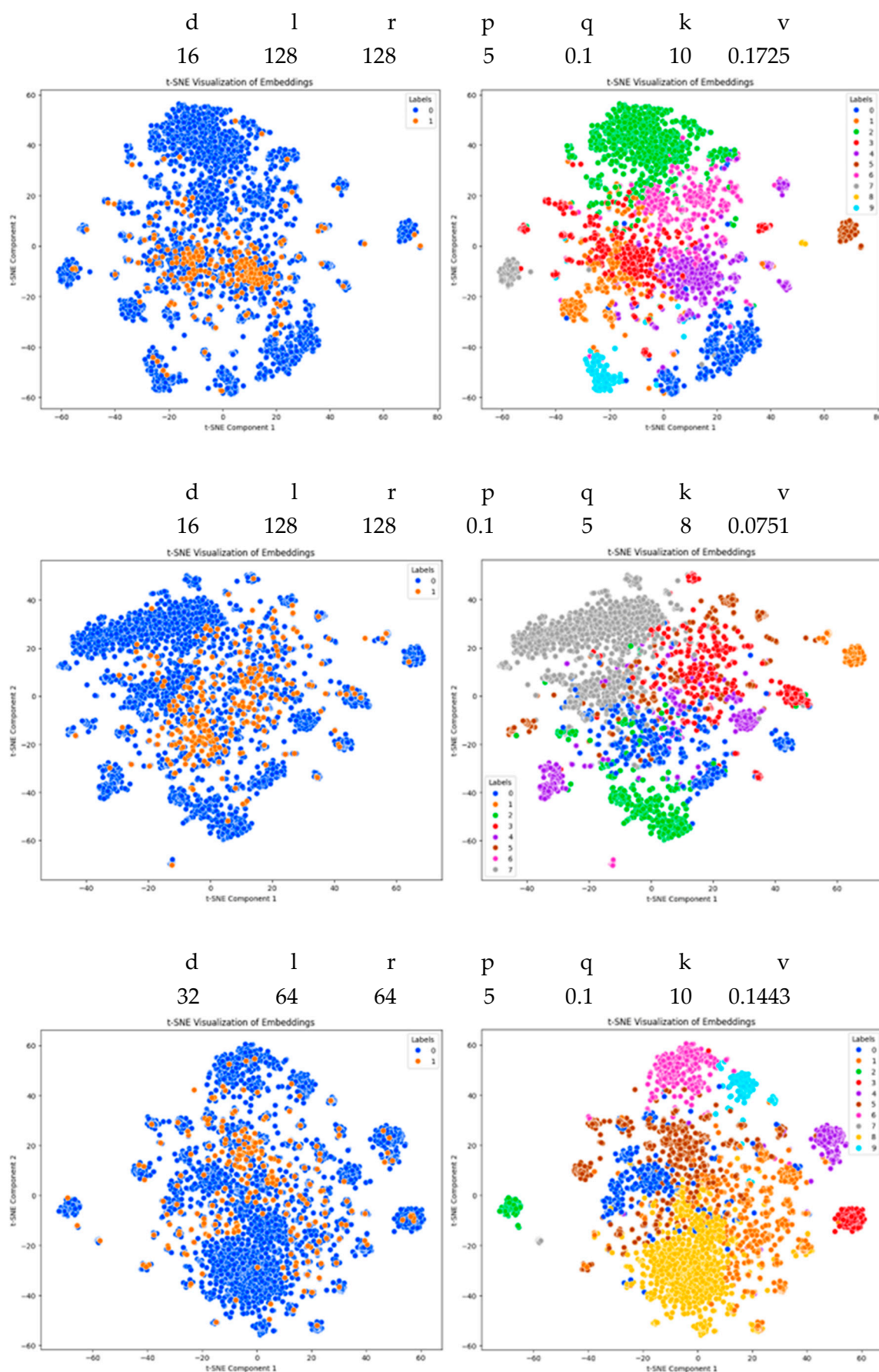


Figure 6. Cont.

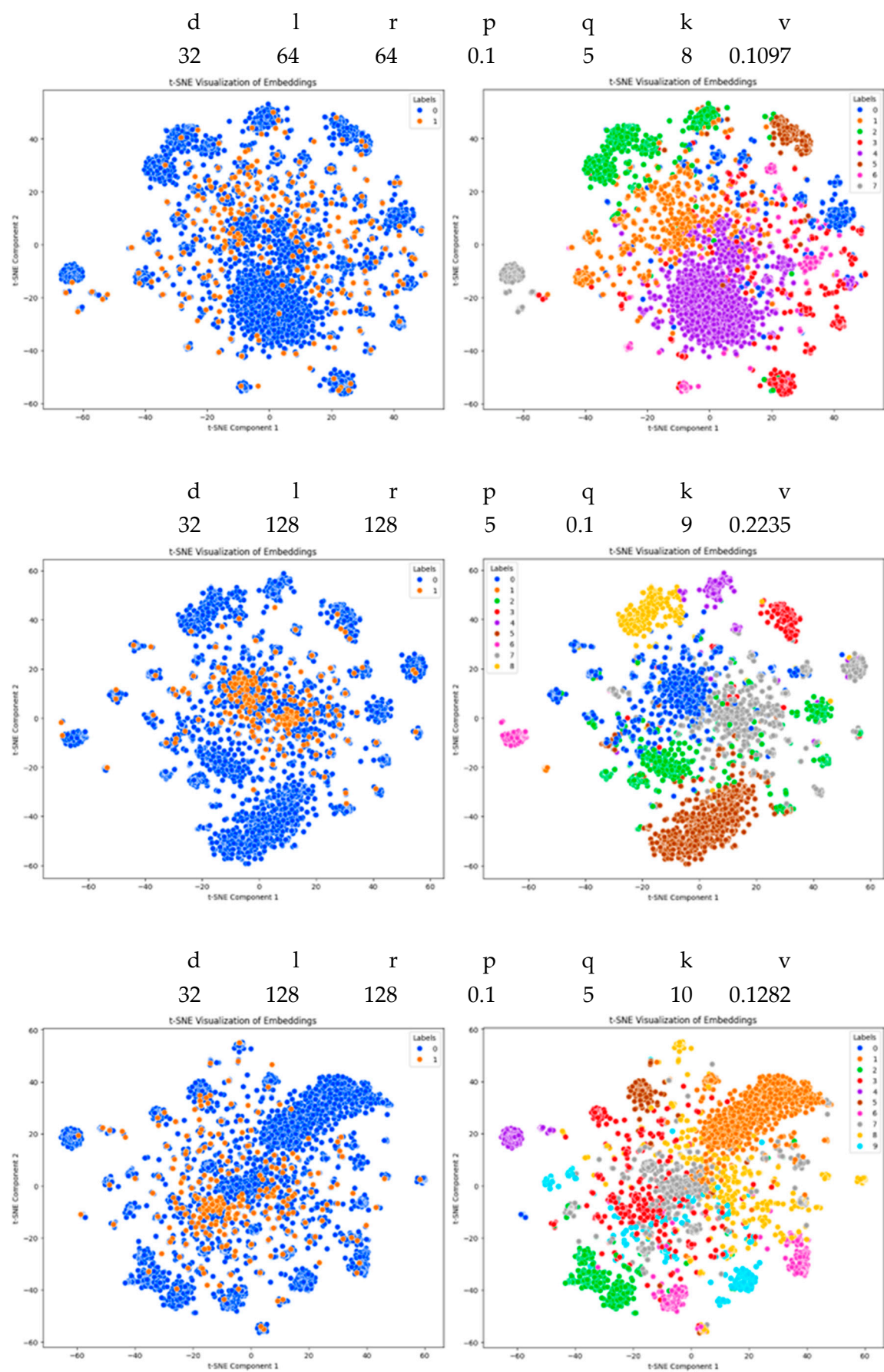


Figure 6. Similarity graphs under various parameter values (December 2019, 279 funds).

4.6. Similarity Results

We present a series of empirical results comparing fund similarity calculated from holdings (raw data) with those obtained from graph-based embeddings generated by node2vec.

The key innovation in applying bipartite graphs to fund similarity is the use of node2vec, which leverages embedding techniques originally developed for textual data to represent network structure. This approach provides a new perspective on fund similarity relative to the traditional measures based on asset holdings.

Graph embeddings capture similarities between funds through their positions in the fund–asset network, rather than through direct overlap in asset holdings. In this section, we conduct a comparative evaluation to assess whether similarity measures based on bipartite graph embeddings offer incremental explanatory power relative to similarity measures computed from raw holdings data.

Specifically, for each month, we compute a cosine similarity score matrix (similar to a correlation matrix) of every fund with others using both portfolio weights and node2vec embeddings. For example, in November 2018, there were 161 funds (holding 2307 assets). Each fund is represented by a vector of portfolio weights (and a vector embedding). We compute two 161×161 cosine similarity score matrices based on embeddings and holdings, separately. As in a correlation matrix, diagonal elements are equal to one. Figure 7 illustrates these similarity matrices.

Similarity (correlation) Matrix

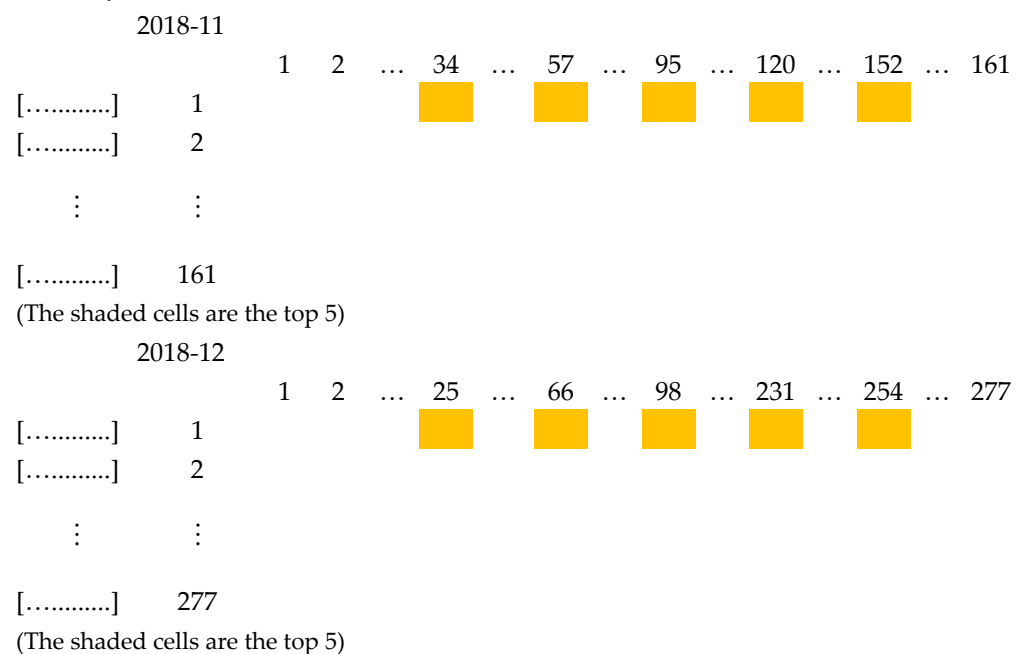


Figure 7. Similarity graphs (November and December of 2018, 161 and 277 funds, respectively). The top 5 most similar funds from fund #1 are highlighted in the figure. For example, in November of 2018 (top panel), the top 5 most similar funds to fund #1 were funds #34, #57, #95, #120, and #152 and in December of 2018 (bottom panel) the top 5 most similar funds to fund #1 were #25, #66, #98, #231, and #254.

This procedure allows us to identify the most similar funds of any given fund (i.e., each row). For example, in Figure 7, excluding itself, fund #1 has 160 similarity scores with other funds, and its 5 highest scores are with #34, #57, #95, #120, and #152. More generally, similarity can be defined using the top n closest funds. In our empirical analysis,

we consider $n = 5, 10, 15, 20, 30$, and 50 . To conserve space, we report the results for the top 5 and top 20 cases; additional results are available upon request.

We repeat this exercise for every month in our data. For example, in December 2018 (Panel B), there were 277 funds, yielding a 277×277 similarity matrix. The top five most similar funds for fund #1 are #25, #66, #98, #231, and #254.

We then run a series of tests. The first test uses the Jaccard index to compare similarity rankings obtained from holdings-based and embedding-based measures. As introduced earlier, a Jaccard index is the ratio of the intersection to the union. If the two sets have no overlap (i.e., disjoint sets), the index equals zero. On the other extreme, if the two sets are identical, the Jaccard index is equal to one.

To compute the Jaccard index for a given fund, we first rank all other funds according to similarity scores obtained from holdings and from embeddings. The Jaccard index then measures the degree of overlap between the two resulting rankings. A value of one indicates that the two methods yield identical similarity rankings, while a value of zero indicates no overlap. Consider November 2018, when the sample consisted of 161 funds. For a given fund, say A, suppose the top five most similar funds are {B, G, Q, X, Z}, while the top five most similar funds based on holdings are {C, F, Q, R, Z}. The intersection of these sets is {Q, Z}, and their union is {B, C, F, G, Q, R, X, Z}, yielding a Jaccard index of $2 \div 8 = \frac{1}{4}$.

For each month, we compute the Jaccard index of the top n most similar funds for every fund. We then take an average of all the funds in that month (e.g., 161 funds in November 2018). We repeat this calculation every month. Note that if we select the top n where n is equal to the total number of funds minus 1, then the Jaccard index is 100% since the intersection is identical to the union. In addition to the top 5, we also compute results for the top 10, 15, 20, 30, and 50. To conserve space, we only report the results of the top 5 (left panel) and top 20 (right panel) in Figure 8.

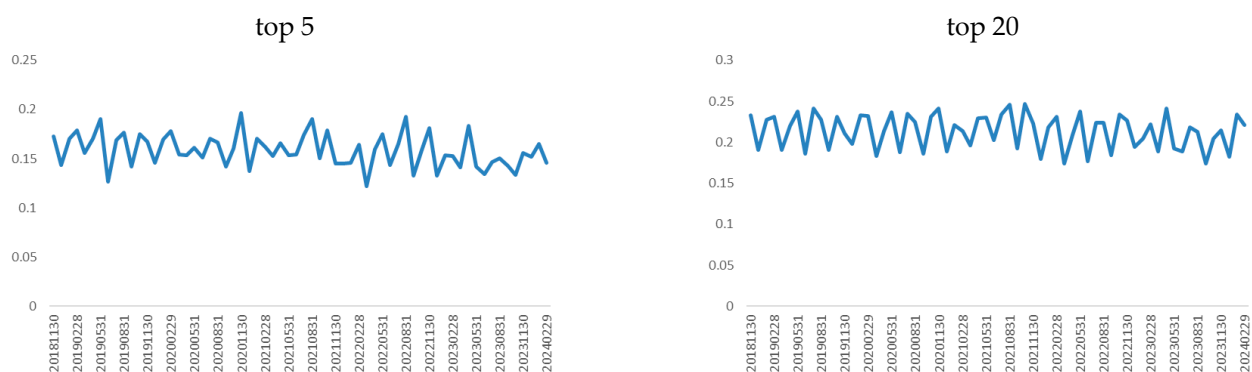


Figure 8. Time series of Jaccard Index. All data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

As shown in Figure 8, the Jaccard index remains stable over time for both specifications, with overall average values of 15.84% for the top 5 and 21.33% for the top 20. These small index values indicate that holdings-based and embedding-based approaches produce substantially different assessments of fund similarity.

Next, we study fund similarity by directly comparing the two similarity matrices. To illustrate, we select two months—November 2018 and February 2024—and plot their heat maps in Figure 9. The patterns observed for these months are representative of those in other periods, which are available upon request.

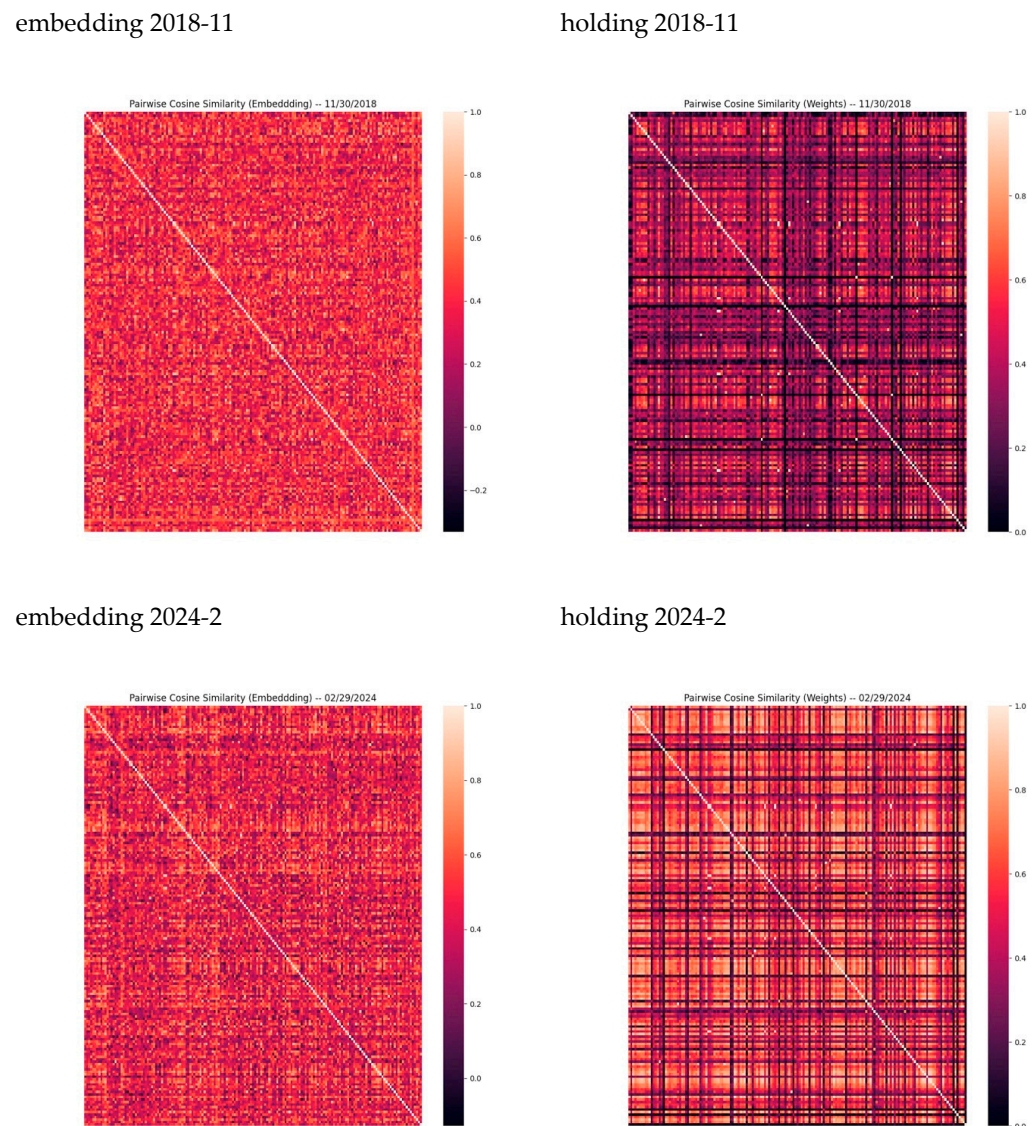


Figure 9. Heat maps under embedding (**left** graphs) and holding (**right** graphs) for two selected dates: November 2018 and February 2024.

The similarity matrices indicate that embedding-based measures generally yield higher similarity scores than those based on holdings. This suggests that graph-based representations capture broader relational similarities among funds than direct portfolio overlap alone.

To further investigate these similarity scores, we examine the distribution (histogram) of these similarity scores. Using November 2018 as an illustrative example, the sample includes 161 funds, yielding 160 pairwise similarity scores for each fund. For each fund, we compute the average of its pairwise similarity scores (i.e., the row-wise mean of the similarity matrix shown in Figure 8). We then plot the distribution of these averaged similarity scores in Figure 10, with the left panel corresponding to embedding-based similarity and the right panel corresponding to holdings-based similarity.

In addition to the distribution based on all pairwise similarities (Panel A), we also construct a distribution using the average of the five highest similarity scores for each fund (“top 5”), shown in Panel B. The distributions based on all similarities differ markedly across the two methods. The holding-based distribution is highly skewed, whereas the embedding-based distribution is more symmetric and exhibits a fat right tail. The mean similarity is 40.14% for embedding and 32.50% for holding, with corresponding medians of

40.47% and 33.88% and standard deviations of 16.46% versus 15.43%, respectively. Overall, the similarity is stronger with embedding than with holding. Comparing medians and means further suggests that the embedding-based distribution is approximately symmetric with mild right skewness, while the holdings-based distribution is strongly left-skewed.

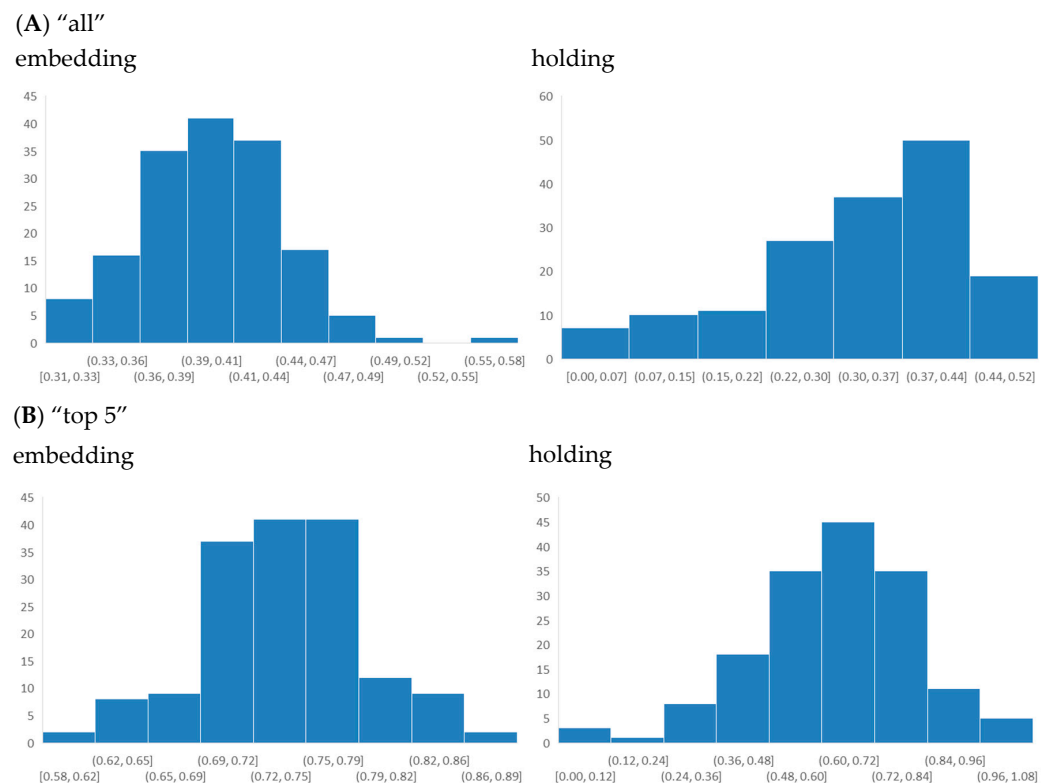


Figure 10. Distribution of mean similarity scores (November 2018, 161 funds).

Differences between the two methods are even more pronounced in the "top five" case. The embedding-based distribution has a mean of 74.19% and a median of 74.77%, indicating a nearly symmetric distribution, whereas the holdings-based distribution has a lower mean of 62.01% and a median of 63.95%, reflecting stronger left skewness. In contrast to the results for all similarities, the embedding-based distribution exhibits a smaller standard deviation (4.22%) than the holdings-based distribution (5.35%), indicating greater concentration among the most similar peers.

To formally assess whether the two distributions are similar, we perform the Kolmogorov–Smirnov (KS) test, a non-parametric test of equality between two cumulative probability functions (see the Appendix B for the description of the test statistic). For the "All" sample, the KS statistic is 0.4161, exceeding the critical values of 0.1814 at the 1% and 0.1513 at the 5% significance. Hence, we reject the null hypothesis that the two distributions are similar. For the "top five" distributions, the KS statistic is 0.4907, which also rejects the null hypothesis at both 1% and 5% significance levels.

Although Figure 10 only presents "all" and the top five cases, we also examine similarity based on the top 10, 15, 20, 30, and 50 most similar funds. These results generally fall between the two reported cases and are therefore omitted to conserve space. They are available upon request.

Figure 10 illustrates results for a single month. To assess the robustness of these findings over time, we extend the analysis to the full sample period of 64 months (from November 2018 to February 2024). Figure 11 plots the time series of the mean similarity values for the two distributions, with Panel A corresponding to all pairwise similarities and Panel B corresponding to the top five case.

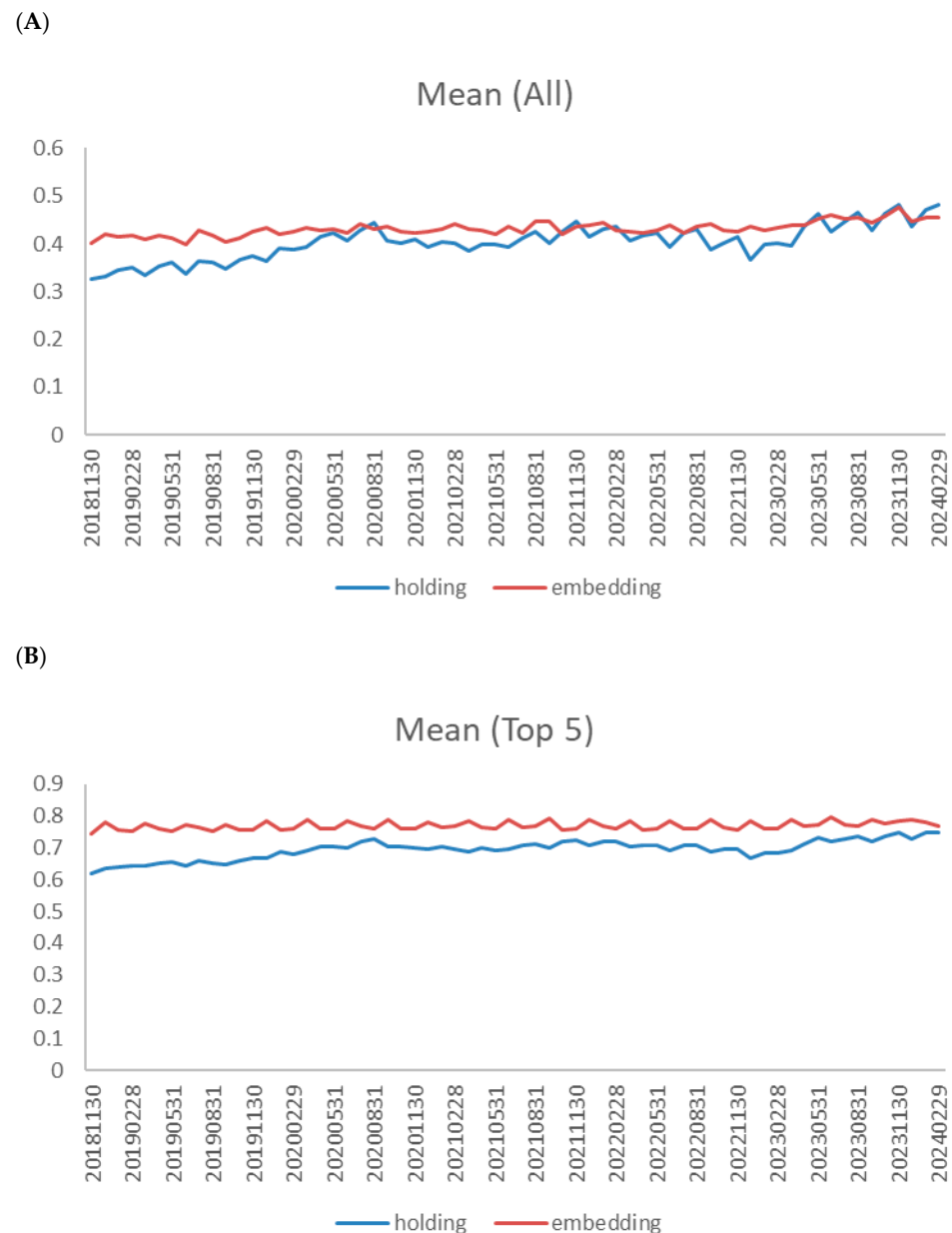


Figure 11. Mean similarity scores over time. All data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

In both panels, embedding-based similarity is higher than holding-based similarity, indicating that the graph-based method captures a broader aspect of fund similarity. However, this advantage of using graphs diminishes over time. We can see that the trend of the similarity scores of the holding distribution is upwards and becomes close to the embedding distribution (in the case of the top five, it even surpasses the embedding distribution).

Figure 12 presents the standard deviations of the two distributions for both “all” and the top five. For “all”, the two distributions are both symmetrical (16.46% mean versus 16.35% median for the embedding distribution and 15.43% mean versus 16.38% median for the holding distribution).

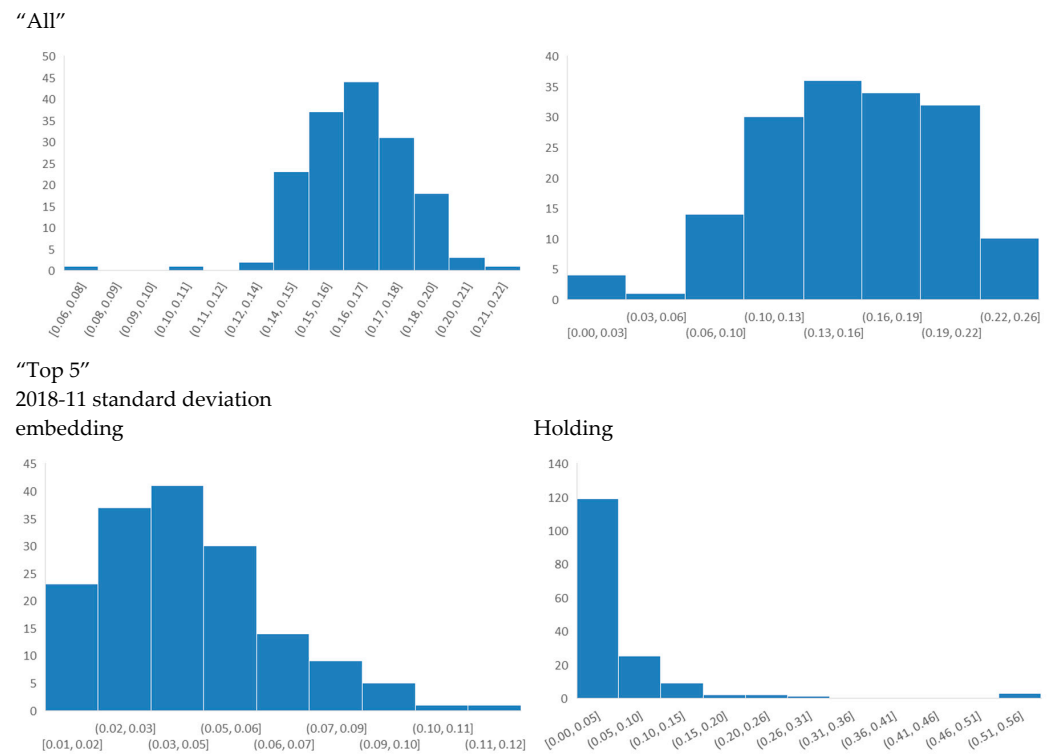


Figure 12. Distribution of standard deviations of similarity scores (November 2018, 161 funds).

The top five are quite different. Although both distributions are right-skewed, the holding distribution is much more skewed than the embedding distribution (4.22% mean versus 3.83% median for the embedding distribution and 5.35% mean versus 3.19% median for the holding distribution).

This indicates that the similarity measures based on the holding data are less reliable than those derived from embedding. For this reason, graph-based representations provide additional insight into how funds are interconnected beyond direct asset overlap. Similarly to Figure 11, Figure 13 presents the time series of the standard deviations of both distributions.

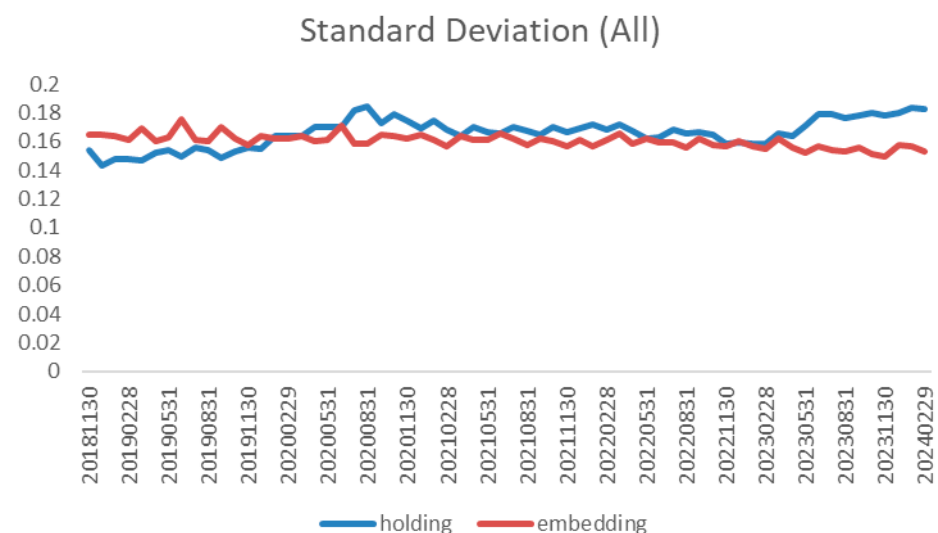


Figure 13. Standard deviation over time. All data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

The final test is to examine the relationship between similarity and fund performance. If similarity measures capture economically meaningful relationships, two similar funds should perform similarly as well. We therefore assess whether performance co-movement is more closely associated with holdings-based or embedding-based similarity.

Focusing on the top five case, we identify, for each fund, its five most similar peers based on holdings. We then compute the pairwise correlations of monthly returns between the fund and each of its five peers and take the average of these correlations. For example, for November 2018, this procedure yields 161 fund-level average correlations. We repeat this calculation for each month over the 64-month sample period. The resulting time series is shown in Figure 14.



Figure 14. Average correlation between returns of the chosen fund and other funds. All data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

Panel A of Figure 14 reports the average correlation over time between each fund and its top 5 most similar funds, while Panel B reports the average correlation for the top 50 peers. The overall average (across time) of the return correlation in the top 50 case based on holdings is 94.02%, which is notably higher than 92.74% for the corresponding embedding-based correlation. For the top 5 case, the correlations are 96.58% and 94.74%, respectively.

This result indicates that returns of funds clearly reflect the returns of the assets they own (i.e., holdings). Embeddings, although they still provide high correlations with returns, are less directly tied to returns than holdings-based measures. This is expected, as fund

returns are simply the weighted averages of the returns of the assets they hold; hence, Figure 14 is simply a tautological phenomenon.

To avoid tautology, we also compute correlations between the similarity and standard deviations of returns. While standard deviations are non-linear transformations of returns and hence alleviate tautology, the source of the dynamics is still the same—returns. Nevertheless, this analysis provides further insight into two similarity measures. The result is plotted in Figure 15.



Figure 15. Average Correlation between standard deviations of the chosen fund and other funds (top 50). All data (June 2018 until March 2024, 3874 funds, 2201 portfolios, and 12,208 assets).

The top graph of Figure 15 plots the average correlation over time between a given fund and its top 50 most similar funds, where similarity is defined using either holdings-based or embedding-based measures. As in Figure 14, correlations are computed pairwise and then averaged across peers for each fund on a monthly basis. The bottom panel of Figure 15 reports analogous results for the top five most similar funds. In contrast to the return-based results, the two correlation series are no longer closely aligned. Holdings-based correlations remain higher on average, whereas embedding-based correlations exhibit greater temporal

variation. Finally, there is no material difference between the patterns observed for the top 50 and top 5 cases.

Taken together with the results from Figure 8 (Jaccard index), these findings indicate that embedding results present a very different composition of assets than that of holdings, yet they still present similar correlation results to those of holdings. This suggests that the bipartite graph preserves certain dimensions of similarity that are not fully reflected in the similarity based on asset holdings.

In summary, the two main results in this section, the similarity test and performance test, appear to point in different directions. This divergence arises because graph-based similarity incorporates information beyond portfolio weights, reflecting broader network relationships among funds. As a result, embedding-based measures tend to produce higher similarity scores. In contrast, fund performance is mechanically determined by portfolio holdings, making holding-based similarity more closely aligned with realized returns than embedding-based similarity measures. Accordingly, while graph embeddings capture structural relationships among funds, they do not dominate holdings-based measures in explaining return co-movement. These findings qualify the conclusions of Satone, Desai, and Mehta [6] by showing that the advantages of graph-based similarity measures do not generalize uniformly beyond settings dominated by index funds.

5. Conclusions

This paper builds on Satone, Desai, and Mehta [6] by studying fund similarity using bipartite graphs and node2vec. The use of graph-based representations is innovative in that it models relationships among funds through a network rather than relying solely on portfolio weights, as in the traditional finance approach.

The purpose of using graph embedding to replace portfolio weights (i.e., holding) is to determine whether embeddings provide a more informative representation of how funds are related. Our empirical results are mixed, however. Using monthly data from June 2018 to March 2024, we obtain three main results. First, we find no evidence supporting the preservation of a bipartite structure in the embedding space for active mutual funds. This contrasts with the findings of Satone, Desai, and Mehta [6], whose results are driven by index funds that, by construction, exhibit dense and overlapping holdings networks. When applied to a broader universe of active funds, bipartiteness is not sustained. Second, similarity measures derived from graph embeddings are substantially higher than those based on portfolio holdings, indicating that embeddings capture broader structural relationships among funds. Third, when similarity is evaluated against fund performance, holdings-based measures explain return co-movement more effectively than embedding-based measures, reflecting the mechanical link between portfolio weights and realized returns.

These results raise an important question regarding the objective of fund similarity analysis. If performance is the only criterion for analyzing fund similarity, then portfolio holdings provide a sufficient tool. However, investors and regulators are concerned with dimensions beyond average returns. For example, various asset pricing theories balance returns with different risks, such as (1) the systemic risk, which is highly related to networks (i.e., graphs), as discovered by Chen and Zhang (2024) [8], (2) the liquidity risk, which caused the 2008 financial crisis, as discussed by Chen, He, and Lin (2016) [24], and (3) the interest risk, which directly impacts bank equities, shown by English, Heuvel, and Zakrajsek (2018) [25], and numerous others. We shall leave this for future research.

Author Contributions: Conceptualization, R.-R.C. and Y.W.; Methodology, R.-R.C. and L.L.; Software, L.L. and X.Z.; Validation, L.L. and X.Z.; Formal analysis, Y.W.; Writing—review & editing, R.-R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available in WRDS at <https://wrds-www.wharton.upenn.edu>, accessed on 5 January 2026.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. node2vec

node2vec is an approach to retrieving information embedded in graphs. node2vec turns a graph into a set of embeddings (quantified), which can then be used for various analyses. The idea of node2vec is to take a graph and create a corpus by walking (randomly) through the vertices. Plainly put, by walking from one vertex to any other vertex, we obtain a sentence-like string of words (where each vertex is like a word). Then, we can apply NLP techniques to analyze the vertex-based corpus. Most importantly, embeddings can be created, which enables us to take advantage of many powerful models in textual learning models (such as Transformer).

In a typical textual analysis, we face the following example (Source: <https://medium.com/nlplanet/two-minutes-nlp-using-word2vec-to-learn-node-embeddings-on-graphs-ebece432a4b6>, accessed on 4 January 2026.):

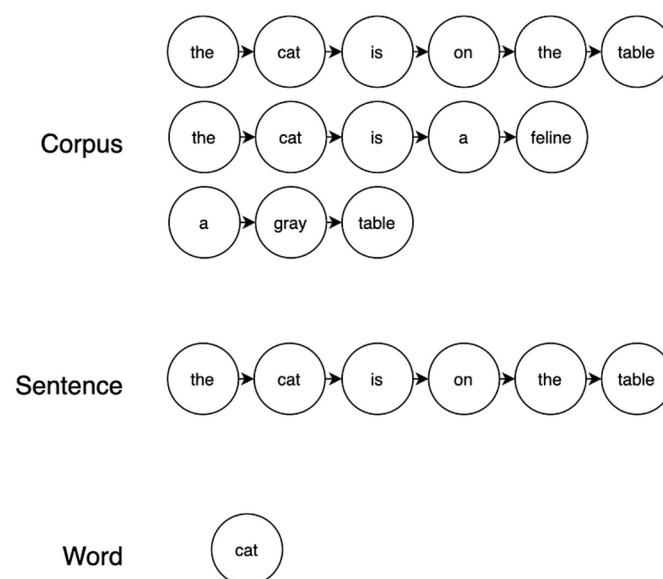
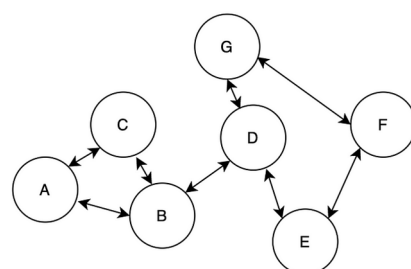
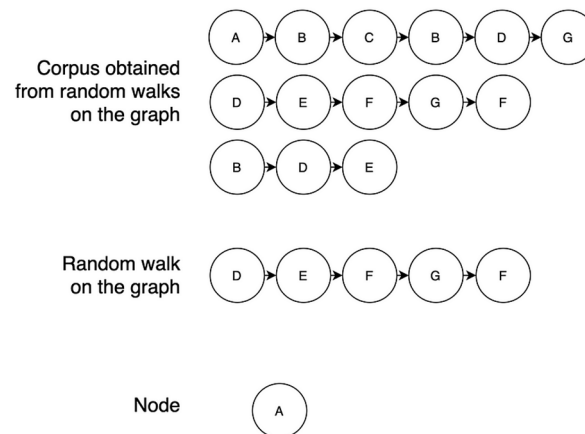


Figure A1. A Demonstration of a word, a sentence, and a corpus in a graph.

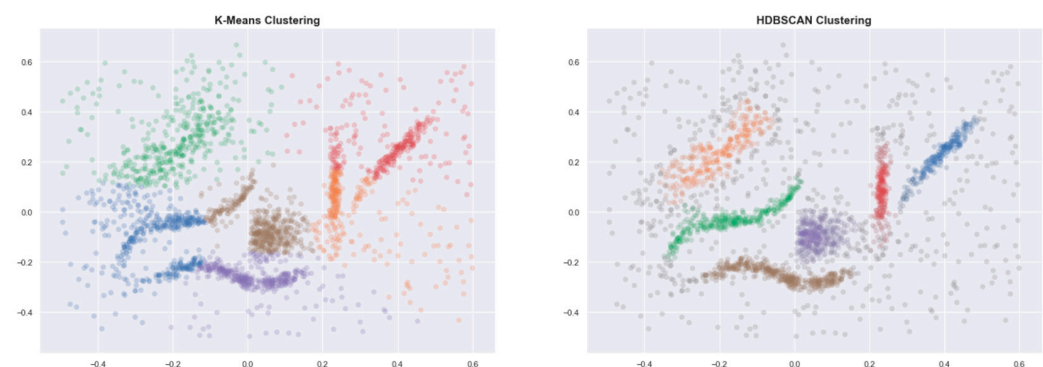
We can then use word2vec or doc2vec to create embeddings, which are vectors of numbers. Now, take the following graph as an example,



in which locations and distances between any two vertices carry no information regarding the relationship. To perform node2vec, we recognize in the above graph that each vertex is a word. To create a “sentence” which is a collection of “words”, we need to string together a set of vertices. In order to accomplish this, we need to create a random walk in the network graph above. For example, we can randomly walk a path, like $A \rightarrow B \rightarrow C \rightarrow B \rightarrow D \rightarrow G$. This is a sentence. We can walk again, randomly: $D \rightarrow E \rightarrow F \rightarrow G \rightarrow F$. Then we can collect a number of such sentences to form a document or even a corpus as follows:



Then we can use word2vec or doc2vec to create embeddings. Once embeddings are available, various similarity metrics can be used to study the relationships of the vertices (<https://pberba.github.io/stats/2020/01/17/hdbscan/>, accessed on 5 January 2026).



Appendix B. Kolmogorov–Smirnov Test

The Kolmogorov–Smirnov is a non-parametric test that may be used to test whether two underlying one-dimensional probability distributions differ. The Kolmogorov–Smirnov statistic quantifies a distance between the empirical distribution function of the sample and the cumulative distribution function of the reference distribution, or between the empirical distribution functions of two samples. The KS test statistic is the difference between two cumulative probabilities as follows:

$$k > c(\alpha) \sqrt{\frac{n_1 + n_2}{n_1 n_2}}$$

where n_1 and n_2 are two sample sizes and α is the significance level. The table for $c(\alpha)$ is given below:

Table A1. Kolmogorov-Smirnov Test Statistic.

α	0.2	0.15	0.1	0.05	0.025	0.01	0.005	0.001
$c(\alpha)$	1.073	1.138	1.224	1.358	1.48	1.628	1.731	1.949

References

1. Sensoy, B.A. Performance Evaluation and Self-designated Benchmark Indexes in the Mutual Fund Industry. *J. Financ. Econ.* **2009**, *92*, 25–39. [\[CrossRef\]](#)
2. Sakakibara, T.; Matsui, T.; Mutoh, A.; Inuzuka, N. Clustering Mutual Funds Based on Investment Similarity. *Procedia Comput. Sci.* **2015**, *60*, 881–890. [\[CrossRef\]](#)
3. Nanda, V.; Kelsey, W. *Correlated Flows, Portfolio Similarity and Mutual Fund Liquidity Management*; University of Texas at Dallas: Richardson, TX, USA, 2018; SSRN3119770.
4. Delpini, D.; Battiston, S.; Caldarelli, G.; Riccaboni, M. *The Network of U.S. Mutual Fund Investments: Diversification, Similarity and Fragility throughout the Global Financial Crisis*; University of Sassari: Sassari, Italy, 2018.
5. Wahal, S.; Wang, A. Competition among Mutual Funds. *J. Financial Econ.* **2011**, *99*, 40–59. [\[CrossRef\]](#)
6. Satone, V.; Dhruv, D.; Dhagash, M. Fund2Vec: Mutual Funds Similarity using Graph Learning. In Proceedings of the CAIF '21: Second ACM International Conference on AI in Finance, Online, 3–5 November 2021; pp. 1–8.
7. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. In Proceedings of the KDD '16: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
8. Chen, R.-R.; Zhang, X. From Liquidity Risk to Systemic Risk: A Use of Knowledge Graph. *Financ. Stab.* **2024**, *70*, 101195. [\[CrossRef\]](#)
9. Razafitombo, H. *Fund Analysis and Selection: An Approach Based on Distances and Similarities Between Performance Measures*; Maître de Conférences; CEREFIGE, ESM IAE-University of Lorraine: Metz, France, 2012.
10. McLemore, P.; Sias, R.; Wan, C.; Yüksel, H.Z. Active Technological Similarity and Mutual Fund Performance. *J. Financ. Quant. Anal.* **2022**, *57*, 1862–1884. [\[CrossRef\]](#)
11. Desai, D.; Dhiman, A.; Sharma, T.; Sharma, D.; Mehta, D.; Pasquali, S. Quantifying Outlierness of Funds from their Categories using Supervised Similarity. In Proceedings of the ICAIF '23: Fourth ACM International Conference on AI in Finance, Brooklyn, NY, USA, 27–29 November 2023; pp. 655–663.
12. Tanimoto, S.L.; Itai, A.; Rodeh, M. Some Matching Problems for Bipartite Graphs. *J. ACM* **1978**, *25*, 517–525. [\[CrossRef\]](#)
13. Maier, C.; Simovici, D. Bipartite Graphs and Recommendation Systems. *J. Adv. Inf. Technol.* **2022**, *13*, 249–258. [\[CrossRef\]](#)
14. Zhu, Z.; Su, J.; Kong, L. Measuring influence in online social network based on the user-content bipartite graph. *Comput. Hum. Behav.* **2015**, *52*, 184–189. [\[CrossRef\]](#)
15. Wang, L.; Wu, H.; Wang, W.; Chen, K.-C. Socially Enabled Wireless Networks: Resource Allocation via Bipartite Graph Matching. *IEEE Commun. Mag.* **2015**, *53*, 128–135. [\[CrossRef\]](#)
16. Fernandez, M.; Galeano, J.; Hidalgo, C. Bipartite Networks Provide New Insights on International Trade Markets. *Netw. Heterog. Media* **2012**, *7*, 399–413.
17. Cao, J.; Lin, X.; Guo, S.; Liu, L.; Liu, T.; Wang, B. Bipartite Graph Embedding via Mutual Information Maximization. In Proceedings of the Poster Session I WSDM '21, Virtual, 8–12 March 2021.
18. Anderson, P.; Farooq, B. A generalized partite-graph method for transportation data association. *Transp. Res. Part C Emerg. Technol.* **2017**, *76*, 150–169. [\[CrossRef\]](#)
19. Zhang, L.; Li, J.; Zhang, Q.; Meng, F.; Teng, W. Domain Knowledge-Based Link Prediction in Customer-Product Bipartite Graph for Product Recommendation. *Int. J. Inf. Technol. Decis. Mak.* **2019**, *18*, 311–338. [\[CrossRef\]](#)
20. Morris, S.A.; Gary, G.Y. *Construction of Bipartite and Unipartite Weighted Networks*; collections of journal papers; Oklahoma State University: Stillwater, OK, USA, 2018.
21. Zhang, L.; Zhao, M.; Zhao, D. Bipartite graph link prediction method with homogeneous nodes similarity for music recommendation. *Multimedia Tools Appl.* **2020**, *79*, 13197–13215. [\[CrossRef\]](#)
22. Manoochehri, H.E.; Nourani, M. Drug-target interaction prediction using semi-bipartite graph model and deep learning. *BMC Bioinform.* **2020**, *21*, 248. [\[CrossRef\]](#) [\[PubMed\]](#)
23. van der Maaten, L.; Geoffrey, H. Visualizing Data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
24. Chen, R.-R.; Lin, W.; Wei, H. A Liquidity Index. *J. Fixed Income* **2016**, *25*, 5–19. [\[CrossRef\]](#)
25. William, B.; Van den Heuvel, S.; Zakrajšek, E. Interest rate risk and bank equity valuations. *J. Monet. Econ.* **2018**, *98*, 80–97.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.