Visualizing Heterogeneous Graphs: from data to visualization

Author: Ramon Dijkstra*
University of Amsterdam

Supervisor: Deep Kayal†
Prosus

Examiner: Marcel Worring‡
University of Amsterdam

Figure 1: The visualization of our case study is an application intended for exploring relations between investors and companies. Pictured: On the top, the filter menu is shown. This menu can select the displayed nodes, the size, and the region of the specific investors and companies. In the middle, the graph shows all companies that are of medium size and in Europe. On the right, detailed information about the selected region can be found. Nodes can be hovered over to get a quick summary and regions can be selected to retrieve a summary on the right. Keywords on the top right can be clicked so that nodes containing the keyword in their description will pop up.

Abstract

In this paper, we will discuss how heterogeneous graph data can be visualized. To get from the data to the visualization, five steps have to be taken. First, the data must be pre-processed so that the interesting information is extracted. Then, the graph must be created by using the relations found in the data pre-processing step. This graph is then trained with the Heterogeneous Graph Transformer (HGT) to receive the node embeddings of each node in the graph. UMAP is applied to the node embeddings to find the 2-dimensional coordinates of the positions of the nodes in the graph. Finally, the visualization can be created from this data. In this paper, we used a case study to show our pipeline and final visualization. Our case study has shown that the pipeline can be used to visualize heterogeneous data. The interactive visualization answers the domain goals and tasks and is visually appealing. Furthermore, the interactive visualization is insightful and knowledge can be gained from using it.

*e-mail: ramon.dijkstra@student.uva.nl - student number: 12017663
†e-mail: deep.kayal@prosus.com
‡e-mail: m.worring@uva.nl

1 Introduction

Visualizations have shown their potential to gain knowledge from raw data [1]. Nowadays, visualizations are widely used to describe complex data and users are familiar with how data is displayed. A visualization must be user-friendly so that it can be intuitively used. Furthermore, the visualization must cover goals and tasks that the user wants to achieve. In this paper, we will look into different types of data instances and different kind of relations between them. To be concrete, we will visualize heterogeneous data with a graph. A heterogeneous graph contains different types of nodes and edges. To get from data to visualization we have to take several steps. We ultimately would like to know the 2D position of the data instances so that in our visualization, clear clustering can be seen. To achieve this, we first need to analyze the raw data files. These data files must be examined to see the data types, the data attributes, the relations, and others. The data is pre-processed so that it can be used for the creation of the heterogeneous graph. This heterogeneous graph is created based on the relations that were found in the pre-processing stage. The different nodes are based on the different data instances that the dataset contains. The different relations are based on the differences found in the pre-processing stage. The graph is initialized with random node embeddings for each node in the graph. This graph can then be trained with a Graph Neural Network (GNN) to receive specific node embeddings [2]. Within information visualization, we are limited to express the information in high dimensionalities. High-dimensional data is not only non-interpretable to humans, it also does not fit on 2D screens.

To visualize the positions of the nodes we have to transform the trained 256-dimensional node embeddings to 2-dimensional node embeddings. The 2 dimensions are the x and y coordinates of a node that can be shown on a screen. We use the dimensionality reduction technique UMAP to achieve this transformation [3]. When the positions of each node of the graph is determined, we can visualize the data.

To show the results of this method we use a case study on data provided by Prosus[1]. In this case study, we started with an investor, a company and a relations dataset. The visualization is based on the MiniConf visualization created by Rush and Strobelt [4]. The opening Figure 1 shows the result of the visualization. We can see clear clusters and the abilities to filter and extract detailed information. We tested three different models based on the Heterogeneous Graph Transformer (HGT) as proposed by Hu et al. and the Relational Graph Convolutional Networks (R-GCNs) proposed by Schlichtkrull et al. to see which model achieves the best quantitative and qualitative results [5, 6]. The quantitative results are measured by the accuracy, precision, recall and fbeta scores. The qualitative results are manually measured by examining the resulting visualizations from the different node embeddings. Besides that, the silhouette scores of the different models are compared.

Contributions: We contribute by providing a pipeline from heterogeneous data to a visualization. The pipeline is implemented in a modular way and can therefore be adjusted to every heterogeneous dataset. We also contribute by showing the potential of the visualization by our case study.

Outline of the paper: We will first discuss related work on GNNs, HGT, and the dimensionality reduction techniques in section 2. In section 3, we will discuss the methodology of the paper including a visualization of the pipeline. Section 4 contains implementation details. Section 5 introduces the case study by describing the domain background and data. Then, section 6 examines the domain goals and tasks. Section 7 described how our final model is chosen. Section 8 shows an extensive explanation of the components of the visualization. The design choices are discussed in section 9. The analysis of the visualization is done in section 10. Lastly, we conclude the paper in section 11.

2 Related Work

In this section, we start by discussing previous work on Graph Neural Networks. Then, the dimensionality reduction techniques are discussed.

2.1 Graph Neural Networks

The past years have shown wide improvements on deep learning on graphs [2]. Graph Neural Networks (GNNs) are becoming the standard way to deal with graphs. At first, only homogeneous graphs were taken into account. Later, heterogeneous graphs were also examined. A heterogeneous graph is a graph containing different types of nodes and edges. According to the work by Bacciu et al. and the taxonomy provided by Chami et al. [2, 7] R-GCN showed to be the best way to deal with heterogeneous graphs in a neural network setting [6]. Building on top of R-GCN, the Heteregenous Graph Transformer (HGT) can be used on heterogeneous graphs in an unsupervised setting.

HGT follows the three main steps of the classical Graph Convolutional Network (GCN) [8]. For every node, it first gathers the neigbours embeddings. On these embeddings, an aggregation function is applied. HGT uses the mean aggregation function while others have used the sum or the max as an aggregation function [8]. The sigmoid function is applied to this outcome to achieve the non-linearity. This is then given to the neural network layer. HGT differs from a classic GCN in two ways. HGT deals with heterogenous graphs and therefore updates a projection matrix for each relation. The default GCN has one projection matrix for all the relations in the graph [8]. The other difference is that HGT applies attention. Attention is used to amplify the interesting nodes in the graph. This shows an improvement in results.

2.2 Dimensionality reduction

Dimensionality reduction techniques have been an extensive field of research within the ML/DL community [9]. Dimensionality reduction techniques nowadays differ from 'simple' Principle Component Analysis (PCA) to mathematically heavy methods like UMAP. The state-of-the art methods are at the time of writing t-SNE and UMAP [3, 10]. They both show excellent results in clustering high-dimensional data. Therefore, both algorithms can be chosen for dimensionality reduction. When comparing the approaches of McInnes et al. [3] and Maaten et al. [10], UMAP showed to better preserve the global appearance of the data in comparison to t-SNE, it is faster, and t-SNE is focussed on the visualization whether UMAP is focussed on the theory behind dimensionality reduction. Therefore, we have chosen to use UMAP.

3 Methodology

In this paper, we will provide a solution to the problem of visualizing heterogeneous graphs. We have created a pipeline from data to an interactive visuazliation. A visualization of the pipeline is shown in Figure 2.

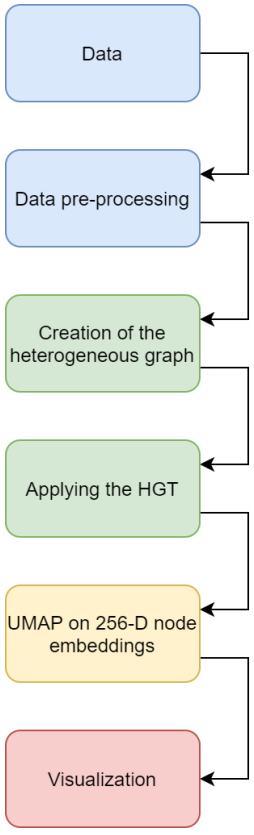


Figure 2: A visualization of the methodology used in this paper. *Blue: data-related. Green: graph-related. Yellow: transformation. Red: end-product.*

[1] https://www.prosus.com/

Within the database, the important data items are selected. After selection, the target data is pre-processed so that it is ready for transformation. During pre-processing, we extract unique data items and remove duplicates. We also take a first look at the nodes and edges that we will include in the heterogeneous graph.

In the next stage, we build the heterogeneous graph by including nodes and edges that are found in the data-preprocessing step. When the graph is created, we use HGT and R-GCN to retrieve node embeddings. We use three different implementations to test if our node embeddings show good results. The first is created in this paper. Next, the HGT and R-GCN models of Harprecht[2] are compared to see which model performs the best [11]. This neural network is used to extract node embeddings from the data in the graph.

These node embeddings are of dimension 256 which is impossible to visualize. We therefore apply the dimensionality reduction technique UMAP to these node embeddings to receive 2D coordinates of the nodes [3].

In the last stage, we combine the processed data with the node embeddings retrieved with the model. We will then use this combination to visualize our data. This visualization can then be interpreted and used so that knowledge is gained. To show how the framework would work in practice, we use a case study.

4 Implementation details

Building and using graphs is covered by the Deep Graph library (DGL) and Pytorch Geometric [12, 13]. Both are python-based libraries that can ensure easy creation and use of heterogeneous graphs. The authors of HGT have built their model[3] on top of both libraries. DGL however has shown to be intuitive in the creation of the graph while it is less intuitive in Pytorch Geometric. Our implementation can be found on github[4].

5 Case Study: Domain background and data

The data used for the examination of our proposed framework is provided by Prosus. As stated on their website, 'Prosus is a global consumer internet group and one of the largest technology investors in the world'.

5.1 Dataset

Three datasets were used during this research. The first dataset is the Company dataset which consists of 11377 different companies. Each company has its own unique identifier. This unique identifier is the primary key to the company. Each company has 98 attributes besides the primary key. These attributes differ from the company name, the net of all capital, the growth rate of their socials, and others. The second dataset is the Investor dataset which consists of 18377 unique investors. Each investor also has its own unique identifier. This unique identifier is also the primary key. For the Investor dataset, each investor has 106 attributes besides the primary key. In this case, investor attributes differ from the investor name, their primary investor type, the maximum fund size that the investor will open, and others. For both the Company and the Investor dataset holds that in some cases not all attributes are relevant or filled in. This means that most of the companies and investors have some empty fields in the dataset. This will be handled by the data preprocessing which is discussed below. Lastly, the InvestorInvestmentRelation dataset is used. This dataset consists of 519123 different deals between investors and companies. Each deal contains the primary key of a company and the primary key of an investor. Besides that, 9 other attributes are associated to each deal. These attributes differ from the deal type, the deal size, the industry, and others. Investors and companies can have multiple deals. To give an example, we could have two deals of size X and Y between investor 123456-78 from the Investor dataset and company 234567-89 from the Company dataset.

5.2 Preprocessing and implementation

Before we could create the graph visualization we have preprocessed the data. The InvestorInvestmentRelation dataset contains information which is not available in the other data. The data items where no metadata could be found are excluded in our research. We have found 10722 unique company IDs and 7320 unique investor IDs that contain metadata. Some companies were also investors. We therefore made a distinction between pure companies and investor/companies. Each investor also belongs to the investor/company class. Our graph ended up with 9779 companies and 7883 investor/company nodes. We created a heterogeneous graph with 50087 (investor/company, invests_in, company) and 5102 (investor/company, invests_in investor/company) links. The invests_in edge corresponds to a deal. We have 55189 edges/deals in our graph.

5.3 The three different models

After preprocessing the data, we implemented the HGT algorithm to receive the node embeddings. The training objective of the first model was based on link prediction [5]. The features were investor IDs. The labels contain company IDs. The algorithm learns to predict if an investor has an investment in a company. With this algorithm, node embeddings of the graph can be created. When the graph is trained, one additional epoch extracts the node embeddings of the final state. These node embeddings are afterwards transformed to 2D coordinates with UMAP [3]. We saw that the overlap of knn (k=5) before and after UMAP was only 8 percent. This can be explained by the fact that dimensionality reduction techniques always loses information.

The second and third model are created by Harprecht at the same time of this case study. These implementations are based on an extensive set of datasets. Besides the company and investor data, additional data about the company websites, patent information, company reviews, news articles, and others are included [11]. In both the R-GCN as well as the HGT implementation, the same objectives were set in training. The training objective is binary funding round prediction in the next 12 months. Because the same settings were used for both R-GCN and HGT, it is thus a fair comparison between these models.

6 Case study: Domain goals and tasks

The main goal of this research is to visualize the datasets in an attractive manner so that the user can easily get insights to the data. We have set up three design goals (DG) to achieve this result.

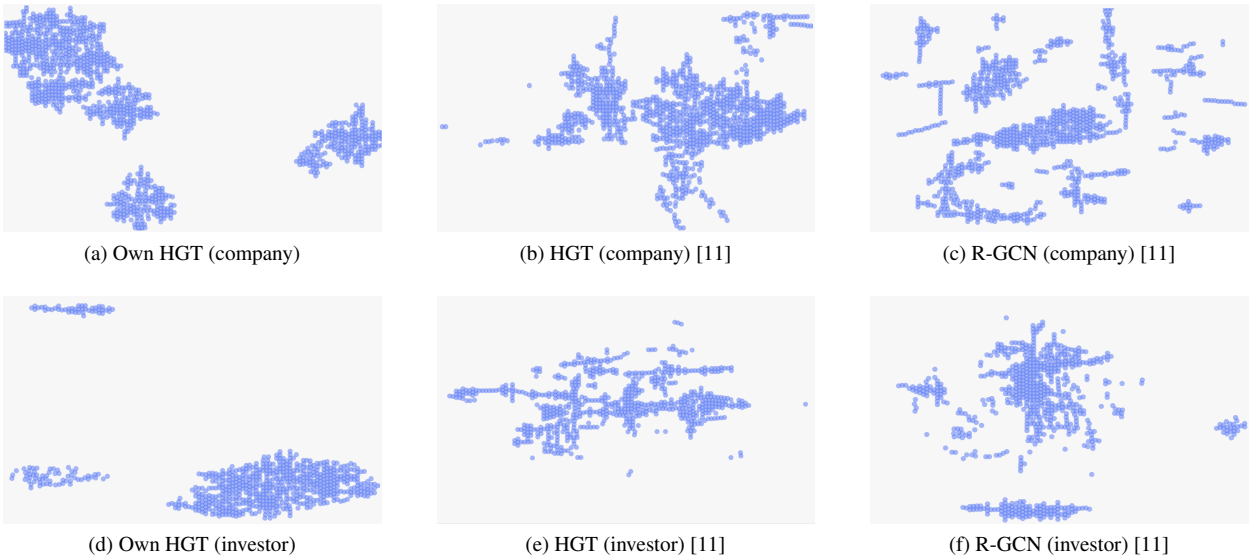
- **DG1:** The user should be able to see a clear overview of the companies and investors.
- **DG2:** The user should be able to explore regions of interest.
- **DG3:** The user should be able to find specific details for companies and investors.

The visualization should be in line with these goals. We will refer to the suboals in further sections.

[2]Carlo Harprecht was part of the AI team of Prosus during the case study. Within the project of Harprecht, an extensive data exploration was performed which resulted in the created models. The results used from Harprecht are at the time of writing still in progress but already show the directions.

[3]https://github.com/dmlc/dgl/tree/master/examples/pytorch/hgt

[4]https://github.com/ProsusAI/Graph-Visualization



(a) Own HGT (company) (b) HGT (company) [11] (c) R-GCN (company) [11]

(d) Own HGT (investor) (e) HGT (investor) [11] (f) R-GCN (investor) [11]

Figure 3: Qualitative comparison between the three different models. Each model is visualized using the same settings.

7 Chosen HGT model

In this section, we quantitatively and qualitatively measure the different implementations of the GNN. In Table 1 we can see the results of the accuracy, precision, recall and fbeta scores for the different models. We would like to achieve a high score on all metrics but are most interested in the precision metric. Therefore, we choose to set the beta of the fbeta score to 0.2, meaning that we give more weight to precision. The HGT model from Harprecht performs the best [11] on all metrics besides precision. The low results of our own HGT model can be explained by the fact that we only use two datasets. Harprecht did an extensive data exploration, combining several datasets and training the models with all this data.

Model	Accuracy	Precision	Recall	Fbeta (0.2)
Own HGT	0.2585	0.1394	0.1109	0.1301
HGT [11]	**0.7558**	0.6176	**0.1458**	**0.5493**
R-GCN [11]	0.7549	**0.6415**	0.1181	0.5480

Table 1: The accuracy, precision, recall, and fbeta scores for the different models.

All models give node embeddings back and can be visualized. The qualitative comparision between the different models is visualized in Figure 3. The qualitative comparison is made on whether the nodes are well clustered. The own HGT models only contain a couple clusters whether the models from Harprecht shows a wider clustering. We also take the way the user interacts with the system into account. When nodes are less cluttered, it is easier in use. R-GCN is the least cluttered graph. Besides that, we calculated the silhouette scores for each figure in Figure 3. The results can be seen in Table 2.

Model	Company	Investor
Own HGT	0.5565	0.5101
HGT [11]	0.4512	0.5209
R-GCN [11]	**0.5835**	**0.5288**

Table 2: The silhouette scores for the different models (the higher, the better).

For the rest of the paper, we will use the node embeddings from Harprechts' R-GCN because it performs the best on the combination of quantitative and qualitative results. The difference between the accuracy, recall and fbeta scores are negligible.

8 Components of the visualization

In this section, we will describe the different components of the visualization. We will describe the filters, the tooltip, the different pages, the way the keywords are chosen, and the way interaction works.

8.1 Displayed nodes

The main part of the visualization is the displayed nodes in the middle of the page. In our case, this corresponds to Figure 3c and Figure 3f. However, this can be adjusted to different views if other node embeddings are used in the visualization. Each node in the graph is a company or an investor. This main part of the visualization covers **DG1** which aimed at a clear overview of the companies and investors.

8.2 Filters

Here, we will discuss the different filters that can be used in our visualization. All the filters that we will showcase in this section are activated at the same time. This choice is made because otherwise we would have too many nodes on one view. This will result in a slower and less comprehensible visualization. Switching between these filter can easily be done by clicking on the buttons. The first filter can be seen in Figure 4 and this filter switches between companies and investors. This distinction is made because we have two different types of nodes. It is initialized on companies so that we first see company nodes.

Figure 4: Filter between companies and investors.

Figure 5 shows the ability to filter on the size of companies/investors. The user can choose small, medium, and large size and it is initialized on medium size. For both companies and investors, all

the attributes of the dataset were investigated to see which attribute would be most suited to filter on. As said previously, the dataset contained missing data values meaning that we could not use every attribute to filter on.

Figure 5: Filter between small, medium, and large companies/investors.

To be precise, for companies, the ActiveInvestors attribute contained the least missing data values for each company instance. We only had 85 missing data values whereas other attributes contain 2500+ missing data values values. We thus chose the ActiveInvestors attribute because otherwise we would lose a large amount of information. We decided that small companies are companies that have 1 or less active investors. Medium companies have between 1 and 5 active investors and large companies have 5 or more active investors. We also took into account that we want the division to be as divided as possible so that we do not have an information overload on one page while having only a couple of nodes on another page. We therefore ended up with 2015 small companies, 3807 medium size companies, and 3872 large companies.

For investors, the Investments and TotalInvestments attributes contained zero missing data values. The others attributes again contained more than 2500 missing data values. We could therefore choose between the Investments and TotalInvestments attribute. We have chosen to use the TotalInvestments attribute. For small investors, we decided that the TotalInvestments attribute must be smaller or equal to 10. Medium size investors have between 10 and 50 total investments. Lastly, large size investors have above 50 total investments. Here, we also take an equal division of nodes into account. We ended up with 2176 small investors, 3291 medium size investors, and 2416 large investors.

Figure 6: Filter between different regions of the world.

Figure 6 contains the filter on regions of the world. Here, we can switch between different continents. Most of the companies and investors are in the region America, Europe, and Asia. However, we also have the option to explore the Middle-East, Oceania, and Africa.

8.3 Tooltip

When a node is hovered, a tooltip is displayed. This tooltip contains the most important information of a node. An example of this can be seen in Figure 7. For investors, we show the title, the HQ location, the primary investor type and the investors status. For companies, the title, the HQ location, the universe, and the business status is shown.

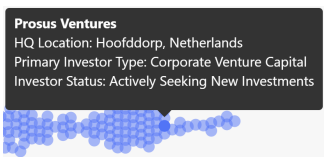
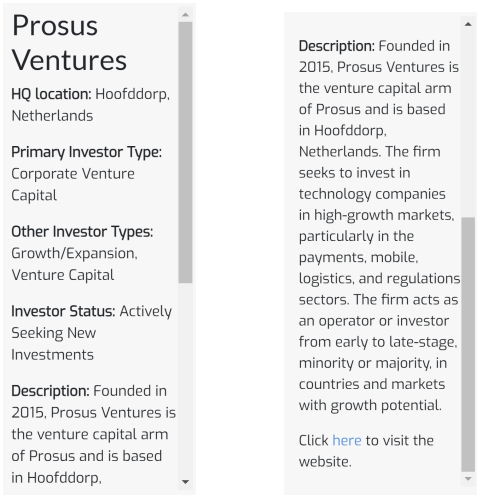


Figure 7: An example of a shown tooltip.

8.4 Detail page

The detail pages for companies and investors are both created in the same way. The idea of this detail page is to provide a detailed description of companies and investors. Thereby selecting the most important data attributes for both investors and companies. In this way, we want to tackle **DG3** which said that users should be able to find specific details for companies and investors. For investors, an example of a detail page can be seen in Figure 8. We display the title on top. Thereby, we show the HQ location, primary investor type, other investor types, the investor status, and a detailed description. Lastly, a clickable link to the website is added. In this way, the user can get a quick but detailed overview of the details of the investor.

The detail page looks the same for companies but with a bit different attributes. For companies, we showcase the title and keywords associated with the company on top. Other attributes are the HQ location, the universe in which the company operates, the business and ownership status, a detailed description and again a clickable link to the website of the company.



(a) (b)

Figure 8: Example detail page investor.

8.5 Exploration page

When a region of nodes is selected, 15 keywords related to the nodes in that region are displayed on the right side of the screen. Below these keywords, a quick overview of other companies or investors are shown. Figure 9 shows an example when multiple nodes are selected. The keywords generated on top are based on TF-IDF [14]. TF-IDF calculates the importance of a word with respect to the document. In our case, the document is the description of a company or investor. We thus examine the importance of words within the descriptions and the 15 most relevant words are shown on top. These keywords can then be used to explore regions of interest and fulfill **DG2**. Note that these keywords differ from the keywords that are displayed on top of companies in the detail page. The keywords in that case were extracted from the data. These keywords are inferred from the description and are calculated over multiple nodes.

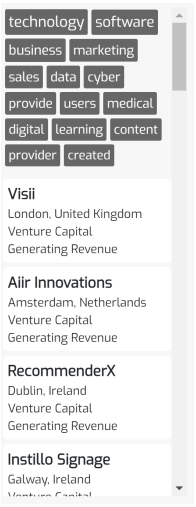


Figure 9: Exploration page

8.6 Interactions

As discussed in the previous sections, we can interact with the system by using the filter options between displayed nodes, the size, and the region. We also showcased how the main part, the exploration page, and the detail page of the visualization looks like and why certain choices are made in the process of building the visualization. In this section, we will elaborate on all the other interactions that are possible with our visualization.

Pike et al. discussed that an interactive visualization can fulfill user goals and tasks [15]. Therefore, we will elaborate how the interaction is implemented in our visualization. We do this by creating an interaction model based on the seven interaction categories from Yi et al. [16]. Yi et al. mentioned that the seven main interaction techniques are select, explore, reconfigure, encode, abstract/elaborate, filter and connect. We will use the select, abstract/elaborate, and filter interaction techniques. Following the terminology from Yi et al., select means 'mark something as interesting', abstract/elaborate means 'show me more or less detail', and lastly filter means 'show me something conditionally'. With this theory, we created the interaction model which can be seen in Figure 10.

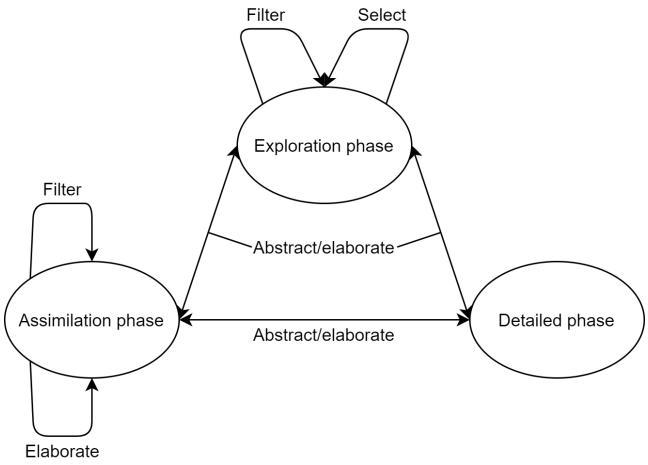


Figure 10: Interaction model of the visualization.

The user starts in the assimilation phase where four interaction can be performed. First, the user can filter the data on displayed nodes, the size, and the region. The second interaction is hovering over a node. This will show the tooltip and is thus the elaborate class. When clicking on a node, the data is elaborated and the detail page is shown. Lastly, when multiple nodes are selected, the data is elaborated and the exploration page is shown.

In the exploration page, again four interactions are possible. We can filter in the same way. The user also has the possibility to click on the keywords. When this action is performed, nodes that contain the keyword in their description are highlighted in red. From there, other nodes can be clicked and the search keywords will be remembered by the system. If we click on the screen, we abstract back to the assimilation phase. Lastly, the companies or investors below the keywords from 9 are clickable. Then, the detail page of the clicked node will be shown on top of the exploration page. In this way, multiple nodes can be explored within a specific region.

Within the detail page, the detailed information for a company or investor is shown. Also, the link to the website can be clicked.

9 Design choices

In this section, we will discuss the design choices that are made for the visualization. We will go over each component of the visualization and describe why certain colours, shapes and ways of displaying are used.

9.1 Displayed nodes

For the displayed nodes, we have chosen to use the blue colour with some opacity. Because we are having thousands of nodes, we have chosen to use opacity. In this way, nodes that partly overlap can still be seen. The blue colour is chosen to not have a colour that is too soft and also not too hard. This is also done because we have thousands of nodes displayed at the same time and therefore we do not want to overwhelm the user. When a node is clicked, it becomes grey. This is done so that users know which nodes are clicked. Besides that, Ware concluded that colours and shapes are the most prominent ways to declare differences [17]. In this way, we can see a clear distinction between clicked and other nodes.

9.2 Filters

The filters that we use are all designed in the same way. This is done so that the user is convinced that each filter works in the same way. The filter part that is selected are all coloured dark gray with white letters. This stands out in comparison to the non-selected parts of the filters and therefore it is intuitive to the user. The filters that are not selected are white with grey letters.

9.3 Tooltip

The tooltip is a black box with white letters. This is done to strongly attract the attention of the user to the tooltip. We have chosen a high contrast in this block so that the important information is even more obvious shown. The title of the node is displayed in bold letters because this is the most important aspect of the tooltip. Below that, the rest of the information is shown in normal letters.

9.4 Detail page

On the detail page, the title of the node is shown with a large header. This is done so that the user is attracted to this first. Then, for every data attribute that is shown, the data attribute title is displayed in bold whereas the information corresponding to this attribute is shown in normal letters. This is again designed in this way to first attract the user to the headers. The link to the website is shown in blue because that is the most intuitive way for a user to know it is clickable.

9.5 Exploration page

The exploration page contains blocks of keywords on top of the summarized description of the nodes below. The keywords are shown in blocks so that it is intuitive for the user that they are clickable. When a keyword is clicked, the keyboard border is made black and the corresponding nodes in the graph are coloured red. This red colour is chosen to amplify that a node is visible there. Furthermore, a node also colours red when the user hovers over the blocks below the keywords on the exploration page. The blocks are again designed to focus on the bold header, and then the rest of the information can be read.

10 Analysis

To analyze our visualization, we will make use of the Trifecta Checkup[5] and Van Wijk's criterion [18].

10.1 Trifecta Checkup

The Trifecta Checkup by Fung states that there are three main criteria for a visualization. The first is that the visualization must answer a question. In our case, we will refer back to the goals and tasks set in section 6. The second criteria is that the data is insightful. Lastly, the visualization is rated in this Trifecta Checkup. Ideally, the visualization should receive a QDV score meaning that it has done the question, the data, and the visualization part well.

The question criteria is covered when the user goals and tasks are fulfilled. We partly discussed this in the components of the visualization section. **DG1** is fulfilled because the user can immediately see an overview of the nodes of companies and investors when the interactive visualization is opened. **DG2** is covered by two main functionalities. The first is the selection of multiple nodes and the exploration page that pops up on the right of the screen. The second is the ability to highlight nodes based on the keywords that are shown in the exploration phase. Both functionalities ensure that the user is able to explore regions of interest. Lastly **DG3** is covered by the detail page that pops up when a user clicks on a node. We can conclude that each design goal is fulfilled and therefore we receive the Q score in the Trifecta Checkup.

The data criteria is covered when the data shows insightful information. We have tested the visualization with three different sets of node embeddings. The first model was focussed on predicting whether a company is invested by an investor. The latter two are focussed on predicting company succes in the next 12 months. The visualization shows the node embeddings in a proper way. However, one could extract even more interesting information when the network is trained on the keywords in the description. Now, multiple clusters contain the same keywords. If we change the model to training on keywords, we could for example have a cluster in the top right about health care, a cluster in the middle for cyber security, etc. Our data shows insightful information. However, this could be improved in the future. Therefore, we do not receive the D score in the Trifecta Checkup.

The visualization criteria is fulfilled when the visualization is user friendly and visually appealing. The interactions that are implemented in our visualization are intuitive and therefore users can easily use the visualization. The colours that we use are intuitive as well. Besides that, the structure of the visualization with filters and the assimilation phase, the exploration phase, and the detailed phase resulted in a good visualizaton. Therefore, we receive the V score in the Trifecta Checkup. We thus end up with a QV score where the D score can be achieved by training the neural network in a different manner.

[5] https://junkcharts.typepad.com/junk_charts/junk-charts-trifecta-checkup-the-definitive-guide.html

10.2 Van Wijk's criterion

Besides the Trifecta Checkup, we will discusss the gained knowledge after our visualization is used based on Van Wijk's criterion [18]. We will describe the actions that can be performed after our visualization is used.

Our visualization shows companies and investors. There are three main actions that can be performed after our visualization is used. The first is that the user now knows relations between the different companies and investors. The user could write down the interesting companies and investors that are related to each other. In this way, an investor could quickly get an overview of companies that are interesting to invest in.

Another action that could be done after our visualization is used is filtering out certain regions or categories. A user could explore whether technologies are used in different continents. When for example, companies in Africa do not focus on cyber security, the user could exclude this from his interesting companies.

Lastly, a company could find out where investors are interested in. It could then approach investors to ask whether they are interested to invest in them. This is especially useful for start-ups who need investments to grow. With this visualization, the interesting companies could be spotted and contacted.

11 Conclusion

In this paper, we discussed how we could create an interactive visualization from raw heterogeneous data. We thereby extensively looked into the visualization aspects of our results. The pipeline from data to visualization is created. When we use the data, we pre-process it so that the heterogeneous graph can be created. This heterogeneous graph can then be trained to gather node embeddings for each node in the graph. These high-dimensional embeddings can then be transferred to 2D coordinates by applying UMAP. The positions of nodes can then be added to the original data to build a visualization. In our case study we have seen that multiple types of node embeddings can be used in the visualization. We discovered that the visualization gets a QV score in the Trifecta Checkup. Besides that, with Van Wijk's criterion, we found out that the visualization can be used for multiple purposes. The case study has shown that the pipeline can be used with heterogeneous data. The interactive visualization can cover domain goals and tasks when they are specified.

References

[1] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases," *AI magazine*, vol. 17, no. 3, pp. 37–37, 1996.

[2] D. Bacciu, F. Errica, A. Micheli, and M. Podda, "A gentle introduction to deep learning for graphs," *Neural Networks*, 2020.

[3] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[4] A. M. Rush and H. Strobelt, "Miniconf–a virtual conference framework," *arXiv preprint arXiv:2007.12238*, 2020.

[5] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," in *Proceedings of The Web Conference 2020*, 2020, pp. 2704–2710.

[6] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *European semantic web conference*. Springer, 2018, pp. 593–607.

[7] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, and K. Murphy, "Machine learning on graphs: A model and comprehensive taxonomy," *arXiv preprint arXiv:2005.03675*, 2020.
[8] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[9] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, no. 66-71, p. 13, 2009.
[10] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. 11, p. 2579 – 2605, November 2008.
[11] C. Harprecht, "Predicting company success using graph data," 2021.
[12] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma *et al.*, "Deep graph library: Towards efficient and scalable deep learning on graphs." 2019.
[13] M. Fey and J. E. Lenssen, "Fast graph representation learning with pytorch geometric," *arXiv preprint arXiv:1903.02428*, 2019.
[14] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, 1972.
[15] W. A. Pike, J. Stasko, R. Chang, and T. A. O'connell, "The science of interaction," *Information visualization*, vol. 8, no. 4, pp. 263–274, 2009.
[16] J. S. Yi, Y. ah Kang, J. Stasko, and J. A. Jacko, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE transactions on visualization and computer graphics*, vol. 13, no. 6, pp. 1224–1231, 2007.
[17] C. Ware, *Visual thinking for design*. Elsevier, 2010.
[18] J. V. Wijk, "Views on visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, p. 421–432, 2006.