Fachhochschule Graubünden
University of Applied Sciences

Consultancy Project 1

Dynamische Graphen mit Python

Fachhochschule Graubünden

2023

Referent

Prof. Dr. Wolfgang Semar
Wolfgang.Semar@fhgr.ch

Autor und Autorinnen

Güner Mahmut
mahmut.guener@stud.fhgr.ch

Koller Sara
sara.koller.msc@stud.fhgr.ch

Kuriger Anna
anna.kuriger@stud.fhgr.ch

Abstract

Die vorliegende Arbeit widmet sich der umfassenden Evaluation und Bewertung von Python-Bibliotheken zur Visualisierung von dynamischen Graphen für den Praxiseinsatz der itopia AG. Zunächst wurde eine Bewertung von insgesamt elf Bibliotheken anhand verschiedener Kriterien (Funktionalität, Flexibilität, Interaktionsmöglichkeiten, Skalierbarkeit und Anpassungsfähigkeit) durchgeführt, um die vielversprechendsten Optionen zu identifizieren. Aufgrund dieser Bewertung wurden die drei am besten abschneidenden Bibliotheken Bokeh, igraph und PyVis genauer untersucht und auf ihre Praxistauglichkeit hin getestet.

In den detaillierten Untersuchungen der ausgewählten Bibliotheken werden ihre Funktionen, Möglichkeiten und Grenzen eingehend erläutert. Bokeh erweist sich als interaktive Bibliothek mit vielfältigen Funktionen zur Visualisierung und Konfiguration von Netzwerkgrafiken. igraph hingegen zeichnet sich durch seine leistungsstarken Funktionen zur Erweiterung des Codes aus. Die finale Evaluation ergab, dass PyVis im Vergleich zu Bokeh und igraph einige Vorteile bietet, wie beispielsweise eine benutzerfreundlichere Oberfläche, ein breiteres Spektrum von Interaktionsmöglichkeiten und eine umfassende Dokumentation. Es wurde jedoch festgestellt, dass PyVis hauptsächlich auf die Visualisierung spezialisiert ist und keine erweiterten Funktionen zur Datenbankanalyse bietet. Für den praktischen Einsatz durch die itopia AG wird empfohlen, PyVis weiter zu untersuchen und mögliche Anpassungen oder Erweiterungen zu testen, um sicherzustellen, dass es den spezifischen Anforderungen gerecht wird.

Die vorliegende Arbeit diskutiert auch die Grenzen der angewandten Methodik und der evaluierten Bibliotheken. Es wird darauf hingewiesen, dass die Bewertung subjektive Einschätzungen beinhaltet und die Auswahl der Kriterien eine gewisse Einschränkung darstellen kann. Die Untersuchung beschränkte sich auf eine begrenzte Anzahl von Bibliotheken und basierte auf einem bestimmten Datensatz. Eine breitere Evaluierung und Performance-Tests mit komplexeren Datensätzen sind empfehlenswert, um die Skalierbarkeit der Bibliotheken noch besser beurteilen zu können.

Abschliessend liefert die Arbeit praktische Implikationen für den Einsatz von PyVis zur Visualisierung von Graphen. Sie gibt einen umfassenden Überblick über die Bewertung der Bibliothek und empfiehlt weiterführende Untersuchungen, um sicherzustellen, dass die ausgewählte Bibliothek den spezifischen Anforderungen und Anwendungsfällen gerecht wird.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

Bspw.	Beispielsweise
CSV	Comma-separated values
et al.	et alii, und andere
etc.	et cetera, und so weiter
JSON	JavaScript Object Notation
S.	Seite
TXT	Text Datei

1 Einführung

Im Rahmen des Moduls Consultancy Project 1 unseres Masterstudiengangs in Data Visualization an der Fachhochschule Graubünden (FHGR) durften wir, Mahmut Güner, Sara Koller und Anna Kuriger ein Projekt mit der Auftraggeberin, die itopia AG mit Sitz in Zürich, durchführen.

Bei der Eintragung für dieses Projekts waren für uns zwei Aspekte von besonderer Bedeutung: Zum einen handelt es sich um ein Projekt im Bereich der IT-Werkzeuge für die Finanzwelt, die allgemein als sehr dynamisch beschrieben werden kann. Zum anderen fordert das Projekt ein Interesse an der Programmiersprache Python. Es bot die Möglichkeit, die aus dem ersten Semester erlangten Python-Programmierkenntnisse in einem praktischen Projekt anzuwenden, zu erweitern und zu vertiefen.

1.1 Ist-Situation itopia AG

Die itopia AG stellt Unternehmen aus der Finanzwelt ihre umfangreiche Expertise im Bereich der Digitalisierung und dem Ersatz von Kernsystemen zur Verfügung, um sie bei ihren IT-Initiativen zu unterstützen. Die Berater der itopia AG verfügen über ein tiefes Verständnis für das Geschäftsmodell der Kunden und beherrschen sowohl die Fachsprache des Managements als auch die der Business- und IT-Experten. Durch ihre langjährige Erfahrung im Management sind sie in der Lage, die Kernaspekte präzise zu fokussieren und auf den Punkt zu bringen. Die itopia AG zeichnet sich durch Neutralität und Unabhängigkeit aus, wobei der primäre Schwerpunkt stets auf dem Geschäftserfolg der Kunden liegt. Die Beratungstätigkeiten und Technologieinitiativen der itopia AG werden gezielt darauf ausgerichtet, die individuellen Ziele und Anforderungen der Unternehmen zu unterstützen. (itopia, 2023)

Um die gewonnenen Erkenntnisse aus Kundenaufträgen, insbesondere Zusammenhänge und Abhängigkeiten, für die Kunden anschaulich darzustellen, werden statische Graphen (Abbildung 1) verwendet. Die Erkenntnisse können beispielsweise IT-Architekturen darstellen, in denen Zusammenhänge zwischen Services, Applikationen und Infrastrukturen visualisiert werden. Aktuell nutzt die itopia AG zur Erzeugung dieser statischen Graphen Graphiz und neo4j Bloom. Letzteres findet überwiegend in Workshops Anwendung, um interaktiv mit graphenorientierten Daten zu arbeiten.

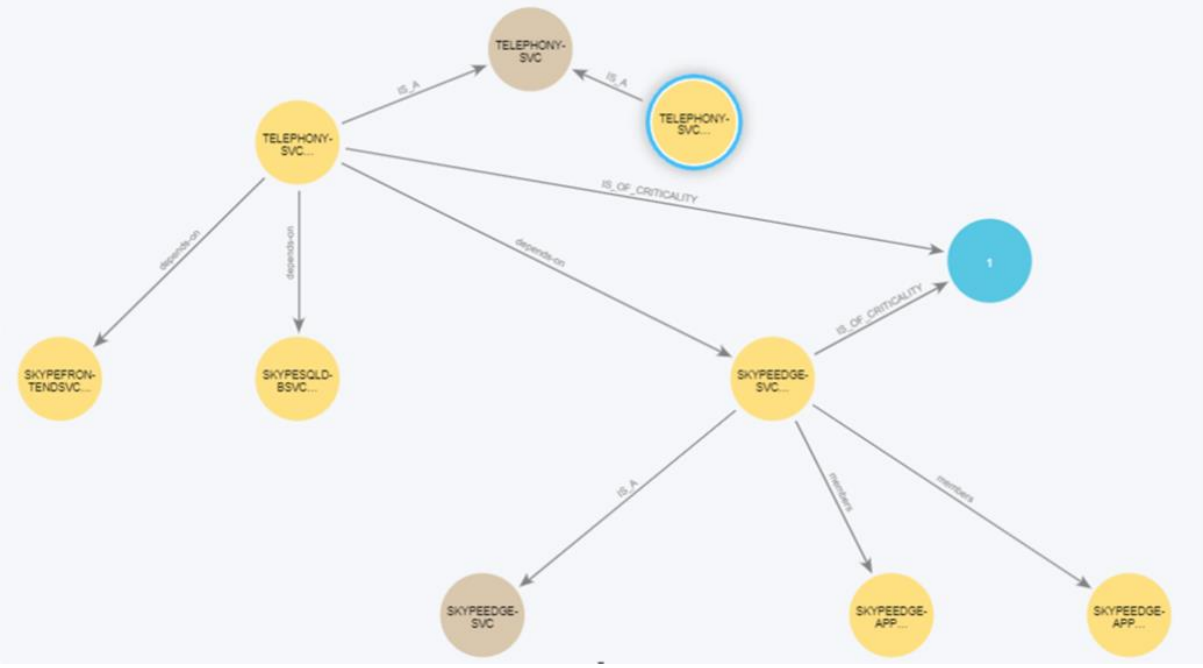

Abbildung 1: Beispiel einer Netzwerkgrafik von der itopia AG.

1.2 Projektauftrag und Erstgespräche

Grundlage des definitiven Projektauftrags war nachfolgende Projektbeschreibung der itopia AG und der FHGR: „Nach einer Bedarfsklärung (High-Level Anforderungen in 2-3 Interviews zu erfassen) ist ein selbständiges Screening und Evaluation von (im besten Fall Python-basierten) Open Source Werkzeugen zur Darstellung von Graphen durchzuführen. Idealerweise erlaubt ein Werkzeug die dynamische Visualisierung oder bestenfalls sogar interaktive Kommentierung von Daten (Nutzung in Workshops / in Reviews etc.). Neben Anforderungen kann itopia Beispiele und bei Bedarf auch synthetische Daten für die Evaluation von Werkzeugen / Frameworks zur Verfügung stellen. Ebenfalls kann ein Python-Experte für Fragen zur Verfügung stehen.” (itopia AG, Projektbeschrieb)

Im Kick-Off-Gespräch am 27. Februar 2023 mit der itopia AG erlangte das Projektteam ein vertieftes Verständnis des Projektes. Basierend darauf wurden gezielt Fragen gestellt, um die spezifischen Anforderungen, Wünsche, Rahmenbedingungen, Abgrenzungen sowie das weitere Vorgehen in einem Interview beziehungsweise einem Austausch mit der itopia AG zu ermitteln. Der Austausch fand am 16. März 2023 zwischen dem Projektteam und der itopia AG statt und ermöglichte die Ausarbeitung des definitiven Projektauftrags.

Der vollständige Projektauftrag kann dem Anhang 1 entnommen werden. Im weiteren Verlauf werden die definitiven Projektziele und Projekt-Abgrenzungen erläutert.

1.2.1 Projektziele

Das übergeordnete Ziel des Projekts besteht darin, eine Empfehlung für die Einführung einer geeigneten Open Source Bibliothek (basierend auf Python) zur Visualisierung von Graphen auszusprechen. Die Bibliothek soll sowohl in der Lage sein, dynamische Graphen zu generieren, als auch interaktive Funktionen wie Kommentar- und Einblendungsfunktionen ermöglichen. Um die Leistungsfähigkeit der ausgewählten Bibliothek zu veranschaulichen, wird ein Anwendungsbeispiel mit synthetischen Daten des Auftraggebers erstellt.

Das Projektteam hat die Projektziele dahingehend eingeschränkt, dass die Empfehlung zwingend eine Python-basierte Bibliothek sein sollte, da dies aus den Gesprächen mit der itopia AG als äusserst wünschenswert erachtet wurde.

1.2.2 Abgrenzung

Während den Gesprächen mit der itopia AG, insbesondere des Interviews, wurde das Projekt in mehreren Bereichen abgegrenzt:

1. Analyse und Evaluation: Ein wesentlicher Teil des Projekts besteht aus einer umfangreichen Analyse, die auf praktischen Anwendungsfällen basiert. Hierbei wird ein Evaluationsschema erstellt, das auf Kriterien aus der Literatur basiert.

2. Implementierungsumfang: Das Python-basierte Tool wird nicht unternehmensweit für den täglichen Einsatz implementiert, sondern dient vorrangig Testzwecken in einer realen Umgebung, möglicherweise innerhalb des Unternehmens.

3. Prüfung der Projektergebnisse: Die erzielten Ergebnisse werden auf ihre Tauglichkeit hin überprüft, um letztendlich eine angemessene Empfehlung aussprechen zu können.

4. Bereitstellung synthetischer Daten: Die itopia AG stellt synthetische Daten zur Verfügung, um die Evaluation der Werkzeuge und Frameworks zu unterstützen.

Mithilfe der durch die itopia AG bereitgestellten synthetischen Daten wird die Leistungsfähigkeit des Tools in einer realistischen Umgebung demonstriert. Deshalb hat das Projektteam nach dem Interview diesbezüglich die Abgrenzung ergänzt, dass keine Daten aus dem Internet verwendet werden, sondern nur die synthetischen Daten der itopia AG.

1.3 Netzwerkgrafiken

Das im Projektauftrag formulierte Ziel beinhaltet die Empfehlung eines Open Source Tools, das sowohl statische als auch dynamische Graphen generieren kann. Nach weiteren Gesprächen wurde klargestellt, dass das gesuchte Tool in der Lage sein muss, Netzwerkgrafiken zu erstellen. Die dominierende Form der Netzwerkgrafik ist das Knoten-Kanten-Diagramm.

Ein Knoten-Kanten-Diagramm ermöglicht die visuelle Darstellung von Verbindungen zwischen verschiedenen Entitäten, wie zum Beispiel Menschen oder Servern. Dabei werden die Verbindungen durch Knoten und Kanten repräsentiert (Barabàsi et al., 2004, S. 62). Knoten verkörpern die Entitäten und können verschiedene Formen, Farben und Grössen haben, um unterschiedliche Entitätstypen darzustellen.

Die Kanten zeigen die Beziehungen zwischen zwei Knoten (Entitäten). Sie können Linien von verschiedener Dicke, Pfeile oder andere visuelle Elemente annehmen, um die Art der Beziehung zu symbolisieren. Die Kantendicke kann beispielsweise die Stärke der Verbindung anzeigen.

Diese Realitätsreduktion auf Knoten und Kanten wird von Mathematiker Graphen beziehungsweise Netzwerke genannt (Barabàsi et al., 2004, S. 62). Es existieren zwei wesentliche Klassen von Graphen, die gerichteten und die ungerichteten (Neumann, 1989). Wenn die Richtung der Kanten vernachlässigt werden kann, werden die Graphen als ungerichtet bezeichnet. Gibt es eine eindeutige Kantenrichtung, ist ein eine Kante gerichtet.

Netzwerkgrafiken mit vielen Knoten- und Kanten-Elementen werden mit Hilfe von Layoutalgorithmen, Farben oder Interaktionsmöglichkeiten einfacher lesbar gestaltet. Layoutalgorithmen wie das kräftebasierte Layout von Fruchterman und Reingold (1991) versuchen beispielsweise, ein optimales Knoten-Kanten-Diagramm zu erzeugen, bei dem keine Überlappungen von Knoten auftreten und die Kanten sich möglichst wenig schneiden. Kräftebasierte Layouts ermöglichen die Darstellung von Symmetrie (Hachul et al., 2007, S. 358). Neben den kräftebasierten Algorithmen sind hierarchische Layouts im Einsatz, welche sich besonders für Netzwerkgrafiken mit gerichteten Kanten eignen (Keller et al. 2006).

Die Verwendung von Farben kann dabei helfen, verschiedene Entitätstypen oder Beziehungstypen im Diagramm zu unterscheiden. Interaktionsmöglichkeiten, wie das Zoomen, Verschieben oder Filtern des Diagramms, können die Lesbarkeit verbessern und den Benutzern die Möglichkeit geben, detaillierte Informationen zu untersuchen. Es ist jedoch zu beachten, dass eine zu hohe Dynamik in der Netzwerkgrafik die Erkennung von Mustern erschweren kann. Zu viele Bewegungen können einerseits dazu führen, dass es schwierig ist, sich die Graphen zu merken, und andererseits das Gehirn überfordern (Burch, 2023).

2 Methodik

Um das festgelegte Ziel zu erreichen, wurde die qualitative Forschungsmethode verwendet. Konkret wurde die Nutzwertanalyse als geeigneter Ansatz ausgewählt, um die Evaluation und Bewertung verschiedener Python-Tools (Bibliotheken) durchzuführen. Die Nutzwertanalyse bietet eine strukturierte Vorgehensweise, um verschiedene Aspekte von Tools (Bibliotheken) zu bewerten und zu vergleichen. Weiter ermöglicht sie eine transparente Dokumentation des Entscheidungsprozesses, welche sowohl für die Erstellenden als auch für die Auftraggeberin von Vorteil ist, um die Nachvollziehbarkeit und Validität der Entscheidung sicherzustellen.

2.1 Anforderungen

Ein wichtiger Schritt bei der Durchführung einer Nutzwertanalyse besteht darin, Anforderungen zu definieren. Dies dient dazu, den Umfang der Untersuchung festzulegen und sicherzustellen, dass relevante Informationen gesammelt werden, um die Forschungsfragen zu beantworten. Durch die Definition von Anforderungen wird auch im Voraus festgelegt, welche Informationen und Kriterien bei der Bewertung von Alternativen oder Optionen berücksichtigt werden sollen.

Die Beschreibung der Anforderungen bildet die Bewertungsgrundlage der vorliegenden Arbeit. Die Anforderungen werden zunächst im Detail vorgestellt, anschliessend in einem Anforderungskatalog zusammengefasst und ist dem Anhang 2 zu entnehmen (Gebhart, 2014, S. 43).

2.1.1 Beschreibung der Anforderungen

Es existiert eine Vielzahl an vorhandenen Definitionen des Anforderungsbegriffes, von denen keine als allgemeingültig betrachtet werden kann. In dieser Arbeit wird der Begriff primär darauf bezogen, was ein Tool erfüllen soll, und nicht wie dies umgesetzt werden soll (Zehnter C., 2012, S.26-27; Sommerville & Sawyer, 1997).

Obwohl die Anforderungen aus der Literatur abgeleitet werden können, zeigen sie von Auftrag zu Auftrag eine erhebliche Variation. Infolgedessen werden weitere Bedürfnisse in qualitativen Interviews mit dem Auftraggeber eruiert (Anhang 3), um eine ausführliche Erfassung der Anforderungen zu gewährleisten. Die Interviews fungieren als wertvolles Instrument zur direkten Erfassung der Erwartungen, Bedürfnisse und Prioritäten des Auftraggebers. Die erlangten Hinweise aus den Interviews werden dokumentiert und analysiert, um eine solide Grundlage für die weitere Planung und Umsetzung des Projekts zu schaffen (Anhang 3).

Die Anforderungen werden in nichtfunktionale und funktionale unterteilt. Die nichtfunktionalen Anforderungen beziehen sich allgemein auf das Tool. Die funktionalen Anforderungen beschreiben hingegen die Vorgaben an die Darstellung der Grafik (Paulus, 2012, S. 71-72).

Die Berücksichtigung sowohl funktionaler als auch nichtfunktionaler Anforderungen ist von Bedeutung, um die Erwartungen der Auftraggeberin effektiv zu erfüllen.

2.1.2 Gewichtung der Anforderungen

Anforderungen lassen sich in Forderungen (muss) und Wünsche (soll) unterteilen. Forderungen müssen erfüllt werden. Wünsche hingegen sollen nach Möglichkeit berücksichtigt werden, müssen jedoch nicht zwangsgebunden erfüllt werden. Daher ist es notwendig, die Anforderungen mit Hilfe einer Skala zu gewichten (Tabelle 1).

Tabelle 1: Die Gewichtung der Anforderungen.

Wert der Skala	Wichtigkeit
1	Nicht wichtig
2	Eher nicht wichtig
3	Teils wichtig
4	Wichtig
5	Sehr wichtig

Durch die Festlegung und Gewichtung von Anforderungen wird ein klarer Rahmen vorgegeben, um begründete Entscheidungen zu treffen. Dabei spielen verschiedene Anforderungen wie Funktionalität, Leistungsfähigkeit, Community-Unterstützung und Dokumentation eine wichtige Rolle.

Um die Validität sicherzustellen, werden die Gewichtungen jeder einzelnen Anforderung zusätzlich in Abstimmung mit dem Auftraggeber festgelegt.

2.2 Auswahl und Beurteilung der Python-Bibliotheken

Nachdem die gewünschten Funktionen und Eigenschaften der Bibliothek definiert sind, erfolgt eine umfassende Recherche zur Identifikation geeigneter Python-Bibliotheken, die in der Lage sind, dynamische Netzwerkgrafik zu erstellen. Hierbei werden verschiedene Quellen wie Suchmaschinen, Fachleute, Vorlesungsunterlagen der FHGR, Entwicklerforen, GitHub und weitere Plattformen herangezogen.

Schliesslich erfolgte die Auswahl der Python-Bibliotheken, die weiter untersucht und getestet werden. Für jede Bibliotheken wird eine detaillierte Beurteilung anhand des Anforderungskatalogs durchgeführt (siehe Kapitel 3: Ergebnisse). Im ersten Schritt findet die Beurteilung statt, ob die jeweilige Anforderung vollständig erfüllt (2 Punkte), teilweise erfüllt (1 Punkt) oder nicht erfüllt (0 Punkte) ist. Durch die Zuweisung von Punkten für jedes Kriterium wird eine quantitative Grundlage geschaffen, um die Alternativen zu vergleichen und eine fundierte Entscheidung zu treffen.

Nachfolgend wird die Beurteilung für jedes Anforderungskriterium detaillierter beschrieben, wodurch zusätzliche Informationen abgeleitet werden konnten.

2.3 Nutzwertanalyse

Der Anforderungskatalog mit den Gewichtungen und die Beurteilungen der einzelnen Bibliotheken wird letztlich in einer Nutzerwertanalysetabelle zusammengeführt. Dabei werden die einzelnen gewichteten Anforderungen mit den transformierten entsprechenden Zahlenwerten der dazugehörigen Bewertungen (Punkte 0-2) multipliziert. In Abbildung 2 ist ein Ausschnitt der Nutzwertanalyse zu sehen.

Anforderung	Gewichtung	Bokeh		dash (Cytoscape)		PyGraphviz		Holoviews		igraph		Py
		Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	
Nicht funktionale (Tool allgemein)												
Lokales, open Source Werkzeug	5	2	**10**	2	**10**	2	**10**	2	**10**	2	**10**	
Sicherheit / Vertrauenswürdige Quelle	5	2	**10**	0	**0**	2	**10**	2	**10**	1	**5**	
Direkter Datenimport (CSV / JSON / Python Objekte)	4	2	**8**	2	**8**	0	**0**	2	**8**	2	**8**	
Netzwerkgrafikexport (PDF, SVG)	3	2	**6**	2	**6**	2	**6**	2	**6**	2	**6**	
Gestaltung												
Beschriftungsoptionen der Knoten	4	1	**4**	1	**4**	1	**4**	1	**4**	2	**8**	
Knotengestaltung mit Farben	5	2	**10**	2	**10**	2	**10**	2	**10**	2	**10**	
Knotengestaltung mit Formen	4	2	**8**	2	**8**	2	**8**	2	**8**	2	**8**	
Kantengestaltung	4	2	**8**	2	**8**	2	**8**	2	**8**	2	**8**	
Corporate Design Integration Möglichkeit	2	2	**4**	0	**0**	1	**2**	0	**0**	1	**2**	
Anwendbarkeit (Interaktivität)												
Zoom Möglichkeit	4	2	**8**	2	**8**	0	**0**	2	**8**	2	**8**	
Suchoption	4	1	**4**	2	**8**	0	**0**	0	**0**	2	**8**	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit	3	1	**3**	2	**6**	0	**0**	1	**3**	2	**6**	
Übersichtlich Darstellung der Inhalte	4	1	**4**	2	**8**	2	**8**	1	**4**	2	**8**	
Kommentarmöglichkeit	3	0	**0**	0	**0**	0	**0**	0	**0**	1	**3**	
Inputs entgegennehmen	3	1	**3**	1	**3**	1	**3**	0	**0**	1	**3**	
Nutzwertsumme			**90**		**87**		**69**		**79**		**101**	
Rang			3		5		9		8		2	

Abbildung 2: Ausschnitt aus der Nutzwertanalyse.

Eine Aufsummierung über alle Anforderungen ermöglicht eine aussagekräftige Entscheidung für die Aufgabenstellung. In der vorliegenden Nutzwertanalyse besteht eine maximal erreichbare Punktzahl von 114, wenn alle Anforderungen vollständig erfüllt werden.

2.4 Evaluation mittels Beispieldatensatzes

Basierend auf den Nutzwertanalyseberechnungen ergeben sich die drei am besten bewerteten Python-Bibliotheken. Im nächsten Schritt wird die Eignung dieser Bibliotheken überprüft, wobei der Beispielsdatensatz der itopia AG verwendet wird. Die Verwendung des vorgegebenen Datensatzes ermöglicht eine realistische Evaluierung der Nutzerfreundlichkeit der Python-Bibliotheken. Darüber hinaus werden während der Überprüfung mögliche Herausforderungen oder Einschränkungen identifizierbar.

Die Ergebnisse dienen als Grundlage für die endgültige Auswahl der am besten geeigneten Python-Bibliothek, die den Anforderungen des Projekts optimal entspricht und eine effektive Erstellung dynamischer Netzwerkgrafiken ermöglicht.

3 Ergebnisse

3.1 Zusammenstellung des Anforderungskatalogs

Aus dem Kapitel 1.3 Netzwerkgrafik geht hervor, dass für eine klare und verständliche Darstellung der Beziehungen zwischen Entitäten folgende Anforderungen bei der Erstellung von Netzwerkgrafiken erfüllt werden müssen: Verwendung von Formen und Farben zur Darstellung unterschiedlicher Entitäts- oder Beziehungstypen. Layoutalgorithmen erleichtern die Lesbarkeit und Analyse von Netzwerkgrafiken. Interaktionsmöglichkeiten wie Zoomen, Verschieben von Knoten oder Filtern des Diagramms bieten den Nutzern und Nutzerinnen die Möglichkeit, detaillierte Informationen zu erkunden.

Mit den Erkenntnissen aus den qualitativen Interviews ergibt sich folgender Anforderungskatalog:

Nicht funktionale Anforderungen

- Lokales, Open Source Werkzeug
  Die Netzwerkgrafik muss mittels eines Open Source Werkzeug generiert werden können. Die Anwendung soll lokal anwendbar sein. Die Netzwerkgrafik wird vorzugsweise mittels Python erstellt.
- Sicherheit
  Mithilfe der Netzwerkgrafiken werden Kundendaten visualisiert. Aus diesem Grund muss beim Werkzeug die Datensicherheit gewährleistet sein. Somit muss die Python-Bibliothek Herkunft vertrauenswürdig sein.
- Datenimport (CSV / JSON / Python File)
  Für Workshops, in welchen zum Beispiel die IT-Netzwerke von Kunden abgebildet werden, werden die Informationen oft mittels CSV angeliefert. Andere Daten werden in Python synthetisch hergestellt. Die Integration von verschiedenen Dateitypen muss deswegen gegeben sein.
- Netzwerkgrafikexport (PDF, SVG)
  Die Netzwerkgrafiken werden den Kunden zur Verfügung gestellt. Dies meist per Mail oder über die Confluence™ (Wiki-Software). Aus diesem Grund sollten die Grafiken per PDF oder SVG exportierbar sein.

Funktionale Anforderungen

Gestaltung:

- Beschriftungsoptionen der Knoten
  In der Netzwerkgrafik sollen die gesamten Knotenbezeichnungen angezeigt werden. Zumindest mittels Hover-Funktion. Zudem werden lange Namen unterstützt. Optimal: Textfeld mit Formatierungsfunktionen.
- Knotengestaltung mit Farben
  Die Knoten in der Netzwerkgrafik müssen mindestens mittels Farbe unterscheidbar sein.
- Knotengestaltung mit Formen
  Vorzugsweise sind die Knoten in der Netzwerkgrafik mittels Formen unterscheidbar (somit ist die Differenzierbarkeit für farbenblinde Personen sicherlich gegeben).
- Kantengestaltung
  Die Kanten in der Netzwerkgrafik sollten mittels Dicke (Beziehungsstärke) unterschieden werden können. Vorzugsweise können Beziehungsrichtungen mittels Pfeile abgebildet werden (gerichtete Graphen). Und die Kantenfarbe (Beziehungsart) sollte verändert werden können.
- Corporate Design Integration Möglichkeit
  Es soll ein Farbtemplate integrierbar sein.

Anwendbarkeit (Interaktivität):

- Zoom Möglichkeit
  In der Netzwerkgrafik sollte sich auf einen bestimmten Bereich konzentriert werden können, um diesen genauer zu besprechen.
- Suchoption
  In der Netzwerkgrafik soll nach bestimmten Knoten gesucht werden können. Möglichkeit: Kann auch über Bowser funktionieren.
- Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit
  Für eine bessere Übersichtlichkeit in grossen Netzwerkgrafiken, sollen die Kanten, Knoten (nur Knoten mit gewissen Eigenschaften) ein- und ausgeblendet werden können.
- Übersichtlich Darstellung der Inhalte
  Optionen zur Beeinflussung der Darstellung durch Layouts. Wie Auswahl aus verschiedenen Layout Algorithmen. Manuelle Platzierung und Anpassung von Knoten.
- Kommentarmöglichkeit
  In der Netzwerkgrafik sollen einzelne Bereiche kommentiert werden können. Möglichkeit: Objekterstellung und Verbindung.
- Inputs entgegennehmen
  Beispielsweise Kanten und Knoten direkt erstellen, aktualisieren und löschen.

3.2 Auswahl und Beurteilung der Bibliotheken

Die Suche nach geeigneten Tools ergab elf potenziellen Python-Bibliotheken: Bokeh, Cytoscape (Dash), PyGraphviz, Holoviews, Igarph, NetworkX, Plotly, PyGraphistry, PyViz, Seaborn, Yfiles-jupiter-graphs. Diese Bibliotheken werden nachfolgend in alphabetischer Reihenfolge beurteilt.

3.2.1 Bokeh

Nachfolgende Beurteilungen wurden anhand der Bibliothek Dokumentation (Bokeh Website, 2023) und dem Quellcode, sowie den Beispielen in Github (Bokeh on Github, 2023) erstellt.

Tabelle 2: Bokeh Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
Bokeh steht unter der Open Source Lizenz BSD-3 [1]und ist über GitHub frei verfügbar. Wird durch «`pip install Bokeh`» in Python installiert und lokal im Python-Code verwendet.	
Sicherheit / Vertrauenswürdige Quelle	Erfüllt
Gemäss dem Cybersicherheitsunternehmen Snyk (synkAdvisor, 2023a), gab es keine bekannte Sicherheitsprobleme mit Bokeh. Für alles andere als die Dokumentation nutzt Bokeh das kostenlose Angebot von «1Password» für Open Source Projekte zur sicheren Speicherung und Verwaltung aller Anmeldedaten. Zusätzlich härtet Bokeh die CI-Pipelines mit GitGuardian ab.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Es besteht die Möglichkeit, entweder manuell Knoten und Kanten der Grafik anzulegen oder Dateiformate wie CSV, JSON zu importieren (und auf Basis DataFrame Grafik zu erstellen).	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Die Netzwerkgrafiken können als: HTML, PNG, SVG gespeichert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Teilweise erfüllt
Das Knoten-Label wird als ganzer Text angezeigt und geht deswegen aus dem Rand der Knoten aus. Im Titel können weitere Details zu den Knoten gespeichert werden. Diese werden bei Aktivierung der Hover-Interaktion angezeigt.	
Knotengestaltung mit Farben	Erfüllt
Die Knoten können in unterschiedlicher Farbe dargestellt werden. Die Farben der Knoten sind mittels «`fill_color`»-Attributs veränderbar oder mithilfe eine Gegebene Palette von Bokeh.	
Knotengestaltung mit Formen	Erfüllt
Grösse und Form der Knoten sind veränderbar. Die Grösse ist auch anhand einer Skala, welche auf den Daten basiert (zum Beispiel Anzahl ausgehende Kanten), anpassbar.	
Kantengestaltung	Erfüllt
Die Dicke und die Farbe lassen sich anpassen. Genauso ist die Darstellung der Kanten in Form eines Pfeiles erlaubt. Und auch eine Beschriftung der Kante ist möglich.	
Corporate Design Integration Möglichkeit	Erfüllt
Farb-Templates sind kann man in Bokeh manuell festlegen.	

[1] Der Urtyp der Lizenz stammt von der University of California, Berkeley (UCB), worauf das Akronym BSD hinweist: Berkeley Software Distribution. Software unter BSD-Lizenz darf frei verwendet werden. Es ist erlaubt, sie zu kopieren, zu verändern und zu verbreiten. Einzige Bedingung ist, dass der Copyright-Vermerk des ursprünglichen Programms nicht entfernt werden darf.

Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Es besteht die Möglichkeit des Zoomens in der Grafik. Auch Pan Funktion (ziehen mit der Maus) ist verfügbar.	
Suchoption	Teilweise erfüllt
Es ist nicht möglich, direkt im Bild nach einem Knoten zu suchen, nur mithilfe von Zoom- oder Pan Funktion.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Teilweise erfüllt
Es besteht die Möglichkeit andere Knoten (mit Tap Funktion oder Interaktive Legende) ausgrauen, aber nicht komplett ausblenden. Es ist auch möglich, mithilfe des Maus einen Knoten und seine Kanten herauszustellen.	
Übersichtlich Darstellung der Inhalte	Teilweise erfüllt
Um die Anordnung der Knoten zu steuern, sind verschiedene Layout-Algorithmen verwendbar. Das Ziehen mit der Maus ist aber nicht möglich.	
Kommentarmöglichkeit	Nicht erfüllt
Es ist nicht möglich, einen Kommentar direkt in der Benutzeroberfläche einzufügen. (Mahlfunktion möglich, wird aber nicht gespeichert)	
Inputs entgegennehmen	Teilweise erfüllt
Nur durch die Zufügung von zusätzlichen Objekten (Knoten). Die direkte Manipulation der Netzwerkgrafik ist jedoch nicht möglich.	

3.2.2 Cytoscape (Dash)

Cytoscape ist eine Python Bibliothek, welche stark in Dash Layouts integriert ist. Somit ist sie einfach mit der umfangreichen Sammlung an Dash-Komponenten kombinierbar.

Nachfolgende Beurteilungen wurden anhand der Bibliothek Dokumentation (Dash Cyposcape, 2019) erstellt.

Tabelle 3: Cytoscape (Dash) Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
Cytoscape ist eine Open Source Bibliothek. Der Quellcode kann eingesehen, bearbeitet und frei verwendet werden.	
Sicherheit / Vertrauenswürdige Quelle	Nicht erfüllt
Gemäss dem Cybersicherheitsunternehmen Snyk (synkAdvisor, 2023b), erfüllt die Bibliothek zwar alle Sicherheitsstandards aber sie wurde vor über zwei Jahre das letzte Mal aktualisiert und wird somit momentan nicht mehr aktiv weiterentwickelt.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Der direkte Datenimport von Formaten wie CSV, JSON und Python ist möglich. Diese können einfach in das Cytoscape-Format umgewandelt werden.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Verschiedene Exportmöglichkeiten werden unterstützt (zum Beispiel PDF, SVG, PNG).	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Teilweise erfüllt
Die Beschriftung der Knoten und das Hinterlegen von weiteren Informationen ist möglich. Jedoch kann die Beschriftung nur ausserhalb des Knoten angezeigt werden.	
Knotengestaltung mit Farben	Erfüllt
Die Knoten können in unterschiedlicher Farbe dargestellt werden.	
Knotengestaltung mit Formen	Erfüllt
Es stehen verschiedene Formen (Kreise, Quadrate, Ellipsen etc.) zur Verfügung.	
Kantengestaltung	Erfüllt
Um die Kanten anzupassen, stehen verschiedene Linientypen, Pfeilformen und Farben zur Auswahl.	
Corporate Design Integration Möglichkeit	nicht erfüllt
Cytoscape bietet keine spezifische Integration von Templates. Es kann ein CSS integriert oder einzelne Elemente angepasst werden.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Es besteht die Möglichkeit des Zoomens in der Grafik.	
Suchoption	Erfüllt
Die Möglichkeit, nach bestimmten Knoten, Kanten oder Attributen zu suchen, besteht.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Erfüllt
Elemente können basierend auf Attribute oder manuell ausgeblendet werden.	
Übersichtlich Darstellung der Inhalte	Erfüllt
Eine Vielzahl von Layout-Algorithmen ist gegeben. Zudem können die Knoten manuell verschoben werden.	
Kommentarmöglichkeit	nicht erfüllt
Es ist nicht möglich, einen Kommentar direkt in der Benutzeroberfläche einzufügen. Es könnte jedoch auf dem Knoten ein Kommentar direkt im Code hinterlegt werden.	
Inputs entgegennehmen	Teilweise erfüllt
Knoten können verschoben und Kanten verbunden werden.	

3.2.3 Holoviews

Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (Holoviews Website, 2023), dem Quellcode und GitHub Beispielen (Holoviews on GitHub, 2023) erstellt.

Tabelle 4: Holoviews Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
Holoviews ist eine Open-Source-Bibliothek und wird lokal in Python verwendet. Die Bibliothek kann auf GitHub (https://github.com/holoviz/holoviewsgefunden werden.	
Sicherheit / Vertrauenswürdige Quelle	Teilweise erfüllt
Holoviews ist eine etablierte Bibliothek, die von einer engagierten Entwickler-Community unterstützt wird. Obwohl es keine bekannten Sicherheitsbedenken gibt, besteht bei der Verwendung von Javascript-Bibliotheken im Browser für die Visualisierung ein potenzielles Risiko (synkAdvisor, 2023c).	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Holoviews ermöglicht den Import von Daten aus verschiedenen Formaten wie CSV, JSON und Python-Objekten.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Mit Holoviews können Netzwerkgrafiken als PDF, SVG oder andere Formate exportiert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Mit Holoviews können Knoten mit Beschriftungen versehen werden. Die Beschriftungen werden entweder als Text innerhalb der Knoten oder ausserhalb angezeigt.	
Knotengestaltung mit Farben	Teilweise erfüllt
Einzelne Farben können individuell festgelegt werden oder über Farbpaletten. Dies ermöglicht eine visuelle Unterscheidung der Knoten basierend auf bestimmten Attributen. Allerdings ist die Farbauswahl eingeschränkt.	
Knotengestaltung mit Formen	Teilweise erfüllt
Die Grösse und Form der Knoten können mit Holoviews angepasst werden. Verschiedene Formen wie Kreise, Quadrate oder Symbole können verwendet werden. Die verfügbaren Formoptionen sind allerdings beschränkt.	
Kantengestaltung	Erfüllt
Mit Holoviews können verschiedene Eigenschaften der Kanten angepasst werden. Bspw. die Dicke, Farbe oder der Stil der Kanten. Die Kanten können auch mit Pfeilen und Beschriftungen ergänzt werden.	
Corporate Design Integration Möglichkeit	Nicht erfüllt
Holoviews bietet keine spezifische Unterstützung für die Integration von Corporate Design. Durch die Anpassung von Farbe und Stil kann das gewünschte Corporate Design manuell erzeugt werden.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Holoviews bietet Zoom-Funktionalität für die Netzwerkgrafiken an, um Details genauer zu betrachten oder einen besseren Überblick über die gesamte Grafik zu erhalten.	
Suchoption	Teilweise erfüllt
Holoviews ermöglicht die Implementierung von Suchfunktionen, um nach bestimmten Knoten oder Kanten zu suchen. Sie deckt aber nicht alle Suchfunktionen ab. Die Hinzunahme weiterer Bibliotheken ist erforderlich.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Teilweise erfüllt
Mit Holoviews können bei einfachen Grafiken Knoten und Kanten ein- und ausgeblendet werden. Für komplexere Grafiken müssen allerdings weitere Bibliotheken implementiert werden.	
Übersichtlich Darstellung der Inhalte	Erfüllt
Holoviews bietet verschiedene Layout-Algorithmen zur Steuerung der Anordnung der Knoten.	
Kommentarmöglichkeit	Teilweise erfüllt
Mit Holoviews können Kommentare oder Anmerkungen als separate Elemente zur Netzwerkgrafik hinzugefügt werden. Eine Kommentarfunktion über die Grafik ist jedoch nicht möglich.	
Inputs entgegennehmen	Teilweise erfüllt
Eine Interaktion über die Grafik scheint nur bedingt möglich zu sein.	

3.2.4 igraph

igraph ist eine leistungsstarke Python-Bibliothek zur Erzeugung und Manipulation von Graphen. Damit lassen sich sowohl statische als auch dynamische Graphen erzeugen. Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (igraph Website, 2023), dem Quellcode und GitHub Beispielen (igraph on GitHub, 2023) erstellt.

Tabelle 5: igraph Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
igraph ist eine Open Source Bibliothek und wird lokal in Python verwendet.	
Sicherheit / Vertrauenswürdige Quelle	Teilweise erfüllt
igraph ist eine etablierte Bibliothek, welche aktiv weiterentwickelt wird. Es gibt keine bekannten Sicherheitsbedenken (synkAdvisor, 2023d). Dennoch benötigt man für die Visualisierung im Browser Javascript- Bibliotheken, welche als externe Quellen gefährdend wirken können.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
igraph ermöglicht den Import von Daten aus verschiedenen Formaten wie CSV, JSON und Python-Objekten.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Mit igraph können Netzwerkgrafiken als PDF, SVG oder andere Formate exportiert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Die Beschriftungen werden entweder als Text innerhalb der Knoten oder ausserhalb angezeigt. Die Position und das Aussehen der Beschriftungen können angepasst werden.	
Knotengestaltung mit Farben	Erfüllt
Man kann einzelne Farben für jeden Knoten festlegen oder Farbpaletten verwenden. Dies ermöglicht eine visuelle Unterscheidung der Knoten basierend auf bestimmten Attributen oder Eigenschaften.	
Knotengestaltung mit Formen	Erfüllt
Die Grösse und Form der Knoten können in igraph angepasst werden. Es können verschiedene Formen wie Kreise, Quadrate oder Symbole verwenden werden.	
Kantengestaltung	Erfüllt
Mit igraph können verschiedene Eigenschaften der Kanten angepasst werden. Bspw. die Dicke, Farbe oder der Stil der Kanten. Man kann die Kanten auch mit Pfeilen und Beschriftungen ergänzen.	
Corporate Design Integration Möglichkeit	Nicht erfüllt
Die Bibliothek bietet ausschliesslich manuelle Anpassungsmöglichkeiten für Farbe und Stil, womit das gewünschte Corporate Design nur manuell erzeugt werden kann.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
igraph bietet Zoom-Funktionalität für die Netzwerkgrafiken an.	
Suchoption	Erfüllt
Mit igraph können Suchfunktionen implementiert werden, mit welchen nach bestimmten Knoten oder Kanten im Netzwerk gesucht werden kann. Damit können bestimmte Elemente schnell gefunden werden.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Erfüllt
Mit igraph können Knoten und Kanten ein- und ausgeblendet werden.	
Übersichtlich Darstellung der Inhalte	Erfüllt
igraph bietet verschiedene Layout-Algorithmen, mit der die Anordnung der Knoten gesteuert werden kann. Dadurch kann die Netzwerkgrafik übersichtlich und optisch ansprechend gestaltet werden.	
Kommentarmöglichkeit	Teilweise erfüllt
Mit igraph werden Kommentare oder Anmerkungen als separate Elemente zur Netzwerkgrafik hinzugefügt.	
Inputs entgegennehmen	Teilweise erfüllt
igraph ermöglicht die direkte Manipulation der Netzwerkgrafik durch Hinzufügen oder Entfernen von Knoten und Kanten. Eine Interaktion über die Grafik scheint aber nur bedingt möglich zu sein.	

3.2.5 NetworkX

NetworkX ist ein Python-Paket für die Erstellung, Bearbeitung und Untersuchung komplexer Netzwerke, gibt aber mit sich allein keine Interaktivität Möglichkeit (Statische Bilder). Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (NetworkX Website, 2023), dem Quellcode und den Beispielen in GitHub (NetworkX on GitHub, 2023) erstellt.

Tabelle 6: NetworkX Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
NetworkX ist eine Open-Source-Bibliothek und wird lokal in Python verwendet. Die Bibliothek kann auf Github gefunden werden. Wird durch «`pip install networkx`» in Python installiert und lokal im Python-Code verwendet.	
Sicherheit / Vertrauenswürdige Quelle	Erfüllt
NetworkX ist eine populäre und gut etablierte Bibliothek, deren Quellcode auf GitHub gehostet wird. Sie wird von einer engagierten Gemeinschaft von Entwicklern und Programmierern unterstützt. Es sind keine bekannten Sicherheitsbedenken in Bezug auf NetworkX bekannt (synkAdvisor, 2023e).	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
NetworkX ermöglicht den Import von Daten aus verschiedenen Formaten wie CSV, JSON und Python-Objekten.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
NetworkX bietet die Möglichkeit, Netzwerkgrafiken in verschiedenen Formaten wie PDF, SVG oder anderen zu exportieren (zum Beispiel `plt.savefig("grapg.pdf", format="pdf")` und sie für zukünftige Bearbeitungen oder die Integration in andere Dokumente zu verwenden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Mit hilfe von `"draw_networkx_labels()"`-Funktion in NetworkX können Knoten beschriften werden . Die Beschriftungen werden entweder als Text innerhalb oder ausserhalb der Knoten angezeigt.	
Knotengestaltung mit Farben	Erfüllt
Mit NetworkX hat man die Möglichkeit die Farben der Knoten anzupassen (node color-Attribute). Man kann einzelne Farben für jeden Knoten festlegen oder Farbpaletten verwenden. Dadurch kann man die Knoten visuel aufgrund bestimmte Atributte oder Eigenschafte unterscheiden.	
Knotengestaltung mit Formen	Erfüllt
Die Grösse und Form der Knoten können in NetworkX angepasst werden. Man kann verschiedene Formen wie Kreise, Quadrate oder Ellipse verwenden, um Knoten darzustellen. Die Grösse der Knoten kann ebenfalls angepasst werden.	
Kantengestaltung	Erfüllt
NetworkX bietet vielfältige Möglichkeiten, um verschiedene Eigenschaften der Kanten anzupassen. Man kann die Dicke, Farbe und den Stil der Kanten ändern. Darüber hinaus kann man den Kanten Pfeile hinzufügen und sie zu beschriften.	
Corporate Design Integration Möglichkeit	Erfüllt
NetworkX bietet manuelle Anpassungsmöglichkeiten für Farbe, womit die Grafik nach Corporate Design manuell erstellt werden kann.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Nicht erfüllt
NetworkX bietet keine Zoom-Funktionalität für die Netzwerkgrafiken an.	
Suchoption	Teilweise erfüllt
Mit NetworkX können Suchfunktionen implementiert werden, mit welchen nach bestimmten Knoten oder Kanten im Netzwerk gesucht werden kann. Bibliothek selbst hat aber keine Suchoption.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Nicht erfüllt
Mit NetworkX ist nicht möglich Knoten und Kanten interaktiv ein- oder auszublenden.	
Übersichtlich Darstellung der Inhalte	Teilweise erfüllt

NetworkX bietet verschiedene Layout-Algorithmen wie: "bipartite_layout", "circular_layout", "kamada_kawai_layout", "random_layout", "rescale_layout", "rescale_layout_dict", "shell_layout", "spring_layout", "spectral_layout", "planar_layout", "fruchterman_reingold_layout", "spiral_layout" mit der Anordnung der Knoten gesteuert werden kann. Manuelle Anpassung ist aber nicht möglich.	
Kommentarmöglichkeit	Nicht erfüllt
Eine Kommentarfunktion über die Grafik ist nicht möglich.	
Inputs entgegennehmen	Nicht erfüllt
In NetworkX ist es nicht möglich, die Netzwerkgrafik direkt zu manipulieren, indem man Knoten und Kanten im Ausgabebild hinzufügt oder entfernt.	

3.2.6 Plotly

Plotly ist eine Python-Grafikbibliothek, mit welcher sich sowohl statische als auch dynamische Visualisierungen erzeugen lassen. Für die Erstellung einer Netzwerkgrafik ist die Bibliothek jedoch nur in Kombination mit NetworkX oder Dash geeignet.

Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (Plotly Website, 2023), dem Quellcode und GitHub Beispielen (Plotly on GitHub, 2023) erstellt.

Tabelle 7: Plotly Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
Plotly ist eine interaktive, browserbasierte, Open-Source-Bibliothek und wird lokal in Python verwendet. Die Bibliothek kann auf GitHub gefunden werden. (`import plotly.graph_objects as go`)	
Sicherheit / Vertrauenswürdige Quelle	Erfüllt
Nach Snyk (synkAdvisor, 2023f) gibt es keine bekannte Sicherheitsprobleme. Code und Dokumentation sind von Copyright 2019 Plotly, Inc. unter der MIT-Lizenz veröffentlicht.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Plotly ermöglicht den Import von Daten aus verschiedenen Formaten wie CSV, JSON und Python-Objekten.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Mit Plotly können Netzwerkgrafiken als PDF, SVG, PNG, JPEG (`fig.write_image("images/fig1.pdf")`, `fig.write_image("images/fig1.png")` oder andere Formate exportiert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Teilweise erfüllt
Mit Plotly können Knoten beschriftet werden, aber nicht beliebig. Im Titel können weitere Details zu den Knoten gespeichert werden. Diese werden bei Aktivierung der Hover-Interaktion angezeigt.	
Knotengestaltung mit Farben	Erfüllt
Mit Plotly können die Farben der Knoten angepasst werden. Entweder werden Farben für jeden Knoten festlegen oder eine Farbpaletten verwenden.	
Knotengestaltung mit Formen	Erfüllt
Die Grösse und Form der Knoten können in Plotly angepasst werden. Verschiedene Formen wie Kreise oder Quadrate können verwendet werden.	
Kantengestaltung	Erfüllt
Mit Plotly können verschiedene Eigenschaften der Kanten (Dicke, Farbe oder Beschriftung) angepasst werden.	
Corporate Design Integration Möglichkeit	Teilweise erfüllt
Die Bibliothek bietet ausschliesslich manuelle Anpassungsmöglichkeiten für Farbe und Stil, womit das gewünschte Corporate Design manuell erzeugt werden kann.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Plotly bietet Zoom-Funktionalität für die Netzwerkgrafiken an.	
Suchoption	Nicht erfüllt
Es ist nicht möglich, direkt im Bild nach einem Knoten zu suchen.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Nicht erfüllt
Plotly bietet keine Möglichkeit, die Knoten und die Kanten ein- oder auszublenden.	
Übersichtlich Darstellung der Inhalte	Teilweise erfüllt
Plotly selbst bietet keine Layout-Algorithmen, sie nutzt die Layouts von NetworkX.	
Kommentarmöglichkeit	Teilweise erfüllt
Mit Plotly können Kommentare oder Anmerkungen als separate Elemente (Knoten) zur Netzwerkgrafik hinzugefügt werden. Eine Kommentarfunktion über die Grafik ist jedoch nicht möglich.	
Inputs entgegennehmen	Teilweise erfüllt
Plotly ermöglicht die direkte Manipulation der Netzwerkgrafik durch Hinzufügen oder Entfernen von Knoten und Kanten. Eine Interaktion über die Grafik ist aber nicht möglich.	

3.2.7 PyGraphistry

PyGraphistry ist eine Bibliothek zur Visualisierung von Graphen, die auf Graphistry (Graphistry Homepage, 2023) basiert. Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (PyGraphistry on GitHub, 2023).

Tabelle 8: PyGraphistry Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Nicht erfüllt
PyGraphistry ist keine Open Source Bibliothek. Login zu der Graphistry GPU ist erforderlich.	
Sicherheit / Vertrauenswürdige Quelle	Teilweise erfüllt
Die Sicherheit der PyGraphistry ist von der zugrundeliegenden Software graphistry abhängig (synkAdvisor, 2023g). Die Bibliothek wird aktiv von rund dreissig Personen weiterentwickelt und es erscheinen regelmässig neue Releases.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Daten können auf unterschiedliche Weise integriert werden (CSV, JSON zum Beispiel über Pandas oder ein direkter Zugriff auf Datenbanken `pygraphistry.edges(file_path)`).	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Mit Plotly können Netzwerkgrafiken als PDF, SVG oder andere Formate exportiert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Knoten können individuell beschriftet werden.	
Knotengestaltung mit Farben	Erfüllt
Farben können den Knoten zugeteilt werden (`node_color=`).	
Knotengestaltung mit Formen	Erfüllt
Es besteht die Option, die Form der Knoten individuell anzupassen (`node_shape=`).	
Kantengestaltung	Erfüllt
Farbe, Stärke und Darstellung der Kanten können angepasst werden.	
Corporate Design Integration Möglichkeit	Teilweise erfüllt
Individuelle Elemente können angepasst werden. Eine Implementierung eines Corporate Designs an einer Stelle ist nicht möglich.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Zoom Möglichkeit zur genaueren Betrachtung einzelner Elemente ist gegeben.	
Suchoption	Erfüllt
Eine Suchfunktion kann implementiert werden. Es kann basierend auf bestimmten Attributen oder anderen Kriterien nach Knoten oder Kanten gesucht werden.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Erfüllt
Es ist möglich, bestimmte Elemente basierend auf Attributen oder anderen Kriterien ein- oder auszublenden.	
Übersichtlich Darstellung der Inhalte	Erfüllt
PyGraphistry bietet integrierte Layout-Algorithmen zur automatischen Anordnung von Knoten und Kanten in der Netzwerkgrafik.	
Kommentarmöglichkeit	Nicht erfüllt
Es können nur Kommentare zu den einzelnen Knoten als Metadaten hinzugefügt werden. Das Hinzufügen eines Kommentars zu einem Bereich ist nicht möglich.	
Inputs entgegennehmen	Teilweise erfüllt
PyGraphistry ermöglicht die direkte Manipulation der Netzwerkgrafik durch Hinzufügen oder Entfernen von Knoten und Kanten. Eine Interaktion über die Grafik ist aber nicht möglich.	

3.2.8 PyGraphviz

Graphviz eine C-basierte Bibliothek zur Graphenvisualisierung. Mit dieser kann über PyGraphiz direkt aus Python interagiert werden. Nachfolgende Beurteilungen wurden anhand der Bibliothek Dokumentation (Graphviz,2004) erstellt.

Tabelle 9: PyGrahviz Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
PyGraphviz ist ein Open Source Werkzeug und steht unter der MIT-Lizenz zur Verfügung und ist das Python Interface zu Graphviz.	
Sicherheit / Vertrauenswürdige Quelle	Erfüllt
Nachdem Cybersicherheitsunternehmen Snyk (synkAdvisor, 2023h), erfüllt die Bibliothek alle Sicherheitsstandards. Rund 50 Personen entwickeln die Bibliothek weiter, wodurch regelmässige Updates veröffentlicht werden.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Nicht erfüllt
Der direkte Import ist nicht möglich. Knoten und Kanten können programmiert werden.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
PyGraphviz ermöglicht den Export von erstellten Graphen in verschiedene Formate.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Teilweise erfüllt
Beschriftungen können zu Knoten hinzugefügt werden, indem ein Attribut Element hinzugefügt wird, jedoch können keine weiteren Informationen hinterlegt werden.	
Knotengestaltung mit Farben	Erfüllt
Farben können den Knoten über ein Attribut zugewiesen werden.	
Knotengestaltung mit Formen	Erfüllt
Formen werden den Knoten mittels Attribut zugewiesen.	
Kantengestaltung	Erfüllt
Die Anpassung der Darstellung von Kanten, einschliesslich Farbe, Linienstärke und -art sind möglich.	
Corporate Design Integration Möglichkeit	Teilweise erfüllt
PyGraphiz bietet keine spezifische Unterstützung für die Integration von Corporate Design. Manuelle Anpassungsmöglichkeiten für Farbe und Stil sind umsetzbar, womit das gewünschte Corporate Design manuell erzeugt werden kann.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Nicht erfüllt
PyGraphiz bietet keine Zoommöglichkeit.	
Suchoption	Nicht erfüllt
PyGraphiz bietet keine integrierte Suchoption. Dies müsste mit einer separaten Logik implementiert werden.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Nicht erfüllt
Die Darstellung von Knoten oder Kanten kann nur programmgesteuert anpassen werden.	
Übersichtlich Darstellung der Inhalte	Erfüllt
Mit integrierten Algorithmen kann das Layout der Grafiken angepasst werden.	
Kommentarmöglichkeit	Nicht erfüllt
PyGraphiz selber bietet keine Kommentarfunktion an.	
Inputs entgegennehmen	Teilweise erfüllt
PyGraphviz bietet APIs zur Programmierung und Manipulation der erzeugten Grafiken. So können Knoten und Kanten hinzufügen, entfernen oder anpassen werden.	

3.2.9 PyVis

PyVis ist das Python Interface von der JavaScript Bibliothek vis.js. Die Grafiken sind in eine Webapplikation integrierbar. Nachfolgende Beurteilungen wurden anhand der Bibliothek Dokumentation (PyVis Dokumentation, 2016) und Github (PyVis on Github, 2018) erstellt.

Tabelle 10: PyVis Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
PyVis ist eine Open Source Bibliothek, welche auf GitHub gehostet wird (https://github.com/WestHealth/pyvis) und wird lokal im Python-Code verwendet. PyVis ist unter der BSD 3 Clause Lizenz veröffentlicht.	
Sicherheit / Vertrauenswürdige Quelle	Teilweise erfüllt
Gemäss dem Cybersicherheitsunternehmen Snyk (synkAdvisor, 2023i), ist die Sicherheit von der zugrundeliegenden JavaScript-Bibliothek abhängig. Diese ist eine gut etablierte Bibliothek. Die Bibliothek wird jedoch aktiv von rund dreissig Personen weiterentwickelt und es erscheinen regelmässig neue Releases. Der Quellcode ist auf GitHub zu finden.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Manuelle Knoten-Anlegung oder Dateiformate wie CSV, JSON und Pandas DataFrame importiert.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Die Netzwerkgrafiken können als HTML, PNG, SVG exportiert und gespeichert.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Das Knoten-Label wird entweder im oder unterhalb des Knoten angezeigt (je nach Knotenform). Im Titel können weitere Details zu den Knoten gespeichert werden. Diese werden über die Hover-Interaktion angezeigt.	
Knotengestaltung mit Farben	Erfüllt
Die Knoten können in unterschiedlicher Farbe dargestellt werden.	
Knotengestaltung mit Formen	Erfüllt
Grösse und Form der Knoten sind veränderbar. Die Grösse ist auch dynamisch, anhand einer Skala, anpassbar.	
Kantengestaltung	Erfüllt
Die Beschriftung, Stärke und der Stil lassen sich beeinflussen. Die Graphen können gerichtet werden.	
Corporate Design Integration Möglichkeit	Erfüllt
Farbtemplates sind in PyVis integrierbar.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Es besteht die Möglichkeit des Zoomens in der Grafik oder mittels Mausrad-Zooms.	
Suchoption	Erfüllt
Ein Filter Menü, in welchem nach bestimmten Knoten oder Kanten gesucht werden kann, ist integrierbar.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Erfüllt
Entweder kann ein Knoten direkt im Code ausgeblendet werden oder dies geschieht über das Suchfeld.	
Übersichtlich Darstellung der Inhalte	Erfüllt
Um die Anordnung der Knoten zu steuern, sind verschiedene Layout-Algorithmen verwendbar. Als Standard ist der Fruchtermann-Reingold-Algorithmus hinterlegt. Die Kräfte zwischen den Knoten können zudem mittels der Physics-Parameter angepasst werden.Oder die Knoten werden manuell per Maus einzeln angeordnet.	
Kommentarmöglichkeit	teilweise erfüllt
Es ist nicht möglich, einen Kommentar direkt in der Benutzeroberfläche einzufügen. Es besteht jedoch die Möglichkeit, Kommentare oder Anmerkungen als separate Elemente zur Netzwerkgrafik hinzuzufügen.	
Inputs entgegennehmen	Teilweise erfüllt
Über den Code werden zusätzliche Objekte erstellt. Die direkte Manipulation der Netzwerkgrafik ist jedoch nicht möglich.	

3.2.10 Seaborn

Seaborn ist eine Python-Bibliothek zur Visualisierung von Daten, welche auf Matplotlib aufbaut. Nachfolgende Beurteilungen wurden anhand der Bibliotheken-Dokumentation (Seaborn Website, 2023), dem Quellcode und in GitHub (Seaborn on GitHub, 2023) erstellt.

Tabelle 11: Seaborn Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Erfüllt
Seaborn ist eine Open-Source-Bibliothek und wird lokal in Python verwendet. Die Bibliothek kann auf GitHub (https://github.com/mwaskom/seaborn) gefunden werden.	
Sicherheit / Vertrauenswürdige Quelle	Erfüllt
Seaborn ist eine etablierte und verbreitete Bibliothek. Der Quellcode wird auf GitHub gehostet und die Bibliothek wird von einer engagierten Community von Entwicklern und Programmieren unterstützt. Es gibt keine bekannten Sicherheitsbedenken in Bezug auf Seaborn (synkAdvisor, 2023j)	
Direkter Datenimport (CSV / JSON / Python Objekte)	Erfüllt
Seaborn ermöglicht den Import von Daten aus verschiedenen Formaten wie CSV, JSON und Python-Objekten, was das Laden von Netzwerkdaten aus verschiedenen Quellen erleichtert.	
Netzwerkgrafikexport (PDF, SVG)	Erfüllt
Mit Seaborn können Netzwerkgrafiken als PDF, SVG oder andere Formate exportiert werden. Damit können die erzeugten Grafiken für eine spätere Weiterbearbeitung (zwischen)gespeichert werden oder in andere Dokumente integriert werden.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Nicht erfüllt
Seaborn ist auf die Visualisierung statistischer Daten spezialisiert und unterstützt keine spezifischen Funktionen zur Beschriftung von Knoten in Netzwerkgrafiken. Um Knoten beschriften zu können, müssen Bibliotheken wie bspw. NetworkX oder Graph-tool verwendet werden.	
Knotengestaltung mit Farben	Nicht erfüllt
Seaborn bietet begrenzte Möglichkeiten zur Farbanpassung von Knoten in Netzwerkgrafiken. Für spezifische Farbanpassungen müssen allerdings weitere Bibliotheken wie NetworkX oder Bokeh implementiert werden.	
Knotengestaltung mit Formen	Nicht erfüllt
Seaborn bietet keine spezifischen Funktionen zur Anpassung der Formen von Knoten in Netzwerkgrafiken. Dafür müssten weitere Bibliotheken importiert werden.	
Kantengestaltung	Nicht erfüllt
Seaborn bietet keine spezifischen Funktionen zur Anpassung von Kanten in Netzwerkgrafiken. Für erweiterte Gestaltungsoptionen müssen andere Bibliotheken importiert werden.	
Corporate Design Integration Möglichkeit	Nicht erfüllt
Seaborn bietet keine spezifische Unterstützung für die Integration von Corporate Design. Die Bibliothek bietet jedoch manuelle Anpassungsmöglichkeiten für Farbe und Stil, womit das gewünschte Corporate Design manuell erzeugt werden kann.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Nicht erfüllt
Seaborn bietet keine integrierte Zoom-Funktion für Netzwerkgrafiken. Dafür müssten weitere Bibliotheken importiert werden.	
Suchoption	Nicht erfüllt
Seaborn bietet keine spezifischen Funktionen zur Implementierung von Suchfunktionen an. Dafür müssen weitere Bibliotheken implementiert werden.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Teilweise erfüllt
Mit Seaborn können bei einfachen Grafiken Knoten und Kanten ein- und ausgeblendet werden, um bestimmte Teile des Graphen zu betonen oder zu verbergen. Dies erfordert jedoch manuelle Anpassungen des Codes. Für spezifische Funktionen müssen weitere Bibliotheken implementiert werden.	
Übersichtlich Darstellung der Inhalte	Teilweise erfüllt

Seaborn bietet verschiedene Layout-Algorithmen zur Steuerung der Anordnung der Knoten in der Netzwerkgrafik. Dadurch kann die Netzwerkgrafik übersichtlich und optisch ansprechend gestaltet werden. Dies beschränkt sich jedoch nur auf die Bereiche, welche in der Seaborn-Bibliothek enthalten sind. Für alle Bestandteile anderer Bibliotheken müssen wiederrum andere Funktionen der entsprechenden Bibliotheken importiert werden.	
Kommentarmöglichkeit	Nicht erfüllt
Seaborn bietet keine Möglichkeit zum Hinzufügen von Kommentaren. Dafür müssten weitere Bibliotheken importiert werden.	
Inputs entgegennehmen	Teilweise erfüllt
Seaborn ermöglicht sehr eingeschränkte Möglichkeiten zur direkten Interaktion mit Netzwerkgrafiken, wie das Hinzufügen oder Entfernen von Knoten oder Kanten. Für umfassendere Interaktionsmöglichkeiten müssten weitere Bibliotheken implementiert werden.	

3.2.11 yFiles-jupyter-graphs

yFiles ist eine Bibliothek, welche auf der Javascript Bibliothek yFiles basiert (yworks, 2023). Nachfolgende Beurteilungen wurden anhand der Dokumentation erstellt (yfiles-jupyter-graph, 2022).

Tabelle 12: yFiles-jupyter-graph Beurteilung

Tool allgemein	
Lokales, Open Source Werkzeug	Teilweise erfüllt
yFiles-jupyter-graphsist eine Open Source Bibliothek. Die Software und alle Kopien bleiben jedoch im Eigentum des Lizenzgebers (yWorks GmbH).	
Sicherheit / Vertrauenswürdige Quelle	Teilweise erfüllt
Gemäss Snyk (synkAdvisor, 2023k), erfüllt die Bibliothek zwar alle Sicherheitsstandards, aber es besteht eine Abhängigkeit zu yWorks. Zudem kann die Bibliothek Analysetechniken enthalten, welche Kundendaten zur Optimierung der Bibliothek nutzen. Und es entwickeln nur drei Personen diese Bibliothek weiter.	
Direkter Datenimport (CSV / JSON / Python Objekte)	Nicht erfüllt
Der direkte Datenimport von Formaten wie CSV, JSON und Python ist nicht möglich. Der Import könnte nur mit Hilfe einer anderen Bibliothek wie NetworkX gemacht werden.	
Netzwerkgrafikexport (PDF, SVG)	Teilweise erfüllt
Verschiedene Exportmöglichkeiten werden unterstützt, jedoch nur über einen vorausgehenden Export in yEd.	
An die konkrete Netzwerkgrafik	
Gestaltung	
Beschriftungsoptionen der Knoten	Erfüllt
Die Beschriftung der Knoten und das Hinterlegen von weiteren Informationen ist möglich.	
Knotengestaltung mit Farben	Erfüllt
Die Knoten können in unterschiedlicher Farbe dargestellt werden, indem die HEX Codes für die Knoten hinterlegt werden.	
Knotengestaltung mit Formen	Nicht erfüllt
Die Gestaltung der Formen ist momentan nicht möglich.	
Kantengestaltung	Teilweise erfüllt
Die Farbgestaltung der Kanten ist möglich. Die Gewichtung der Kanten jedoch nicht.	
Corporate Design Integration Möglichkeit	Nicht erfüllt
yFiles-jupyter-graphs bietet keine spezifische Integration von Templates.	
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit	Erfüllt
Es besteht die Möglichkeit des Zoomens in der Grafik. In einer Miniaturübersicht der Grafik wird zudem angezeigt, wo man sich in der Grafik befindet.	
Suchoption	Erfüllt
Die Möglichkeit, nach bestimmten Knoten, Kanten oder Attributen zu suchen, besteht über eine Freitextsuche im Balken rechts neben der Grafik.	
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit.	Erfüllt
Elemente können basierend auf Attribute ausgeblendet werden.	
Übersichtlich Darstellung der Inhalte	Teilweise erfüllt
Eine Vielzahl von Layout-Algorithmen ist gegeben. Jedoch können die Knoten nicht manuell verschoben werden.	
Kommentarmöglichkeit	Teilweise erfüllt
Es ist nicht möglich, einen Kommentar direkt in der Benutzeroberfläche einzufügen. Wenn die Grafik in yEd exportiert wird, können weitere Felder hinzugefügt werden, welche als Kommentar verwendet werden können.	
Inputs entgegennehmen	Teilweise erfüllt
Im Jupiter Notebook können Knoten nur über den Code integriert werden. Nach dem Export ins yEd besteht die Möglichkeit manuell Knoten hinzuzufügen.	

3.3 Ergebnis Nutzwertanalyse

Aus der Nutzwertanalyse geht hervor, dass sich die drei Python Bibliotheken PyVis, igraph und Bokeh am besten zur Erstellung von dynamischen Netzwerkgrafiken eignen sollten.

Anforderung	Gewichtung	Python Bibliotheken					
		PyVis		igraph		Bokeh	
		Beurt.	Wert	Beurt.	Wert	Beurt.	Wert
Nicht funktionale (Tool allgemein)							
Lokales, open Source Werkzeug	5	2	**10**	2	**10**	2	**10**
Sicherheit / Vertrauenswürdige Quelle	5	1	**5**	1	**5**	2	**10**
Direkter Datenimport (CSV / JSON / Python Objekte)	4	2	**8**	2	**8**	2	**8**
Netzwerkgrafikexport (PDF, SVG)	3	2	**6**	2	**6**	2	**6**
Gestaltung							
Beschriftungsoptionen der Knoten	4	2	**8**	2	**8**	1	**4**
Knotengestaltung mit Farben	5	2	**10**	2	**10**	2	**10**
Knotengestaltung mit Formen	4	2	**8**	2	**8**	2	**8**
Kantengestaltung	4	2	**8**	2	**8**	2	**8**
Corporate Design Integration Möglichkeit	2	2	**4**	1	**2**	2	**4**
Anwendbarkeit (Interaktivität)							
Zoom Möglichkeit	4	2	**8**	2	**8**	2	**8**
Suchoption	4	2	**8**	2	**8**	1	**4**
Ein- und Ausblenden von Objekten Möglichkeit	3	2	**6**	2	**6**	1	**3**
Übersichtlich Darstellung der Inhalte	4	2	**8**	2	**8**	1	**4**
Kommentarmöglichkeit	3	1	**3**	1	**3**	0	**0**
Inputs entgegennehmen	3	1	**3**	1	**3**	1	**3**
Nutzwertsumme			**103**		**101**		**90**
Rang			1		2		3

Abbildung 3: Drei bestbeurteilten Python-Bibliotheken.

Diese werden nachfolgend mit Hilfe des von der itopia AG zur Verfügung gestellten Datensatzes getestet. Im vorliegenden Fall erscheint dies umso wichtiger, da die erst- und zweitplatzierte Bibliothek nur zwei Punkte auseinanderliegen.

Die weiteren untersuchten Bibliotheken schnitten wie folgt ab (Gesamtbeurteilung im Anhang 4 zu finden):

4. Platz: PyGrahistry, Hauptausschlussgrund: keine Open Source Bibliothek

5. Platz: Cytoscape (Dash), Hauptausschlussgrund: wird nicht mehr weiterentwickelt

6. Platz: Plotly, Hauptausschlussgrund: keine wirkliche Interaktivität ausser Zoom-Funktion.

7. Platz: networkX, Hauptausschlussgrund: nur Statische Darstellung der Grafiken.

8. Platz: Holoviews, Hauptausschlussgrund: sehr starre Grafiken

9. Platz: PyGraphviz, Hauptausschlussgrund: kaum Interaktivität

10. Platz: yFiles-jupyter-graphs, Hauptausschlussgrund: wenige Entwickler arbeiten daran

11. Platz: Seaborn, Hauptausschlussgrund: Visualisierung sehr statisch

3.4 Finale Evaluation mittels Beispieldatensatzes

Zur genaueren Überprüfung der drei Bibliotheken liegen zwei, von der itopia AG zur Verfügung gestellten Datensätze als CSV-Dateien vor. In der einen Datei (`product_instance_relations1.csv`) sind die Beziehungen zwischen den Knoten gespeichert (Anhang 5). Sie bildet die Grundlagen zur Erstellung der Netzwerkgrafik. Die andere Datei (`product_instances1.csv`) enthält eine Liste von Attributen zu den jeweiligen Entitäten (Knoten).

	parent	relation_type	child
0	linuxappp-02	hosted-on	PRD1
1	linuxappt-02	hosted-on	PRD1
2	linuxappt-01	hosted-on	PRD1
3	linuxappp-01	hosted-on	PRD1
4	SAMPLEAPP (linuxappp-02)	runs-on	linuxappp-02

Abbildung 4: Ausschnitt aus dem Bespieldatensatz mit Beziehungen.

	serial	name	ci_type	product_name	vendor_model
0	1234	esx02	platform	Cisco ESXi Host	Cisco Systems Inc UCSB-B200-
1	1235	linuxappp-02	platform	Server Linux Virtual	VMware, Inc. VMware Virtual Platfo
2	1236	linuxappt-02	platform	Server Linux Virtual	VMware, Inc. VMware Virtual Platfo
3	1237	esx03	platform	Cisco ESXi Host	Cisco Systems Inc UCSB-B200-
4	1238	linuxappt-01	platform	Server Linux Virtual	VMware, Inc. VMware Virtual Platfo

Abbildung 5: Ausschnitt aus dem Beispieldatensatz mit Attributen.

3.4.1 Anwendungsbeispiel mit Bokeh

Bokeh ist eine interaktive Python-Bibliothek zur Visualisierung von Daten. Die Bibliothek bietet die Möglichkeit zur Darstellung von Netzwerkgrafiken und es lassen sich Interaktionen zwischen Kanten und Knoten konfigurieren.

Für die Erstellung der Grafik besteht die Möglichkeit, Knoten manuell anzulegen oder Daten in verschiedenen Dateiformaten wie CSV, JSON oder TXT zu importieren und zu bearbeiten. Anschliessend besteht der Prozess zur Ausführung eines Bokeh-Codes aus zwei Schritten: Zuerst wird ein vordefinierter Block ausgewählt, um die Visualisierung zu erstellen. Danach wird zusätzlicher Code hinzugefügt, um die Visualisierung so präzise wie möglich an die spezifischen Anforderungen anzupassen.

Im Folgenden werden die zentralen Abschnitte des Codes ausführlich erklärt. Der vollständige Code kann aus der bereitgestellten Code-Sammlung in Coltero entnommen werden.

Um eine Netzwerkgrafik zu erstellen, werden zunächst die erforderlichen Bibliotheken importiert: pandas, bokeh, networkx und matplotlib. Danach wird eine CSV-Datei (`product_instance_relations1.csv`) mit Produktinstanzen eingelesen und ein pandas DataFrame mit dem Namen `products_1_def` erstellt. Dafür werden zwei Sets erstellt: `node_names` enthält alle eindeutigen Knotennamen (Eltern- und Kindknoten) aus den Spalten "parent" und "child" des DataFrames. `node_indices` ist ein Wörterbuch, das jedem Knotennamen einen Index zuordnet. Zwei neue Spalten `parent_idx` und `child_idx` werden zum DataFrame hinzugefügt. Diese Spalten enthalten die Indexwerte für die Eltern- und Kindknoten, die aus dem Wörterbuch `node_indices` abgerufen werden.

In einem weiteren Schritt wird ein leeres, gerichtetes Graphenobjekt `G` erzeugt. Die Kanten des Graphen werden mit Hilfe der Methode `nx.from_pandas_edgelist` aus dem DataFrame erstellt. Dabei werden die Spalten `parent_idx` und `child_idx` als Quell- und Zielknoten sowie die Spalte `relation_type` als Attribut verwendet.

Für weitere Umgestaltung der Grafik werden die Bokeh-Modelle und -Funktionen importiert (Abbildung 6).

```
from bokeh.io import show, output_notebook, save
from bokeh.models import Range1d, Circle, Ellipse, Rect, Arrow, ColumnDataSource, Bezier, MultiLine, Plot,
 LinearColorMapper, Circle, GraphRenderer, EdgesAndLinkedNodes, NodesAndLinkedEdges, LabelSet
from bokeh.palettes import Spectral4, all_palettes, Blues8, Magma, Category10, Inferno, Viridis, Category20
from bokeh.transform import linear_cmap
from bokeh.plotting import from_networkx, figure
```

Abbildung 6: Import benötigte Funktionen aus Bokeh

Das Layout für die Netzwerkgrafik wird festgelegt. Im vorliegenden Beispiel wird das `spring_layout` von networkx verwendet. Dabei werden die Skalierung und der Mittelpunkt festgelegt.

Die Darstellung von Kanten und Knoten kann individuell angepasst werden. Der Personalisierungsprozess ist intuitiv gestaltet, wobei englische Begriffe verwendet werden. Bei Kanten (Edges) können Farben (zum Beispiel Blau), Dicke und Form personalisiert werden. Bei Knoten (Nodes) erfolgt zuerst die Auswahl der Form (zum Beispiel Rechtecke). Anschliessend können weitere Anpassungen vorgenommen werden (Abbildung 7).

```
# #anpassen Knoten und Kanten
g.edge_renderer.glyph = MultiLine(line_color="blue", line_alpha=0.5, line_width=1, line_join="round")
g.node_renderer.glyph = Rect(name="name", height=4, width=7, fill_color="lightblue", line_color="gray")
```

Abbildung 7: Personalisierung von Knoten und Kanten.

Die Beschriftungen der Knoten werden hinzugefügt, indem die Koordinaten der Knoten aus dem Layout verwendet werden. Die Daten für die Labels werden als `ColumnDataSource` definiert, um sie mit den Bokeh-Tools zu verknüpfen. Die Labels werden mit Hilfe der `LabelSet`-Funktion erstellt und der Grafik hinzugefügt (Abbildung 8).

```
#Add Labels
x, y = zip(*g.layout_provider.graph_layout.values())
node_labels = list(G.nodes("name"))
source = ColumnDataSource({'x': x, 'y': y, 'name': [node_labels[i] for i in range(len(x))]})
labels = LabelSet(x='x', y='y', text='name', source=source, background_fill_color='white', text_alpha=1, text_font_size='13px',
  background_fill_alpha=0)
plot.renderers.append(labels)
```

Abbildung 8: Zufügen von Beschriftungen.

Die Integrationsfunktionen im Graphen sind in Bokeh nativ implementiert, was bedeutet, dass selbst der einfachste Graph bereits über eine Toolbox auf Graphenoberfläche verfügt. Mit der Methode `add_tools()` können zusätzliche Tools zu diesem Toolkit hinzugefügt werden. Zum Beispiel können mithilfe der `HoverTool`-Klasse Funktionen hinzugefügt werden, die beim Bewegen des Mauszeigers über das Bild aktiviert werden. Zusätzliche Erweiterungsmöglichkeiten sind unter anderem das `ZoomInTool`, das `ZoomOutTool`, das `TapTool`, das `PanTool` oder andere (Abbildung 9).

```
#Interaktive Funktionen: Zoom, Hover, Pan, Knoten hervorheben ...
HOVER_TOOLTIPS = [("name", "@name")]

plot = figure(tooltips=HOVER_TOOLTIPS, background_fill_color="white",
              tools="pan, zoom_in, zoom_out, wheel_zoom, save, reset, freehand_draw, tap", active_scroll='wheel_zoom',
              x_range=Range1d(-60, 60), y_range=Range1d(-60, 60), title=title)
```

Abbildung 9: Interaktive Tools

Um die interaktive Grafik im Browser anzuzeigen, wird die `plot`-Figure mithilfe der Funktion `show` dargestellt (Abbildung 10).
Die erstellte Netzwerkgrafiken können als:

- HTML (*`save (plot, filename= Name.html`*)),
- PNG (*`from bokeh.io import export_png`*),
- SVG (*`from bokeh.io import export_svg`*)

gespeichert werden. Das bietet die Möglichkeit, Grafiken für Präsentationen oder Berichte zu verwenden.

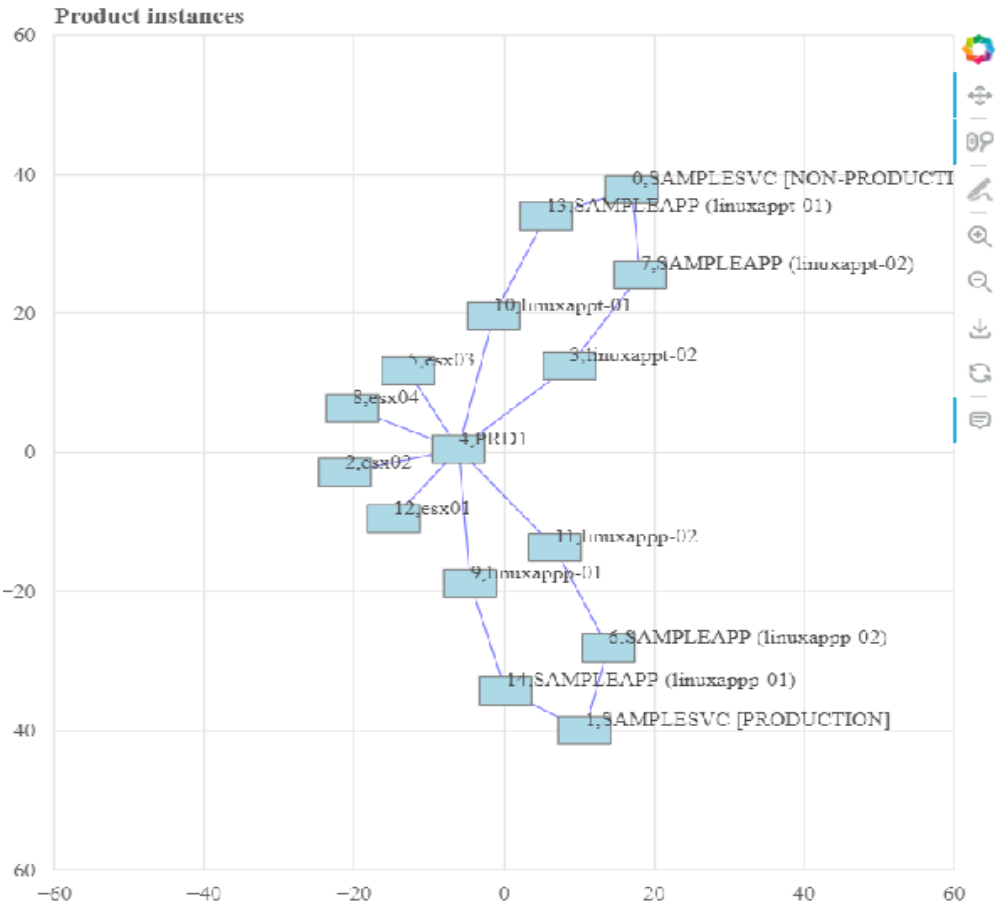

Abbildung 10: Netzwerkgrafik mit Bokeh

3.4.2 Anwendungsbeispiel mit igraph

igraph ermöglicht die Erstellung von dynamischen Graphen in Python mit interaktiven Funktionen und Kommentarmöglichkeiten, sofern zusätzlich weitere Bibliotheken wie plotly und ipywidgets importiert werden. Die Bibliothek bietet eine umfassende Palette von Funktionen zur Erzeugung, Visualisierung und Analyse von Netzwerkgrafiken. Die Erstellung einer Grafik mit igraph ermöglicht sowohl das manuelle Hinzufügen von Knoten als auch den Import und die Bearbeitung von Daten aus verschiedenen Dateiformaten wie CSV, JSON oder TXT. Damit können Netzwerke aus unterschiedlichen Quellen geladen und analysiert werden. Veränderungen und Anpassungen der Netzwerkgrafik werden direkt im Code vorgenommen.

Um die erstellten Netzwerkgrafiken weiterzuverwenden, bietet igraph verschiedene Speicheroptionen: Die Grafiken können als HTML, PNG oder SVG gespeichert werden, um sie in Präsentationen, Berichten oder anderen Dokumenten zu nutzen. Dies ermöglicht die nahtlose Integration der Grafiken in verschiedene Medien und Anwendungen.

Folgend werden die wichtigsten Code-Ausschnitte im Detail erläutert. Der ganze Code wird über Coltero zur Verfügung gestellten.

Der Code (Abbildung 11) erzeugt einen leeren Graphen durch Einbezug des DataFrame `df`, welcher Informationen über die Eltern-Kind-Beziehungen enthält. Zur Erstellung einer Netzwerkgrafik wird ein leeres igraph-Objekt `g` erstellt, das als Container für den Graphen dient. Zusätzlich wird eine Hilfsstruktur `added_nodes` als leere Menge initialisiert, um bereits hinzugefügte Knoten zu verfolgen. Anschliessend iteriert eine Schleife über jede Zeile des DataFrames `df`, wodurch für jede Zeile der Elternknoten (`parent`) und der Kindknoten (`child`) abgerufen und überprüft wird, ob die Knoten bereits in `added_nodes` vorhanden ist. Falls nicht, wird ein neuer Knoten mit dem Namen `parent` bzw. `child` zum Graphen `g` und zu `added_nodes` hinzugefügt. Schliesslich wird eine Kante zwischen dem Elternknoten und dem Kindknoten zum Graphen `g` hinzugefügt.

```
# Graph erstellen
g = igraph.Graph()
added_nodes = set()  # Hilfsstruktur zum Verfolgen bereits hinzugefügter Knoten

for i, row in df.iterrows():
    parent = row["parent"]
    child = row["child"]

    # Knoten hinzufügen, falls noch nicht hinzugefügt
    if parent not in added_nodes:
        g.add_vertex(name=parent)
        added_nodes.add(parent)
    if child not in added_nodes:
        g.add_vertex(name=child)
        added_nodes.add(child)

    # Kante hinzufügen
    g.add_edge(parent, child)
```

Abbildung 11: Erstellung des Graphen

Die Anpassung von Kanten und Knoten (Abbildung 12) erfolgt separat. Für Kanten können Farben, Dicke und Form personalisiert werden. Dies ermöglicht eine präzise Darstellung und Unterscheidung verschiedener Verbindungen und Beziehungen. Bei Knoten wird zunächst die Form ausgewählt und anschliessend können weitere Anpassungen wie Farben oder Grössenänderungen vorgenommen werden. Dadurch lassen sich Knoten in der Netzwerkgrafik visuell hervorheben oder bestimmten Kategorien zuordnen.

```
# Knotenfarben festlegen
g.vs.find(name="PRD1")["color"] = "green"
g.vs.find(name="SAMPLEAPP (linuxappp-02)")["color"] = "red"
```

Abbildung 12: Festlegung der Knotenfarben

Die Beschriftung (Abbildung 13) der Knoten erfolgt separat und kann individuell angepasst werden. Es besteht die Möglichkeit, Beschriftungen innerhalb der Knoten oder ausserhalb der Knoten anzuzeigen. Die Position, das Erscheinungsbild und die Formatierung der Beschriftungen können an die Anforderungen und den Kontext der Netzwerkgrafik angepasst werden.

```
# Knotenbeschriftungen festlegen
g.vs["label"] = g.vs["name"]
```

Abbildung 13: Festlegung der Knotenbeschriftungen

Für jeden Knoten im Graphen `g.vs` wird eine visuelle Darstellung (`trace`) erstellt und zur Plotly-Grafik `fig` hinzugefügt (Abbildung 14).
Die Eigenschaften jedes Knotens werden dabei wie folgt festgelegt:

- Die x-Koordinate des Knotens wird mit `layout[node.index][0]` definiert, wobei `layout` das zuvor festgelegte Layout des Graphen ist.
- Die y-Koordinate des Knotens wird mit `layout[node.index][1]` definiert.
- Der Text, der den Knoten beschriftet, wird mit `node["label"]` festgelegt, wobei `node` ein Knotenobjekt aus dem Graphen ist.
- Der Modus für den Knoten ist auf "`markers+text`" gesetzt, was bedeutet, dass der Knoten als Marker angezeigt wird und der Text neben dem Marker platziert wird.
- Die Grösse des Markers wird mit `marker=dict(size=10, color=node["color"])` festgelegt, wobei `size=10` die Grösse des Markers und `color=node["color"]` die Farbe des Markers entsprechend der im Graphen festgelegten Farbe ist.
- Die Position des Textes relativ zum Marker wird mit `textposition="top center"` festgelegt.

```
# Knoten hinzufügen
for node in g.vs:
    fig.add_trace(
        go.Scatter(
            x=[layout[node.index][0]],
            y=[layout[node.index][1]],
            text=node["label"],
            mode="markers+text",
            marker=dict(size=10, color=node["color"]),
            textposition="top center",
        )
    )
```

Abbildung 14: Hinzufügen der Knoten

Auch für jede Kante im Graphen `g.es` wird eine visuelle Darstellung (`trace`) erstellt und zur Plotly-Grafik `fig` hinzugefügt (Abbildung 15).

Die Positionen der Kanten werden festgelegt, indem die x-Koordinaten der Kanten mit `[layout[edge.source][0], layout[edge.target][0]]` definiert werden. Wobei `layout` das zuvor festgelegte Layout des Graphen ist. `edge.source` und `edge.target` geben die Start- und Endpunkte der Kante an. Danach werden die y-Koordinaten der Kanten mit `[layout[edge.source][1], layout[edge.target][1]]` definiert.

Die Kanten werden im vorliegenden Beispiel als Linie angezeigt (Modus der Kante ist auf `"lines"` gesetzt). Für diese werden die Linienbreite von eins und Farbe grau mit `line=dict(width=1, color="gray")` festgelegt

```
# Kanten hinzufügen
for edge in g.es:
    fig.add_trace(
        go.Scatter(
            x=[layout[edge.source][0], layout[edge.target][0]],
            y=[layout[edge.source][1], layout[edge.target][1]],
            mode="lines",
            line=dict(width=1, color="gray"),
        )
    )
```

Abbildung 15: Hinzufügen der Kanten

Es wird ein Dropdown-Menü erstellt (Abbildung 16), das die Namen der Knoten im Graphen enthält. Das `options`-Argument erhält die Namen der Knoten aus dem Attribut `"name"` der Knoten im `g`-Graphen. Der Parameter `"Knoten:"` wird als Beschreibung für das Dropdown-Menü angezeigt.

```
# Dropdown-Widget für Knotenauswahl erstellen
node_dropdown = widgets.Dropdown(
    options=g.vs["name"],
    description="Knoten:",
)
```

Abbildung 16: Erstellung eines Dropdown-Menüs fur das Knotenauswahl

Mittels des Codes in Abbildung 17 wir ein Kommentar-Widget mit einem Textfeld und einem Hinzufügen-Button erstellt. Dafür wird ipywidgets als zusätzliche Bibliothek benötigt. Das Textfeld wird mit dem Placeholder-Text `"Geben Sie hier Ihren Kommentar ein"` initialisiert. Der Hinzufügen-Button wird mit dem Beschriftungstext `"Kommentar hinzufügen"` erstellt.

```
# Kommentar-Widget erstellen
comment_box = widgets.Text(placeholder='Geben Sie hier Ihren Kommentar ein')
comment_button = widgets.Button(description='Kommentar hinzufügen')
```

Abbildung 17: Erstellung eines Kommentar-Widgets

Wenn der `"Kommentar hinzufügen"` angeklickt wird, wird die Funktion namens `add_comment`, ausgeführt (Abbildung 18). Dafür sind folgende Schritte nötig: Zuerst wird der aktuelle Kommentarwert aus dem `comment_box`-Textfeld abgerufen und in der Variablen `comment` gespeichert. Gleichzeitig wird der ausgewählte Knotenwert aus dem Dropdown-Menü `node_dropdown` in der Variablen `selected_node` gespeichert. Anschliessend wird mit der `find`-Methode von `g.vs` der Index des ausgewählten Knotens basierend auf seinem Namen gefunden und in der Variablen `node_index` gespeichert. Der Kommentarwert `comment` wird dem Knotenattribut `"comment"` des entsprechenden Knotens `g.vs[node_index]` in `g` zugewiesen. Zudem wird der Hovertext des entsprechenden Knotens in der `fig` aktualisiert, sodass der Kommentar im Hovertext angezeigt wird. Schliesslich wird `fig` aktualisiert und angezeigt, um den neuen Kommentar darzustellen.

```
# Kommentar hinzufügen
def add_comment(b):
    comment = comment_box.value
    selected_node = node_dropdown.value
    node_index = g.vs.find(name=selected_node).index
    g.vs[node_index]["comment"] = comment
    fig.data[node_index].update(hovertext=comment)  # Kommentar im Hovertext aktualisieren
    fig.show()


comment_button.on_click(add_comment)
```

Abbildung 18: Die Kommentarfunktion

Durch die Funktion wurde dem Knoten PDR1 der Kommentar „Test-Kommentar“ (Abbildung 19) hinzugefügt, welcher als Hovertext über die Hover-Funktion kenntlich gemacht wird.

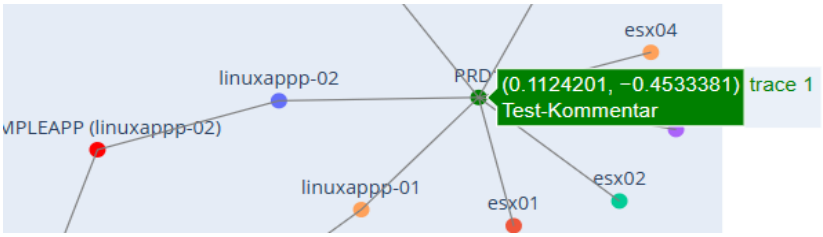

Abbildung 19: Beispiel eines eingefügten Kommentars

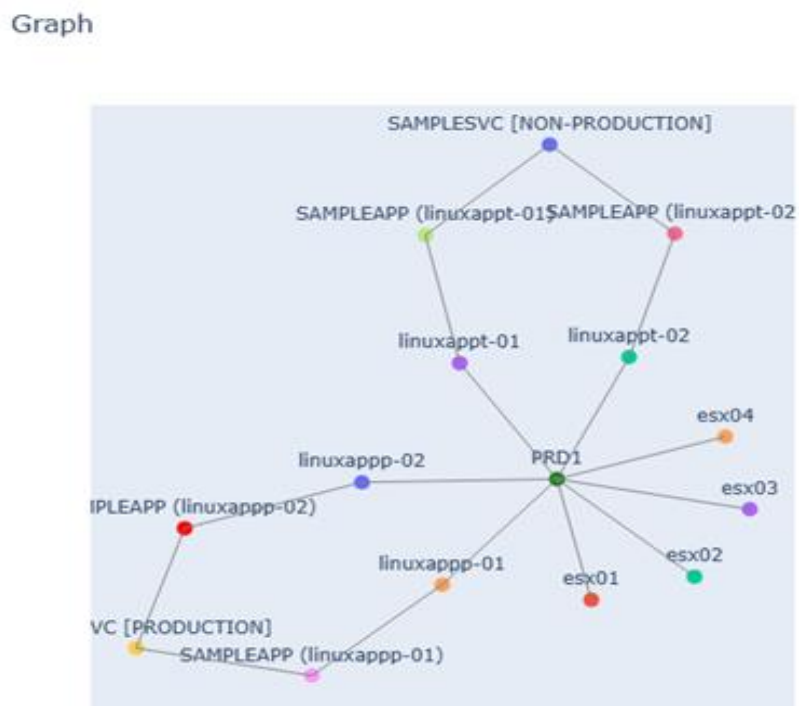

Abbildung 20: Beispiel einer mit igraph erzeugten Netzwerkgrafik mit dem Namen "Graph"

3.4.3 Anwendungsbeispiel mit PyVis

PyVis ist eine Python-Bibliothek, welche als Open Source Software verfügbar ist und deren Quellcode auf GitHub zugänglich ist. Um Graphen im Browser zu visualisieren, ist die JavaScript Bibliothek vis.js erforderlich, da PyVis das Python Interface von vis.js ist. Dadurch besteht eine gewisse Abhängigkeit von einem Drittanbieter.

PyVis setzt die BSD 3 Clause Lizenz ein. Diese Lizenz wird häufig für Softwareprojekte verwendet. Sie ist eine kommerziell freundliche Lizenz, da sie Entwicklern und Entwicklerinnen erlaubt, die Bibliothek weiterzuentwickeln, ohne die Änderungen am Quellcode offenzulegen. Es ist lediglich erforderlich, den Urheberrechtsvermerk und den Haftungsausschluss beizubehalten.

Dies trägt seinen Teil dazu bei, dass PyVis von einer aktiven Community profitiert. Rund 30 Personen entwickeln die Bibliothek kontinuierlich weiter. Wodurch regelmässig neue Funktionen und Verbesserungen bereitgestellt werden und es eine umfangreiche Dokumentation gibt.

Für den Datenimport werden verschiedene Datenformate von PyVis unterstützt (CSV, JSON, Pandas DataFrames), was ein schnelles Importieren von verschiedenen Datenquellen ermöglicht. Die für diese Arbeit zur Verfügung gestellten Datensätze werden mit Hilfe von Pandas eingelesen. Nach dem Import der Daten bietet PyVis eine Vielzahl von Anpassungsmöglichkeiten für die Darstellung der Netzwerkgrafik.

Um eine neue Netzwerkgrafik zu erstellen, wird zunächst ein leeres Netzwerkobjekt als Container für das Diagramm erzeugt. In diesem Objekt können verschiedene Eigenschaften und Einstellungen festgelegt werden, die für die gesamte Netzwerkgrafik Gültigkeit haben (Abbildung 21):

```
net = Network(
    # Darstellung
    heading = 'Testdatensatz',  # Name der Grafik

    notebook = True,            # True, wenn die Grafik im Jupyter Notebook dargestellt werden soll

    height = '500px',           # Höhe der Netzwerkgrafik
    width = '100%',             # Breite der Netztwerkgrafik

    bgcolor = '#F7F6F5',        # Hintergrundfarbe hinzufügen
    font_color = 'black',       # Label Farbe bestimmen

    # Interaktivität hinzufügen
    select_menu=True,           # Suchbalken integrieren
    filter_menu = True,         # Filteroptionen integrieren
)
```

Abbildung 21: PyVis Netzwerkobjekt

Die Such- und Filterbalken erleichtern das Analysieren eines Datensatzes, weshalb die Integration zumindest eines Balkens Sinn macht.

Nach der Erstellung des Netzwerkobjekts werden die Knoten und Kanten hinzugefügt. Dies kann entweder durch das Hinzufügen einzelner Knoten und Kanten erfolgen:

```
net.add_node(100, label='Knoten 1')
net.add_node(200, label='Knoten 2')
net.add_edge(100, 200)
```

Alternativ - wie in vorliegenden Beispiel - kann dies auch durch Iterieren über ein DataFrame erfolgen. Dabei werden Knoten festgelegt und über Kanten zusammengefügt. Ergänzende Informationen werden als `title = ''` den jeweiligen Knoten und Kanten hinzugefügt (ersichtlich im Anhang 6).

PyVis bietet die Möglichkeit, die Knoten in verschiedenen Formen und Farben zu gestalten, um Informationen zu codieren und wichtige Merkmale hervorzuheben. Dazu kann beispielsweise ein `node_typ` Dictionary erstellt werden, das abhängig von definierten Attributen den Knoten ein Design zuweist. Zu unterscheiden sind zwei Arten von Knotenformen: die einen werden innerhalb der Form beschriftet, wobei die Grösse des Knoten automatisch der Textlänge angepasst wird (`'ellipse', 'circle', 'database', 'box', 'text'`). Bei den anderen Knotenformen wird das Label ausserhalb des Knotens platziert (`'image', 'circularImage', 'diamond', 'dot', 'star', 'triangle', 'triangleDown', 'square', 'icon'`). Dadurch ist die Grösse des Knotens individuell festlegbar. Im vorliegenden Beispiel wurde die Grösse der Knoten davon abhängig gemacht, ob eine Entität als 'Non-Production' klassifiziert ist (Abbildung 22).

```python
net = Network()

node_types = {
    'platform': {'color': '#DDDCD8'},
    'application': {'color': '#BBDBF2', 'shape': 'diamond'},
    'service': {'color': '#FCBE10', 'shape': 'star'}
}

for index, row in df_merge.iterrows():
    mother = row['parent']
    child = row['child']
    relation_type = row['relation_type']

    for node in [mother, child]:
        node_info = df_info[df_info['name'] == node]
        node_type = node_info['ci_type'].values[0]
        serial_number = node_info['serial'].values[0]
        productivity = node_info['ci_stage'].values[0]

        node_properties = node_types.get(node_type, {})

        node_label = node
        title = f"Type: {node_type}" \
                f"\nSerial Number: {serial_number}" \
                f"\nProductivity: {productivity}"

        node_size = 15 if productivity == 'NON-PRODUCTION' else 30

        net.add_node(node, label=node_label, title=title, size=node_size, **node_properties)

    net.add_edge(mother, child, title=relation_type)

net.show('Knotengestaltung-Netzwerk.html')
```

Abbildung 22: PyVis Knotengestaltung

Knoten können auch zum Einfügen von Kommentaren verwendet werden. Dafür wird eine `'box'` oder `'text'` Form gewählt. Für längere Texte ist dies jedoch ein mühsames Vorgehen, da kein automatischer Zeilenumbruch vorgenommen wird.

Ein weiterer Einsatzbereich von Knoten ist die Integration von Firmenlogos oder anderen Bildern. Hierfür wird der Pfad oder die URL zur Bilddatei angegeben. Zu beachten ist, dass das Bild die richtige Grösse und Auflösung aufweist.

Neben den Knoten können auch Kanten in verschiedenen Farben und Stärken gestaltet werden. Zudem können die Kanten im gesamten Diagramm gerichtet integriert werden. Dies wird bereits bei der Erstellung des Netzwerkobjekts mitgegeben (`directed = True`). Alternativ werden einzelne gerichtete Graphen eingefügt, indem der `add_edges` Methode der Parameter `arrows = 'to'` hinzugefügt wird.

Knoten können in der Netzwerkgrafik manuell verschoben werden. In PyVis sind jedoch auch verschiedene Algorithmen und Layout-Optionen integriert. Diese werden entweder durch die Verwendung von Funktion direkt im Code angepasst oder sie können in der Grafik durch Hinzufügen einer interaktiven Button-Anzeige eingestellt werden (Abbildung 23).

```
net.show_buttons(filter_=['layout', 'interaction', 'physics'])
```

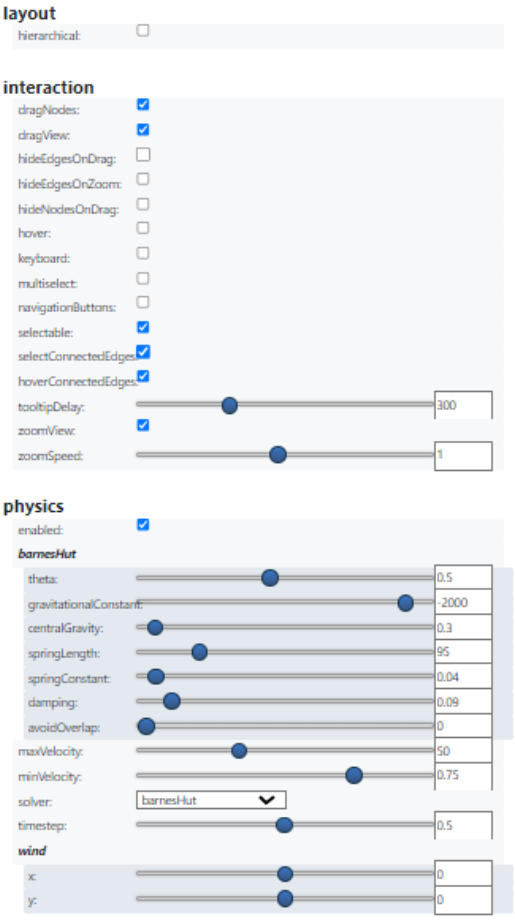

Abbildung 23: PyVis Buttons

Über die Buttons können zudem Interaktionsmöglichkeiten wie zum Beispiel die Hover-Funktion oder das Zoomen aktiviert oder deaktiviert werden.

PyVis bietet Möglichkeiten zur Integration in Webanwendungen: Die Bibliothek unterstützt die Erstellung von HTML-Widgets, die in Webseiten oder Jupyter Notebooks eingebettet werden können. Dies ermöglicht es, Visualisierungen in bestehende Projekte zu integrieren und sie gemeinsam mit anderen interaktiven Elementen anzuzeigen (Abbildung 24).

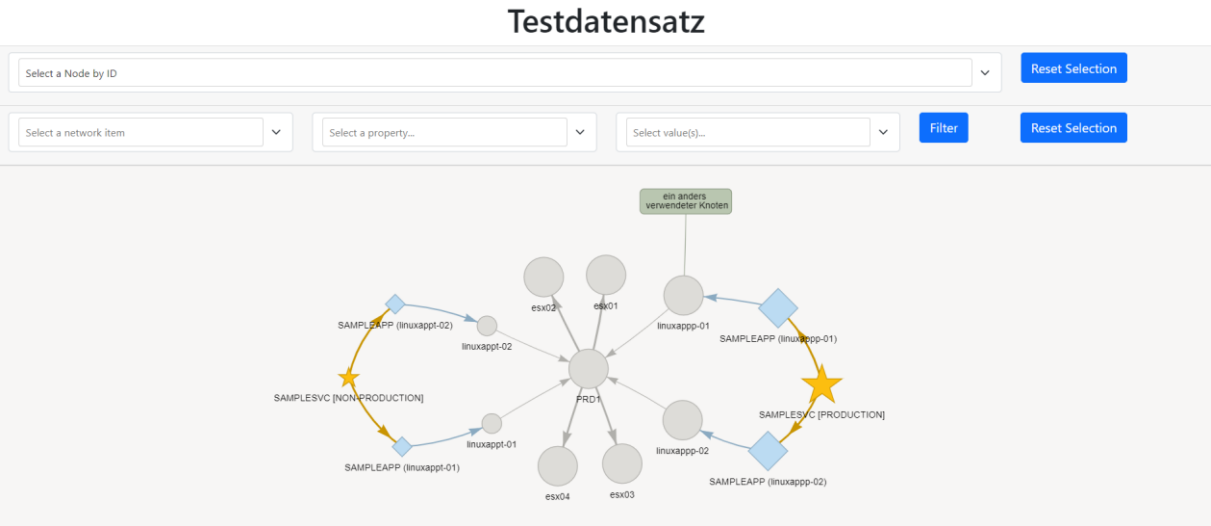

Abbildung 24: PyVis Beispielnetzwerkgrafik

Somit ist es möglich, komplexere Corporates Designs über ein benutzerdefiniertes CSS oder eine HTML-Vorlagen zu integrieren. Dies wurde im Rahmen dieser Arbeit nicht getestet, da keine HTML-Kenntnisse vorhanden sind.

Die Netzwerkvisualisierungen können in verschiedenen Formaten (PNG, SVG, PDF) exportiert werden. Dadurch können die Grafiken für Präsentationen oder Berichte verwendet werden.

3.4.4 Resultat der Evaluation

Genau wie PyVis bieten auch igraph und Bokeh Funktionen zur Anpassung des Grafikstils wie die Farbe und Grösse der Knoten und Kanten sowie verschiedene Layout-Algorithmen zur Anordnung der Knoten. Weiter bieten alle drei Bibliotheken die Möglichkeit, Kommentare über den Code hinzuzufügen und grundlegende Interaktionsmöglichkeiten wie das Zoomen.

Jedoch gibt es einige Aspekte, bei denen PyVis im Vergleich zu igraph und Bokeh zusätzliche Vorteile bietet:

- Einfachere Benutzeroberfläche und Dokumentation: Die Benutzeroberfläche von PyVis ist bekannt für ihre Benutzerfreundlichkeit und die klare Dokumentation, die es Benutzer und Benutzerinnen erleichtert, schnell und effektiv Grafiken zu erstellen und anzupassen. Die Benutzeroberfläche von igraph kann als weniger ansprechend und der Code als komplexer wahrgenommen werden, insbesondere für Anfänger, und die Dokumentation kann weniger umfangreich oder weniger zugänglich sein. Ausserdem bietet die

Benutzeroberfläche von PyVis diverse Filtermöglichkeiten und Regler, welche beispielsweise bei igraph weder vorhanden sind noch direkt in die Benutzeroberfläche implementiert werden können. Der Code könnte allenfalls entsprechend erweitert werden, wobei aber immer über den Code interagiert werden müsste.

- Breitere Palette an Interaktionsmöglichkeiten: PyVis bietet eine umfassendere Palette von interaktiven Funktionen, die direkt in die Grafik integriert sind. Dazu gehören Funktionen wie das Ein- und Ausblenden von Knoten und Kanten, das Hervorheben von Verbindungen und das Anzeigen von Informationen beim Überfahren mit der Maus. igraph und Bokeh bieten zwar auch eine Hover-Funktion, aber nicht dieselbe Vielfalt und Leichtigkeit der Integration für diese und andere Interaktionen. Die Benutzeroberfläche von igraph beschränkt sich auf eine mehr statische als dynamische Ausgabe des Graphen. Das Verschieben, Anordnen, Bewegen der Graphen-Elemente, insbesondere der Knoten, ist über die Ausgabe nicht möglich und muss im Code vorgenommen werden.

Es ist wichtig zu beachten, dass die Wahl zwischen den drei Bibliotheken von den spezifischen Anforderungen, der Komplexität des Projekts und den individuellen Vorlieben abhängt. igraph und Bokeh sind bewährte und leistungsstarke Bibliothek, während PyVis eine relativ junge Bibliothek ist, die speziell auf die Erzeugung interaktiver Graphen ausgelegt ist. Alle drei Bibliotheken bieten Funktionen zur Erstellung von dynamischen Graphen.

Basierend auf den vorgelegten Anwendungsbeispielen erweist sich PyVis als die überlegene und endgültige Empfehlung. Die Bibliothek zeichnet sich durch ihre Benutzerfreundlichkeit und den umfangreichen Interaktionsmöglichkeiten aus.

4 Diskussion und Grenzen

4.1 Grenzen der angewandten Methodik

Die Nutzwertanalyse ermöglicht eine transparente Dokumentation des Entscheidungsprozesses, weist jedoch auch gewisse Beschränkungen auf. Ein wesentlicher Nachteil besteht darin, dass sie auf subjektiven Bewertungen beruht. In dieser Arbeit wurde diesem Problem entgegengewirkt, indem das Autorenteam die Anforderungen gewichtet hat und anschliessend die Auftraggeberin die Gewichtung kritisch hinterfragt und angepasst hat.

Bei der Definition des Anforderungskatalog ist darauf zu achten, dass alle Projektmitglieder dasselbe Verständnis der einzelnen Anforderungen haben und wann sie als erfüllt und nicht erfüllt gelten. Andernfalls können Unterschiede bei der Beurteilung auftreten. Aus diesem Grund wurde die Beurteilung nur von einer Person vorgenommen und im Anschluss von den anderen Projektmitglieder kritisch hinterfragt.

Die Nutzwertanalyse berücksichtigt nur die zuvor festgelegten Anforderungen. Bei den Bewertungen der Bibliotheken sind keine weiteren Kriterien aufgefallen, die hätten in Betracht gezogen werden müssen. Jedoch wurde festgestellt, dass Kriterien wie die Gestaltung der Knoten einen erheblichen Einfluss auf die Gesamtbewertung haben, da diese in zwei Anforderungen definiert wurden (Farbe und Form). Die Gestaltung der Kanten hingegen wurde nur in einer Anforderung definiert.

Die Grenzen der Nutzwertanalyse waren bekannt, daher wurde nicht nur eine, sondern die drei höchstbewerteten Bibliotheken genauer betrachtet. Es steht aber zur Diskussion, ob die Python Bibliothek Bokeh, die den dritten Platz belegt hat, tatsächlich noch weiter hätte untersucht werden müssen, da sie einen deutlichen Abstand zur zweitplatzierten Bibliothek aufwies (11 von möglichen 114 Punkten).

Die überprüften Bibliotheken wurden lediglich anhand eines begrenzten Datensatzes getestet. Die Skalierbarkeit der Lösungen wurde deshalb nur durch Beispiele von anderen Datensätzen und Tutorials aus dem Internet beurteilt. Eine Evaluierung der Leistung und Skalierbarkeit der Bibliotheken in Bezug auf komplexe Datensätze, mit welchen die itopia AG arbeitet, ist notwendig, um festzustellen, wie gut sie in solchen Szenarien performen.

4.2 Grenzen der evaluierten Python-Bibliothek

PyVis ist eine Python-Bibliothek, die ein hohes Potenzial aufweist, die heute verwendeten Tools Neo4j und Graphviz zu ersetzen. Sie ermöglicht das Einlesen von grossen Datensätzen wie auch das manuelle Erstellen von Netzwerkgrafiken.

PyVis kann mit verschiedenen Dateiformaten und Quellen arbeiten und bietet umfangreiche Anpassungsmöglichkeiten für die Gestaltung der Netzwerkgrafik an, einschliesslich der Integration von CSS und HTML-Vorlagen. Zudem ermöglicht PyVis eine interaktive Exploration der Daten durch Funktionen wie das Hovern über Elemente und das Zoomen. Die Interaktionsmöglichkeiten können dabei individuell an die Anforderungen eines Projekts angepasst werden.

Allerdings konzentriert sich PyVis hauptsächlich auf die Visualisierung und bietet keine erweiterten Funktionen zur Exploration und Analyse von Datenbanken, wie sie beispielsweise mit Neo4j möglich wären. Dies ist als eine Einschränkung zu betrachten, falls umfassende Datenbankanalysen erforderlich sind.

Zusätzlich zu berücksichtigen ist, dass das Hinzufügen von Kommentaren umständlich ist und einen Umweg über die Erstellung von manuellen Knoten erfordert. Dies kann als ein grösserer Nachteil angesehen werden, da Kommentare oft eine wichtige Rolle bei der Dokumentation und Erklärung von Netzwerkgrafiken spielen.

Wenn die PyVis Bibliothek verwendet wird, müssen bei Workshops weiterhin zwei Personen anwesend sein: eine Person, die den Workshop leitet, und eine zweite Person, die die besprochenen Konzepte direkt im Code umsetzt. Um diesen Umstand zu ändern, sollten Weiterentwicklungen der PyVis Bibliothek in Betracht gezogen werden. Auf diese werden im nachfolgenden Kapitel eingegangen.

5 Implikation für die Praxis

Vor der Implementierung der PyVis Bibliothek in der Praxis sind weitere Abklärungen erforderlich, damit sichergestellt werden kann, dass die Integration reibungslos verläuft:

- Zunächst sollte getestet werden, ob PyVis in der Lage ist, mit den komplexen Datenstrukturen umzugehen, die von einigen Kunden der itopia AG bereitgestellt werden.
- Weiter ist zu prüfen, wie HTML-Vorlagen oder angepasste HTML in den PyVis-Code integriert werden können. Mit der Integration der HTML-Vorlagen könnte die Anpassungsfähigkeit Netzwerkgrafiken verbessert werden.
- Auch zu testen ist die Integration von PyVis mit anderen Tools, Datenbanken oder Frameworks, falls dies gewünscht ist.

Wenn alle diese Aspekte erfolgreich getestet wurden, wird empfohlen, die Anwendung von PyVis in einer internen Nachstellung einer echten Workshop-Situation zu prüfen. Dabei sollten verschiedene von der itopia AG erläuterte Workshop-Szenarien nachgestellt werden, um die Effektivität und Benutzerfreundlichkeit von PyVis zu bewerten. Es ist wichtig, während dieser Tests eventuell erforderliche Anpassungen oder Erweiterungen zu identifizieren. Beispielsweise könnte ein Add-On erwogen werden, das zusätzliche Anpassungsmöglichkeiten bietet, wie etwa das einfache Einfügen von Logos oder die Anpassung von Farben.

Durch die Erfüllung der genannten Punkte wird sichergestellt, dass PyVis den Anforderungen der itopia AG hinsichtlich Datenverarbeitung, Integration und Anpassungsfähigkeit gerecht wird und somit in einer Workshop-Umgebung effektiv eingesetzt werden kann.

Zusätzlich sind im Verlauf der Arbeit weitere Features identifiziert worden, die bei der Implementierung einer Netzwerkgrafik-Bibliothek in Betracht gezogen werden können:

- Ipywidgets: Diese Python-Bibliothek stellt interaktive HTML-Widgets für Jupyter Notebooks, JupyterLab und andere IPython-basierte Umgebungen bereit. Mit Ipywidgets können Benutzer und Benutzerinnen interaktive Steuerelemente wie Kommentare, Schieberegler, Dropdown-Menüs, Auswahllisten oder Schaltflächen erstellen, um die Visualisierungen von Daten interaktiv zu gestalten. (Ipywidgets, 2023)

- Ipysigma: Die Bibliothek bietet ein Jupyter-Widget, mit dem die Benutzer und Benutzerinnen visuelle Netzwerkanalysen mit einem Notebook durchführen können. ipysigma macht es einfach, die visuellen Variablen eines Netzwerks zu personalisieren. Auf diese Weise kann eine Arbeit an der Schnittstelle zwischen der Python-Verarbeitung von Graphdaten und der visuellen Exploration durchführen. ipysig unterstützt networkx und igraph nahtlos und kann auch von Numpy- und Pandas-Benutzern problemlos verwendet werden. (ipysigma, 2023)

Literaturverzeichnis

Barabási, A.& Bonabeau E. (2004). Skalenfreie Netze. Spektrum der Wissenschaft Juli, 2004:62– 69.

Bokeh Documentation. http://bokeh.org/. Besucht am 11.04.2023.

Bokeh on GitHub. https://github.com/bokeh/bokeh. Besucht am 11.04.2023.

Burch, M. (2023, 24. März). Graphen und Netzwerke [Vorlesungsfolien]. Fachhochschule Graubünden. https://moodle.fhgr.ch/mod/resource/view.php?id=596292

Dash Cyposcape Dokumentation (2019). Dash Cyposcape. https://dash.plotly.com/cytoscape. Besucht am 15.04.2023.

Fruchterman, T. M. & Reingold E. M. (1991). „Graph drawing by force-directed placement“. Software: Practice and experience, 21(11), 1129–1164.

Gebhart M. (2014). Qualitätsorientierter Entwurf von Anwendungsdiensten. KIT Scientific Publ. S.332.

Graphistry Homepage (2023). https://www.graphistry.com/. Besucht am 23.04.2023.

Hachul, S. & Jünger, M (2007). Large-Graph Layout Algorithms at Work: An Experimental Study. J. Graph Algorithms Appl., 11(2), 345–369.

Holoviews Dokumentation (2023). https://holoviews.org/. Besucht am 21.04.2023.

Holoviews on GitHub (2023). Holoviews. https://github.com/holoviz/holoviews. Besucht am 21.04.2023.

itopia (2023). https://www.itopia.ch/. Besucht am 24.05.2023

igraph Dokumentation (2023). https://igraph.org/python/. Besucht am 02.06.2023.

igraph on GitHub (2023). Python-Igraph. https://github.com/igraph/python-igraph. Besucht am 22.04.2023.

ipysigma (2023): a Jupyter widget for interactive visual network analysis. Workshop.

FOSDEM 2023 - ipysigma: a Jupyter widget for interactive visual network analysis Besucht am 02.06.2023.

Ipywidgets Dokumentation (2023). https://ipywidgets.readthedocs.io/en/stable/ Besucht am 02.06.2023.

Keller, R., Eckert, C. M., & Clarkson, P. J. (2006). Matrices or Node-Link Diagrams: Which Visual Representation is Better for Visualising Connectivity Models? Information Visualization, 5(1), 62–76.

NetworkX Dokumentation (2023). https://networkx.org. Besucht am 11.04.2023

NetworkX on GitHub (2023). https://github.com/networkx/networkx. Besucht am 11.04.2023

Neumann, K. (1989). Graphen und Netzwerke (pp. 1-164). Springer Berlin Heidelberg.

Paulus S. (2012). Basiswissen Sichere Software. Aus- und Weiterbildung zum ISSECO Certified Professionell for Secure Software Engineering. dpunkt.verlag. S.286.

Plotly Dokumentation (2023). https://plotly.com/python/. Besucht am 10.04.2023.

Plotly on GitHub (2023). https://github.com/plotly/plotly.py. Besucht am 10.04.2023.

PyGraphistry on GitHub (2023). https://github.com/graphistry/pygraphistry#install. Besucht am 23.04.2023.

PyGraphviz Dokumentation (2004). PyGraphviz. https://pygraphviz.github.io/documentation/latest/. Besucht am 23.04.2023.

PyVis Dokumentation (2016). Interaktive network visualizations. https://pyvis.readthedocs.io/en/latest/index.html. Besucht am 14.04.2023.

PyVis on Github (2018). WestHealth/pyvis. https://github.com/WestHealth/pyvis. Besucht am 14.04.2023.

Seaborn Dokumentation (2023). https://seaborn.pydata.org. Besucht am 21.04.2023.

Seaborn on GitHub (2023). Seaborn. https://github.com/mwaskom/seaborn. Besucht am 21.04.2023.

Sommerville I. & Sawyer P. (1997). Requirements Engineering: A Good Practise Guide. New York: Wiley.

synkAdvisor (2023a). Bokeh. https://snyk.io/advisor/python/bokeh. Besucht am 11.04.2023.

synkAdvisor (2023b). dash-cytoscape. https://snyk.io/advisor/python/dash-cytoscape. Besucht am 15.04.2023.

synkAdvisor (2023c). Holoviews. https://snyk.io/advisor/python/holoviews. Besucht am 21.04.2023.

synkAdvisor (2023d). igraph. https://snyk.io/advisor/python/igraph. Besucht am 22.04.2023.

synkAdvisor (2023e). NetworkX. https://snyk.io/advisor/python/networkx. Besucht am 11.04.2023.

synkAdvisor (2023f). Plotly. https://snyk.io/advisor/python/plotly. Besucht am 10.04.2023.

synkAdvisor (2023g). Pygraphistry. https://snyk.io/advisor/python/graphistry. Besucht am 23.04.2023.

synkAdvisor (2023h). pygrahviz v1.11. https://snyk.io/advisor/python/pygraphviz. Besucht am 23.04.2023.

synkAdvisor (2023i). PyVis. https://snyk.io/advisor/python/pyvis. Besucht am 14.04.2023.

synkAdvisor (2023j). Seaborn. https://snyk.io/advisor/python/seaborn. Besucht am 21.04.2023.

synkAdvisor (2023k). yfiles-jupyter-graphs. yfiles-jupyter-graphs. https://snyk.io/advisor/python/yfiles-jupyter-graphs. Besucht am 28.04.2023.

yFiles (2023). Product details. https://www.yworks.com/products/yfiles. Besucht am 28.04.2023.

yfiles-jupyter-graphs on github (2022). https://github.com/yWorks/yfiles-jupyter-graphs/tree/main. Besucht am 28.04.2023.

Zehnter C. (2012). Key-Performance-Analyse Von Methoden Des Anforderungsmanagements. (KIT Scientific Reports ; 7620). Universität Karlsruhe Universitätsbibliothek. S.109.

Anhang

Anhang 1: Projektauftrag

Projektauftrag	
Projektname	Dynamische Graphen mit Python
Ausgangslage	Die itopia AG verwendet als Tool für die Visualisierung von Graphen (statisch oder halbstatisch) aktuell Graphviz und/oder neo4j Bloom. Die Visualisierungen werden heute primär in Workshops genutzt, um Zusammenhänge und Abhängigkeiten aufzuzeigen. Die itopia AG möchte ein neues Tool einführen, welches neben der Darstellung von dynamischen Graphen auch die Möglichkeit zur interaktiven Nutzung dieser bietet.
Ziel	Das Projektziel ist, eine Empfehlung für die Einführung eines möglichen Open Source Tools (Python basiert) zu machen, welches interaktiv genutzt werden (Kommentar- und Einblendfunktionen) und neben statischen auch dynamischen Graphen erzeugen kann. Die Empfehlung geschieht in Form eines Anwendungsbeispiels, bei welchem das Tool durch den Einsatz synthetischer Daten des Auftraggebers demonstriert wird.
Abgrenzungen	1. Das Projekt sollte eine Analyse beinhalten, die sich auf praktische Anwendungsfälle bezieht. Hierbei sollte ein Evaluationsschema verwendet werden, das auf Kriterien basiert, die aus der Literatur abgeleitet wurden. 2. Das Python-Tool wird nicht unternehmensweit für das Tagesgeschäft implementiert, soll aber allenfalls für Testzwecke in einer realen Umgebung (evtl. innerhalb des Unternehmens) angewendet werden. 3. Die Projektergebnisse werden auf Tauglichkeit geprüft, um abschliessend eine angemessene Empfehlung abgeben zu können. 4. Synthetische Daten werden von itopia AG zur Verfügung gestellt
Meilensteine	● 17.02.23 - Startphase (Kick-off FHGR) ● 27.02.23 - Bedarfsklärung mit der itopia AG ● 03.03.23 - Abgabe Projektauftrag/Projektplan ● 16.03.23 - 2. Bedarfsklärung/Interview mit Auftraggeber ● 18.04.23 - Definition Anforderungskatalog ● 16.05.23 - Besprechung Zwischenergebnisse itopia AG und Referent ● 10.06.23 - Präsentation Projekt FHGR ● 11.06.23 - Bestimmung empfohlene Bibliothek ● 25.06.23 - Anwendungsbeispiel & Präsentation itopia AG erstellt ● 30.06.23 - Präsentation Projekt itopia AG und Referent ● 21.07.23 - Abgabe des Berichtes (FHGR)
Zeitaufwand	ca. 150 Stunden pro Studierende, Zugfahrt nach Zürich (nach Bedarf)
Projektorganisation	
Auftraggeber	itopia AG: Benjamin Schlup, Philippe Schädler
Referent	Prof. Dr. Wolfgang Semar
Projektleitung	Anna Kuriger, Mahmut Güner, Sara Koller
Unterschriften	_____ _____ Projektauftraggeber Referent/Projektleitung
Beilagen	Projektplan

Anhang 2: Anforderungskatalog

	Gewichtung
Nicht funktionale (Tool allgemein)	
Lokales, Open Source Werkzeug Die Netzwerkgrafik muss mittels eines Open Source Werkzeug generiert werden können. Die Anwendung soll lokal anwendbar sein. Die Netzwerkgrafik wird vorzugsweise mittels Python erstellt.	5
Sicherheit / Vertrauenswürdige Quelle In der Netzwerkgrafiken werden Kundendaten visualisiert. Aus diesem Grund muss beim Werkzeug die Datensicherheit gewährleistet sein. Somit muss die Python-Bibliothek Herkunft vertrauenswürdig sein.	5
Direkter Datenimport (CSV / JSON / Python Objekte) Für Workshops, in welchen zum Beispiel die IT-Netzwerke von Kunden abgebildet werden soll, werden die Netzwerke oft mittels CSV angeliefert. Andere Daten werden in Python synthetisch hergestellt. Integration von verschiedenen Dateitypen muss gegeben sein.	4
Netzwerkgrafikexport (PDF, SVG) Die Netzwerkgrafiken werden den Kunden zur Verfügung gestellt. Dies meist per Mail oder über Confluence. Aus diesem Grund sollten die Grafiken per PDF oder SVG exportierbar sein.	3
Funktionale (an die konkrete Netzwerkgrafik)	
Gestaltung	
Beschriftungsoptionen der Knoten In der Netzwerkgrafik sollen die gesamten Knotenbezeichnungen angezeigt werden. Zumindest mittels Hover-Funktion. Zudem werden lange Namen supportet. Optimal: Textfeld mit Formatierungsfunktionen	4
Knotengestaltung mit Farben Die Knoten in der Netzwerkgrafik müssen mindestens mittels Farbe unterscheidbar sein.	5
Knotengestaltung mit Formen Vorzugsweise sind die Knoten in der Netzwerkgrafik mittels Formen unterscheidbar (somit ist die Differenzierbarkeit für farbenblinde Personen sicherlich gegeben).	4
Kantengestaltung Die Kanten in der Netzwerkgrafik sollten mittels Dicke (Beziehungsstärke) unterschieden werden können. Vorzugsweise können Beziehungsrichtungen mittels Pfeile abgebildet werden. Und die Kantenfarbe (Beziehungsart) sollte verändert werden können.	4
Corporate Design Integration Möglichkeit Es soll ein Farbtemplate integrierbar sein.	2
Anwendbarkeit (Interaktivität)	
Zoom Möglichkeit In der Netzwerkgrafik sollte sich auf einen bestimmten Bereich konzentriert werden können, um diesen genauer zu besprechen.	4
Suchoption In der Netzwerkgrafik soll nach bestimmten Knoten gesucht werden können. Möglichkeit: Kann auch über Bowser funktionieren.	4
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit Für eine bessere Übersichtlichkeit in grossen Netzwerkgrafiken, sollen die Kanten, Knoten (nur Knoten mit gewissen Eigenschaften anzeigen) ein- und ausgeblendet werden können.	3
Übersichtlich Darstellung der Inhalte Optionen zur Beeinflussung der Darstellung durch Layouts. Wie Auswahl aus verschiedenen Layout Algorithmen. Manuelle Platzierung und Anpassung von Knoten.	4
Kommentarmöglichkeit In der Netzwerkgrafik sollen einzelne Bereiche kommentiert werden können. Möglichkeit: Objekterstellung und Verbindung.	3
Inputs entgegennehmen zum Beispiel Kanten und Knoten direkt erstellen, updaten, löschen.	3

Anhang 3: Verwendete Textstellen des Transkripts zum Experteninterview

Experte:

- Für synthetische Daten wird das Tool täglich verwendet. Im Beratungsmandat (Workshop): etwa 10 Mandaten pro Jahr.
- Für Beratungen, um bestehende Netzwerke abzubilden, arbeiten mit dem Tool 3-4 Berater (Technische Seite) und 10 Personen Management (Beratung).
- Man muss die Daten nicht direkt in der Grafik bearbeiten, es geht nur um Visualisierung. Die Daten werden separat aufbereitet und dann im bestimmten Format aufgeladen.

- Synthetische Daten werden direkt im Workshop verwendet. Für die Beratungen werden die Daten bereits im Voraus (grosse Daten) geschickt als CSV oder JSON Dateien. Daten werden ausserhalb der Grafik (mit Pandas in Python) aufbereitet und dann momentan zum Beispiel in neo4j gespielt. Es werden zum Beispiel zwei bis drei angelieferte Tabellen (Instanzen-Tabelle, Beziehungs-Tabelle (mother-child Beziehung)) über ID gemappt.
- Es gibt zwei Arten von Workshops, wo das Tool benutzt wird. Einerseits laufend mit den Kunden, andererseits auf Grund von gelieferten Datensätzen, welchen nur von itopia Mitarbeiter bearbeitet werden und danach den Kunden gezeigt werden.
- Ganze Ausschnitte, aber auch einzelne Punkte werden in der Grafik kommentiert beispielsweise mittels Farbe kategorisiert. Da dies aber nicht immer optimal ist (Stichwort "Farbenblindheit"), wäre die Kategorisierung mittels Formen besser.
- Neo4j ist ein Graphdatenbank, das wurde aber nicht so wirklich in dem Sinne gebraucht. Vorteile bei neo4j: Suchmöglichkeiten, Kanten können ein- und ausgeblendet werden können. Zoom Funktion ist praktisch.
  Was fehlt bei neo4j: begrenzte Farben- und Formen Auswahl von Knoten, Kommentarfunktion (wäre aber nur ein “nice to have”), Namen der Knoten sollten ganz eingeblendet werden können (wenigsten mittels Hoover), Beziehungstypen sind immer gleich und können nicht unterschieden werden. neo4j ist sehr langsam.
- Den Kunden werden die Grafiken oft über Confluence zur Verfügung gestellt. Die Grafiken werden zum Teil jetzt als PowerPoint geteilt (Nachteil: kann man keine Suchfunktion verwenden). Internet: Diagrams.net. Am PC: Visiolab. Wäre nett als PDF, SVG speichern zu können. Im Browser integrieren und anzeigen.

- Schliesslich ist es zwischen schön und brauchbar zu entscheiden. Brauchbare Darstellung wird bevorzugt (nach Benjamin).

- Ja, Zoomfunktion ist definitiv praktischer.

- Ja, es wäre gut das zu sehen oder auszublenden die Knoten, weil bei Grössere Datensätze wird schnell unübersichtlich.
- Die Grafiken werden in jedem Mandat anders gemacht (was um Farben, Formen… geht), und werden nach Kunden gerichtet, deswegen ist Corporate Design nicht ein Muss, sondern “nice to have”.

- **Resultat der Arbeit (nach Benjamin):** Soll am Beispiel von einem Tutorial sein. Wo man zeigen kann: wie sieht es optisch aus, aber gleichzeitig, was muss man machen, bis es dazu kommt. Kann an einem kleinen Beispiel sein. Dynamisch - sehr wichtiger Punkt.

Anhang 4: Nutzwertanalyse Gesamtübersicht

Anforderung	Gewichtung	Python Bibliotheken																									
		Bokeh		dash (Cytoscape)		PyGraphviz		Holoviews		igraph		networkX		Plotly		PyGraphistry		PyVis		Seaborn		yfiles-jupyter-graphs	
		Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert	Beurt.	Wert
Nicht funktionale (Tool allgemein)																							
Lokales, open Source Werkzeug	5	2	10	2	10	2	10	2	10	2	10	2	10	2	10	0	0	2	10	2	10	1	5
Sicherheit / Vertrauenswürdige Quelle	5	2	10	0	0	2	10	2	10	1	5	2	10	2	10	1	5	1	5	2	10	1	5
Direkter Datenimport (CSV / JSON / Python Objekte)	4	2	8	2	8	0	0	2	8	2	8	2	8	2	8	2	8	2	8	0	0	0	0
Netzwerkgrafikexport (PDF, SVG)	3	2	6	2	6	2	6	2	6	2	6	2	6	2	6	2	6	2	6	2	6	1	3
Gestaltung																							
Beschriftungsoptionen der Knoten	4	1	4	1	4	1	4	1	4	2	8	2	8	1	4	2	8	2	8	0	0	2	8
Knotengestaltung mit Farben	5	2	10	2	10	2	10	2	10	2	10	2	10	2	10	2	10	2	10	2	10	2	10
Knotengestaltung mit Formen	4	2	8	2	8	2	8	2	8	2	8	2	8	2	8	2	8	2	8	0	0	0	0
Kantengestaltung	4	2	8	2	8	2	8	2	8	2	8	2	8	2	8	2	8	2	8	0	0	1	4
Corporate Design Integration Möglichkeit	2	2	4	0	0	1	2	0	0	1	2	2	4	1	2	1	2	2	4	0	0	0	0
Anwendbarkeit (Interaktivität)																							
Zoom Möglichkeit	4	2	8	2	8	0	0	2	8	2	8	0	0	2	8	2	8	2	8	0	0	2	8
Suchoption	4	1	4	2	8	0	0	0	0	2	8	1	4	0	0	2	8	2	8	0	0	2	8
Ein- und Ausblenden von Objekten (Kanten, Knoten) Möglichkeit	3	1	3	2	6	0	0	1	3	2	6	0	0	0	0	2	6	2	6	0	0	2	6
Übersichtlich Darstellung der Inhalte	4	1	4	2	8	2	8	1	4	2	8	1	4	1	4	2	8	2	8	0	0	1	4
Kommentarmöglichkeit	3	0	0	0	0	0	0	0	0	1	3	0	0	1	3	0	0	1	3	0	0	1	3
Inputs entgegennehmen	3	1	3	1	3	1	3	0	0	1	3	0	0	1	3	1	3	1	3	0	0	1	3
Nutzwertsumme			90		87		69		79		101		80		84		88		103		36		67
Rang			3		5		9		8		2		7		6		4		1		11		10

Anhang 5: Beispieldatensatz mit Beziehungen von itopia AG

parent	relation_type	child
linuxappp-02	hosted-on	PRD1
linuxappt-02	hosted-on	PRD1
linuxappt-01	hosted-on	PRD1
linuxappp-01	hosted-on	PRD1
SAMPLEAPP (linuxappp-02)	runs-on	linuxappp-02
SAMPLEAPP (linuxappt-02)	runs-on	linuxappt-02
SAMPLEAPP (linuxappt-01)	runs-on	linuxappt-01
SAMPLEAPP (linuxappp-01)	runs-on	linuxappp-01
SAMPLESVC [PRODUCTION]	members	SAMPLEAPP (linuxappp-02)
SAMPLESVC [NON-PRODUCTION]	members	SAMPLEAPP (linuxappt-02)
SAMPLESVC [NON-PRODUCTION]	members	SAMPLEAPP (linuxappt-01)
SAMPLESVC [PRODUCTION]	members	SAMPLEAPP (linuxappp-01)
PRD1	members	esx01
PRD1	members	esx02
PRD1	members	esx03
PRD1	members	esx04

Anhang 6: PyVis Codebeispiel

```
net = Network(
   heading = 'Testdatensatz', bgcolor = '#F7F6F5', font_color = 'black', select_menu=True,
   filter_menu = True, height = '500px', width = '100%',)
node_types = {
   'platform': {'color': '#DDDCD8'},
   'application': {'color': '#BBDBF2', 'shape': 'diamond'},
   'service': {'color': '#FCBE10', 'shape': 'star'}}

# Netzwerkgrafik füllen
for index, row in df_merge.iterrows():
   mother = row['parent']
   child = row['child']
   relation_type = row['relation_type']

   if relation_type == 'members':
       edge_weight = 3
   elif relation_type == 'runs-on':
       edge_weight = 2
   else:
       edge_weight = 1

   for node in [mother, child]:
       node_info = df_info[df_info['name'] == node]
       node_type = node_info['ci_type'].values[0]
       productivity = node_info['ci_stage'].values[0]
       node_properties = node_types.get(node_type, {})
       node_label = node
       title = f"Type: {node_type}\nProductivity: {productivity}"
       node_size = 15 if productivity == 'NON-PRODUCTION' else 30
       net.add_node(node, label=node_label, title=title, size=node_size, **node_properties)
   net.add_edge(mother, child, title=relation_type, width=edge_weight, arrows='to')

# Integration Logo
logo = 'data/logo.png'
net.add_node('logo', shape='image', image=logo, label = ' ', x=-400, y=-400)

# manuelle Erstellung eines Knotens und Kante
net.add_node(100, label='ein anders \n verwendeter Knoten', shape='box', color='#B9C6B0')
net.add_edge(100, 'linuxappp-01')

net.show_buttons(filter_=['layout', 'interaction', 'physics'])
net.show('network.html')
```

Wir erklären hiermit, dass wir die vorliegende Studienarbeit selbständig, ohne Mithilfe Dritter und nur unter Benutzung der angegebenen Quellen verfasst haben. Ohne ausdrückliche Zustimmung des Referenten werden wir diese Arbeit nicht an Dritte aushändigen oder veröffentlichen.

Mahmut Güner	Sara Koller	Anna Kuriger