Knowledge Recombination and Diffusion in Patent Data
An Explorative Framework

Master Thesis

presented by
Alexander Haberling
Matriculation Number 1450868

submitted to

Data and Web Science Group
Prof. Dr. Heiner Stuckenschmidt
University of Mannheim

Chair of Organization and Innovation
Prof. Dr. Karin Hoisl
University of Mannheim

13. October 2021

Abstract

The dynamics of knowledge recombination and diffusion shape all progress made, and to be made in science, technology and culture, yet quantifying both is associated with enormous challenges and contested in innovation research. This thesis provides an explorative framework measuring these dynamics, utilizing probabilistic topic modeling, and network analysis. Additionally, a small introduction into the prediction of recombination aims to spark further research interests. The dataset employed consists of patents concerned with robot innovation, sampled from the PATSTAT database of the European Patent Office. The results classify the proposed measurements into two classes. The future incorporation of external robot innovation expertise is expected to grant further insights into the favorability of both classes and the presented approaches in general. The easy generalization of the methodology to other written knowledge records emphasizes the flexibility of the proposed framework.

Contents

1 Introduction **1**

2 Literature **3**
2.1 Heterogeneous Concepts 3
2.2 Prevailing Methods 5

3 Data **8**

4 Methodology **12**
4.1 Patent Abstract Preprocessing 13
4.2 Latent Dirichlet Allocation 13
4.2.1 Model 14
4.2.2 Evaluation 17
4.3 Data Transformation 19
4.3.1 Sliding Windows 19
4.3.2 Network Representation 20
4.4 Data Structure of Measurements 21
4.5 Direct Measurement 22
4.6 The Limited Community Detection Measurement 22
4.6.1 Community Detection 23
4.6.2 Identifying Community Cores 24
4.6.3 Community Tracing 25
4.6.4 Community Labeling 27
4.6.5 Measuring Recombination 28
4.7 The Simplified Community Detection Measurement 30
4.7.1 Topic Distribution Pruning 30
4.7.2 Simplified Recombination and Diffusion 31
4.7.3 Community Detection Algorithms 32
4.8 The Edge Weight Measurement 41

4.9 Comparative Analysis 41
4.9.1 Comparing SCMs 42
4.9.2 Comparing CCMs 44
4.10 Predicting Recombination 45
4.10.1 Node2Vec 46

5 Results **51**
5.1 Patent Abstract Preprocessing 51
5.2 Probabilistic Topic Modeling 53
5.2.1 Hyperparameter Estimation 53
5.2.2 Topic Descriptives 58
5.3 Data Transformation 61
5.3.1 Sliding Windows 61
5.3.2 Network Representation 62
5.4 The Direct Measurement 65
5.5 Community Detection Measurement 67
5.6 Edge Weight Measurement 74
5.7 Comparative Analysis 76
5.8 Predicting Recombination 82

6 Conclusion **85**

7 Limitations & Future Work **87**

A Additional Preprocessing Information **94**

B Additional LDA Grid Search Results **96**

List of Algorithms

1 Lais^2 - Link Aggregate Algorithm 39
2 Lais^2 - Improved Iterative Scan Algorithm 40
3 Node2Vec 49
4 Node2Vec - node2vecWalk 50

List of Figures

3.1 IPC Distribution 10
3.2 Patent Publications by Month 11

4.1 Graphical Representation of LDA 16
4.2 Bipartite Network Projection 21
4.3 Recombination in Crisp Communities 29
4.4 Recombination in Overlapping Communities 29
4.5 Degree Sequence 36
4.6 K-Clique Overlap Matrix 38

5.1 Grid Search Results Gensim: Number of Topics 55
5.2 Grid Search Results Mallet: Number of Topics 55
5.3 Grid Search Results Gensim: α 56
5.4 Grid Search Results Mallet: α 57
5.5 Grid Search Results Gensim: Optimization Interval 57
5.6 Grid Search Results Gensim: Training Iterations 58
5.7 Distribution Abstract Topic Possession 59
5.8 Distribution Topics Abstract Occurrences 60
5.9 Patents and Topics by Time 62
5.10 Network Densities 63
5.11 Excerpt SCM - Direct Measurement 65
5.12 Excerpt CCM - Direct Measurement 67
5.13 Community Detection Modularity 68
5.14 Excerpt SCM - Crisp Community Detection Algorithms 70
5.15 Excerpt SCM - Overlapping Community Detection Algorithms . 71
5.16 Excerpt CCM - Crisp Community Detection Algorithms 72
5.17 Excerpt CCM - Overlapping Community Detection Algorithms . 73
5.18 Excerpt SCM - Edge Weight Measurement 74
5.19 Excerpt CCM - Edge Weight Measurement 76

5.20 Aligned SCM Similarities 81
5.21 Aligned CCM Similarities 82
5.22 Node2Vec Loss 84

B.1 Extended Grid Search Gensim: Number of Topics 96
B.2 Extended Grid Search Mallet: Number of Topics 97

List of Tables

4.1 Topic Distribution Pruning 30
4.2 Normalization and Binarization of CMs 42

5.1 Abstract Preprocessing 52
5.2 Topic Descriptives 59
5.3 Slicing Window Descriptives 61
5.4 Network Descriptives 64
5.5 Aligned SCM and CCM diffusion pattern descriptives 77
5.6 Aligned SCM Similarities 79
5.7 Aligned CCM Similarities 80

Chapter 1

Introduction

Understanding how and when knowledge diffuses and recombines has troubled innovation research for decades. Accurately quantifying both grants a detailed picture of the dynamics at place, and facilitates a more effective allocation of research efforts, by identifying under-researched, yet promising areas. However, prevailing literature presents itself heterogeneous in the measurement of these fuzzy concepts.

The aim of this thesis is to contribute to this research area, by introducing an explorative framework quantifying knowledge recombination, its diffusion and the diffusion of single knowledge components. For this purpose, 3.781 patents concerned with robot innovation are sampled from the PATSTAT database provided by the European Patent Office. Subsequently, probabilistic topic modeling in the form of Latent Dirichlet Allocation (following LDA) is utilized to extract the topics within patent abstracts. After identifying these topics within each patent, three recombination and diffusion measurements are proposed.
The *Direct Measurement* lives up to its name and considers knowledge components to be the topics extracted from the patent abstracts. By simply measuring when which topic appears, diffusion patterns of combined and single knowledge components are identified.
For the other two measurements, the patents and their extracted topics are transformed into a bipartite network representation. Subsequently, this network is projected onto a unipartite patent network and a unipartite topic network.
The *Community Detection Measurement* was supposed to treat knowledge components as unobservables that are approximated by the communities found in the patent network projection and their topic distributions. This required the development of an algorithm tracing communities over different points in time. Although deemed unfitting, the build algorithm and its limitations are presented. Alterna-

tively, a simplified version of the *Community Detection Measurement* operationalizes knowledge components as the most prominent topic within a patent community.
At last the *Edge Weight Measurement* considers knowledge components to be active nodes within the topic network projection. A node is considered active, if it is engaged in at least one edge.
In addition to the three proposed measurements, a short introduction into the prediction of future knowledge recombination is given. This introduction understands the prediction of recombination as a link prediction problem within the topic network projection. This approach utilizes edge embeddings to predict the weight of topic combining edges.

In the following chapters, a concise review over relevant innovation literature is provided. Thereafter, the used dataset is presented. Next, the methodology is detailed in the same order it was presented in this introduction. The computed results are showcased correspondingly. At last the thesis is concluded by a discussion of the result, limitations and future prospects.

Chapter 2

Literature

The dynamics of knowledge diffusion and recombination have been of primary interest for innovation research for many years. Numerous publications have concerned themselvess with the identification of these dynamics, by incorporating patents as records of innovation. However, prevailing research is still lacking cohesive concepts for knowledge recombination and diffusion.

In the following sections, the heterogeneous use of both terms in existing literature is exemplified. Additionally, common methodological approaches are reviewed to illustrate the shortcomings of common practice and the novelty of the here proposed framework.

2.1 Heterogeneous Concepts

The multiformity of knowledge recombination and diffusion in innovation research is characterized by academic efforts focusing on either the former, the latter, or a concept representing a mixture of both. While publications concerned with the former or latter don't seem to contribute to the described heterogeneity, their shared operationalization does.

Exemplary, [43] is interested in diffusion only. In their paper, the effect of interpersonal networks on geographic based, and firm intern knowledge diffusion is estimated. For this purpose, diffusion is operationalization as a patent citation count dependent variable in a regression framework. Works solely focusing on knowledge recombination are represented by [17] and [16]. The former work defines innovation as a process of recombinant search. The authors analyze whether the increasing differentiation of technological knowledge challenges inventors. For

this purpose knowledge recombination is defined within a regressional framework as dependent variable of patent citation counts as well. [16] addresses the relationship between the novelty of knowledge components within recombination. While analyzing how novelty influences recombination failure and breakthrough, recombination is once again measured as a dependent, patent citation count variable in a regressional framework.

Popular concepts combining recombination and diffusion are exemplified with the notion of Technological Trajectories (following TTs). First brought forward by [20, 12], TTs are associated with technological change in a broad sense. Picking up this idea, [47] describes them as a combined process of radical breakthroughs, incremental innovations and diffusion. In their analysis on fuel cell patents they measured TTs as the main flow of ideas in a patent citation network. For this purpose the author introduces an algorithm constructing this path by combining edges with regard to their importance (i.e. centrality). [28] adapt the notion of [47] and consider TTs to be the main flow of knowledge predicated upon a technological paradigm. In their work, the influence of technological standards on emerging TTs is analyzed in patents concerned with machine to machine communication and the internet of things. The authors rely on an operationalization of TTs similar to [47]. Following this tradition, [15] instrumentalized a modified version of [47] to analyze TTs in biotechnology patents.

Contrary to what the previous paragraph might suggest, even the concept of TTs itself is heterogeneous. Exemplary, [44] refers to TTs as directions of technical development that are cumulative and self-generating. They investigate the moderating influence of TTs on firm-level determinants of innovation. Instead of developing a quantitative measurement for TTs, they utilize a taxonomy to categorize Greek firms into different kinds of TTs. Further research illustrating this point includes [11, 39, 18, 6] and can be found in the bibliography. Regardless of how TTs are measured or defined, the concept mostly comprises an entanglement of the creation of technological innovations and their diffusion.

Further efforts utilizing concepts mixing recombination and diffusion are found in [8, 31, 35]. [8] address technological diffusion and technological development with the concepts of core and emerging technologies. Both are identified in a patent citation network with regard to degree centrality and structural holes. In their analysis of Taiwanese patents, degree centrality is argued to measure inbound and outbound technological flow, and structural holes between patent classes are argued to indicated advantageous positions patents can take to broker information and to facilitate emerging technologies. [31] utilizes the same notion of core technologies in a patent citation network to analyze knowledge transfer efficiency and capability. With patents concerned with nanoscale science and engineering, citation graphs are constructed and compared with random graphs in order to evaluate efficiency

and capability. At last, [35] considers knowledge creation to be a path-dependent evolutionary process that involves recombining knowledge spread over time. By sampling pharmaceutical patents and employing a regression analysis, the impact of patents on future knowledge creation is measured by a dependent, patent citation count variable.

2.2 Prevailing Methods

For many years, innovation research was characterized by regressional analyses of patent citation counts. Prominent examples include [17, 16, 35, 43] listed in section 2.1. On a similar notion, [27], employs a panel regression to investigate the relationship between search behavior of firms and new product introductions. The novel contribution of the authors is to differentiate between search depth and search scope. The former addresses how frequently a firm is using existing knowledge. The later how widely a firm explores new knowledge. The data used was gathered in form of new product introduction announcements made by firms, and corresponding patent applications of European, American and Japanese industrial robot companies. With the help of a regressional analysis, the authors capture the influence of these search dimensions on new product introduction. The dependent variable was defined as a count of how often new products are introduced, combining old existing and novel knowledge.

However, focusing solely on patent citation counts entails major disadvantages. [43] states, that not all citations reflect knowledge flows. They remark, that citations might be included to avoid litigation or for other strategic reasons. The notion, that patent citations might not accurately reflect the knowledge diffusion that influence the inventor is supported by [3]. By sampling all patents released by the United States Patent and Trademark Office (following USPTO) between January 2001 and August 2003, they sample 442.839 patents and 5.434.483 citations to patents release prior to 2001. Analyzing this sample the authors report, that 40% of all citations were added by the patent office examiners, instead of the inventors. Moreover, 40% of all patents have all citations made by the examiners. On average examiners add 66 % of the reported citation to a patent. [46] validates this findings with a subsample of 2.670 papers and 31.377 citations sampled in the first week of January 2003. Additionally, they report different citation behavior between inventors and examiners in citing geographically similar patents. They finds that inventors are 20% more likely to cite patents with the same national origin than examiners. This effect is reported to be slightly higher for U.S. patents.

Tackling this issue, researchers have started to incorporate alternative, more sophisticated methods of measuring innovation processes. Advances in Natural

language Processing (following NLP) opened the door for a variety of semantical patent analyses of innovation [26, 25, 14]. [26] investigates the influence of knowledge recombination on the economic values of patents. After gathering 2.826 fullerene and nanotube patents from the USPTO throughout the year 2008, the corresponding abstracts were fed to a probabilistic topic model. By utilizing LDA the authors were able to identify novel ideas in form of topic originating patents. Subsequently, these originating patents are introduced to a negative binominal regression as a mediator. The economic value of a patent was operationalized once again as citation count. The authors conclude that economic value is tied to the recombination of distant ideas and that local search is associated with the emergence of novel ideas. Building on LDA as well, [25] introduce an approach of automatically constructing Knowledge Organisation Systems for patent data. For this purpose 1.364 patents concerned with Large Aperture Optical Elements, published between 2000 and 2011 are sampled from the Derwent Innovations Index. After clumping terms in patents titles, abstracts and TechFocus, topics are extracted via LDA. Therafter, K-Means clustering and Principal Component Analysis is used to refine the obtained topics. At last, [14] propose a framework for identifying matching patterns among coal-seam gas patents. For this purpose, 378 patents published between 2000 and 2018 and concerned with coal-seam gas are sampled from the Chinese Intellectual Property Office and the Derwent database. The topics extracted by LDA are used to analyse the depth and diversity of knowledge elements. Subsequently, association rules of knowledge elements are analysed.

Another track of innovation research incorporates ideas from network analysis. Popular works in this field include [13, 51]. [13] concern themselvess with emerging technologies similar to [8]. The focus of their work lies on identifying clusters of patents in patent citation networks, using community detection. These clusters are referred to as technological branches. The patents used for their analysis revolve around agriculture, food and textile and are sampled from the USPTO. Furthermore, the temporal change of the structure of these clusters is predicted. This is done relying on a predictor constructed for each patent. This predicting citation vector is based on the citations a patent receives from patents of various technological subcategories. [51] focus on patent value. The authors create four citation networks based on the citation data in 1.426 optical disk patents from the U.S. sampled between 2000 and 2010. The first network represents a traditional citation network. The second is an indirect citation network in which patents with intergenerational relationships are linked. The third expresses a co-citation network in which patents are linked if they make a common citation. The last of the four represents a co-citation network in which patents are linked if they receive a common citation. The construction of the networks is followed by a preprocessing that allows to integrate them into an aggregation network. By applying different

filter criterions on this aggregation network, the authors derive three subnetworks. These subgraphs are then integrated into a final comprehensive patent citation network. The characteristics of the described networks serve as predictors within a logistic regression. The target of the regression is reported to be an approximation of patent value in form of a dichotomous variable measuring the existence of triadic patent families.

Advances combining NLP and network methodology were made by [32, 9, 50]. [32] analyse papers concerned with patent mining. By calculating the citation eigenvector centrality of 143 papers from the Web of Science database, the authors identify the most influential papers in their sample. Additionally, a keyword network is constructed relying on keywords extracted from the paper abstracts. This network is then clustered, and inference about the evolution of the cluster over time is drawn from the mean value of publishing time within clusters. The three main clusters identified are patents concerned with bibliometrics and citation analysis, patents concerned with cluster and network analysis, and patents concerned with text mining, semantic and ontology analysis. Similar to [32], [9] relies on keyword networks and their clustering. 331 LED and 346 wireless broadband patents between 2000 and 2011 from the USPTO were used to gain insight into technology development efficiency. After the extraction of keywords from the patent abstracts, domain knowledge of experts was employed to standardize the keywords. Subsequently a weighted keyword network was constructed. By relying on correlation between keywords and label propagation community detection, communities were identified. The authors report several keyword and network characteristics and a faster rate of technological changes in wireless broadband compared to LED. At last, [50] constructs a weighted keyword network using 414 biofuel patents from the USPTO between 2000 and 2013. After extracting keywords from patent abstracts, domain knowledge from experts is utilized to refine the keyword sample. Based on this refinement a weighted keyword network is constructed to analyse network indices. Namely, a density-based index, an index measuring technological concentration and an index of technological importance relying on closeness centrality. By employing these methods, the authors outline the development in biofuel technology over the sample time span.

Few publications are utilizing both NLP in combination with network analysis. Of the here presented papers, none does so by extracting communities based on LDA topic extraction. Furthermore, no paper presents cohesive measures for both, knowledge recombination and diffusion. This thesis aims to contribute to the field of innovation research by bridging this gap.

Chapter 3

Data

The dataset utilized is a subset of the PATSTAT patent database of the European Patent Office (following EPO) and was provided by University of Mannheim's chair of Organization and Innovation. PATSTAT contains information concerned with the bibliography of patents and their legal events (i.e. publications, etc.) [36]. In it, every patent is classified with at least one International Patent Classification (following IPC). An IPC contains information about classification sections, classes, subclasses and groups. This structure is illustrated with the exemplary IPC H01S 03/14, to facilitate a better understanding of the following sampling and stochastic verification process.

The first position of every IPC indicates the broadest categorization. In this example, H classifies an invention into the area of Electricity. Further, the class in positions two to three of H01 narrows the scope to 'basic electric elements'. The fourth position in H01S identifies the subclass 'Devices using the process of light amplification ...'. At last, the digits in position five to eight categorize the invention into main and subgroups. H01S 03/00 refers to patents in the maingroup 'lasers', H01S 03/14 more specifically to lasers with a certain characteristic. [37]

The analysed PATSTAT subset contains patents concerned with cleaning robots and a broader set of robots sharing IPCs with cleaning robots. The patents are sampled in four steps. First, the IPCs of all patents resulting from the key word query 'cleaning robots' are extracted. Subsequently, these IPCs are used to sample all patents in the database with a priority date between 1. January 2000 and 31. December 2016, containing these IPCs. Thereafter, the sample is restricted to only patents with the word 'robot' in their abstract. At last, patent duplicates are identified via the 'family_id' variable provided by the EPO, and eliminated.

This procedure results in 3.844 patents. Each patent is characterized by their

patent id, publication date, title, abstract and IPCs. Out of these 3.844 patents, 48 German-language and 15 French-language patents are excluded in order to prevent the construction of suboptimal topics in later steps. Otherwise, the application of LDA topic modelling would lead to a clustering of German and French words into separate undesired, German and French topics, unrelated to the actual topical nature of the non-English patent abstracts. All of the remaining 3.781 English patents contain the word 'robot' in their abstracts. In 398 of them, the word 'clean' is referred to at least once. In the next paragraphs, the term 'patents' refers to these 3.781 surviving, English-language patents.

The IPCs contained in the dataset are truncated on maingroup level (after position six). Differentiating on this level is sufficient for thematic insight into each patent. Considering IPC subgroups as well would lead to a representation too fine-grained to be meaningful. On subgroup level, few patents would share IPCs.

Each patent contains at least 1 and at most 13 IPCs. The average number of IPCs a patent is classified with is 2,5, the median is 2 and the mode falls on 1. All patents together account for 9.449 IPCs. Broken down to the hierarchical classification levels, all 8 sections are represented in the dataset. On class level, 91 unique IPCs are observed. On subclass and maingroup level, 272 and 954 unique IPCs are present.

In order to find and exclude patents unfitting for this thesis, the patent distribution on section level is investigated more closely. All areas of innovation are covered by the eight following sections:

- **A** Human necessities
- **B** Performing Operations; Transporting
- **C** Chemistry; Metallurgy
- **D** Textiles; Paper
- **E** Fixed Constructions
- **F** Mechanical Engineering; Lighting; Heating; Weapons; Blasting
- **G** Physics
- **H** Electricity [37]

The histogram in figure 3.1 illustrates the overall distribution of patents per section. With the majority of patents located in sections A, B and G, all other less represented sections were investigated more closely. The stochastic sampling and manual evaluation of patents from sections C, D, E, F and H did not recommend the exclusion of additional unfitting patents. However, this conclusion was made

without extensive patent and IPC background knowledge, and a conservative data exclusion approach in mind. With more expertise, further patents might be excluded in future efforts.

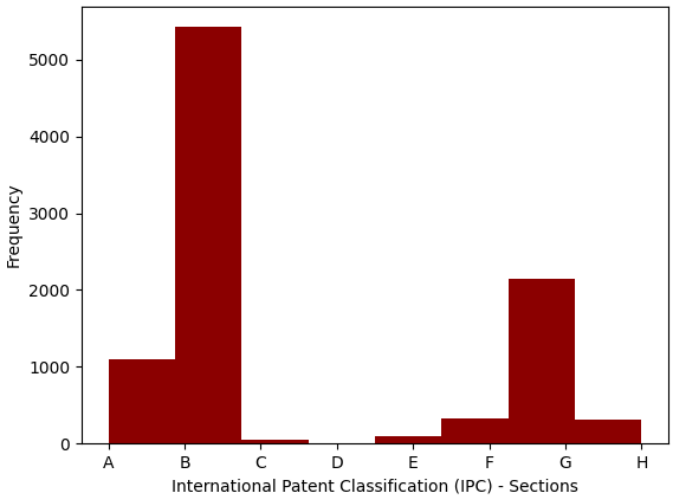

Figure 3.1: IPC distribution on section level. D is represented with two IPCs

The first publication, out of the 3.781 remaining patents dates to the 1. August 2001, the last publication to the 31. January 2018. This deviation from the previously described PATSTAT sampling time span is explained by the difference between publication date and priority date. The latter refers to the date a patent was first registered. This registration grants the patent holders right to priority. Cases in which multiple entities are trying to claim an invention are settled in favor of the entity with the oldest priority date. A more detailed breakdown is provided in [1]. Hence, an exemplary patent with a priority date in 2016 and publication in 2018 would be included in the dataset.

The time span between the earliest and last patent publications includes 6.027 days. However not on every day, publications were made. With 817 days on which publications were made, the average publication interval is 7,38 days. Figure 3.2 illustrates the relationship between time and patent publications in the described field. Over the observed span the amount of published patents concerned with (cleaning) robot innovation increases.

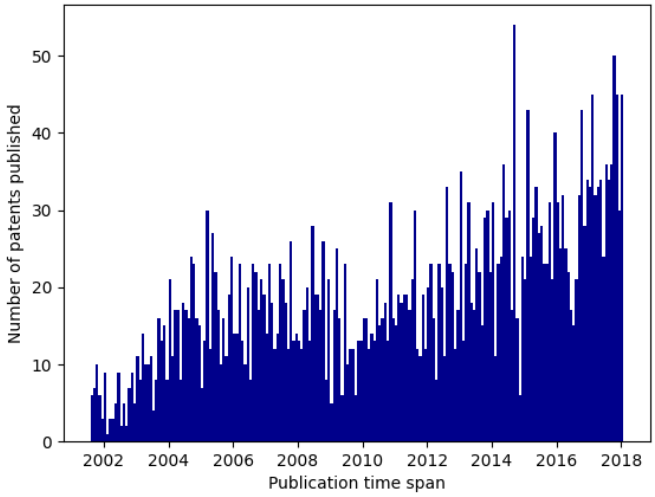

Figure 3.2: Patent publications contained in the dataset by month

Chapter 4

Methodology

In this chapter, all methods and steps involved in the proposed exploration are explicated. The three measurements and the prediction task are build on top of topics extracted by probabilistic topic modeling. Classifying patents via these extracted topics, instead of the already provided IPCs, yields a major advantage. The well established IPCs classify patents in a rigid and static manner. Whenever new patents are classified, prefabricated IPC templates are applied to categorize the knowledge contained in them. By relying on topic modelling, knowledge is not only not forced into prefabricated templates, but actively shapes the classes they are categorized in. This is argued to allow for a more dynamic and contextual identification of knowledge components.

During the first part, the extraction of topics from patent abstracts via LDA, and the necessary preprocessing preceding it are addressed. Subsequently, the process of transforming the patent sample into sliding windows and network representations of sliding windows is described. Thereafter, the first of the three measurements, the Direct Measurements relying solely on sliding windows is elucidated. Next, the community detection based measurement, relying on the network representation is introduced. Included in this section is the development of an approach connecting the detected communities between sliding windows. This connection is necessary to account for the time dependent aspect of diffusion. Following, the Edge Weight Measurement and its usage of an additional network representation is presented. A description of the steps taken to align and compare the results of the three measurements concludes this part. At last, an introduction to the prediction of recombination is presented.

4.1 Patent Abstract Preprocessing

Applying LDA or other forms of probabilistic topic modelling on the abstracts themselves would result in topics far less useful than they could be. Irrelevant and synonymous terms would render most extracted topics unfitting to represent underlying knowledge components contained in the patent abstract. Addressing this issue, several preprocessing steps are introduced to improve the quality of the topic modelling results.

In a first step, all non-alphabetic characters are removed. Characters and sequences like '&' and '(10)' are not expected to relate to patent innovation, and thus are considered meaningless. Subsequently, all single letter terms are removed from the abstracts, following a similar rationale. The characters 'x', 'y', and 'z' might be argued to carry information about a spacial categorization of movements and positions of entities. However, accompanying terms like 'axis' or 'three_dimensional' account for the same information and are not subject to this filter. Next, all terms are converted to be lower case only. This accounts for case sensitivity and prevents the model to differentiate between terms like 'Clean' and 'clean'. Both are synonymous, carrying the same information relevant for this use case. Thereafter, irrelevant stopwords are filtered out and bigrams are extracted. This facilitates the forming of only innovation relevant bigrams. At last, lemmatization was applied on all abstracts and stopwords were filtered out once again. Lemmatization prunes all inflected forms of a word to one single term. By doing so, terms carrying very similar information are once again subsumed into one item. The filtering of stopword after lemmatization catches stopwords previously undetected, due to a (suffix) variation. Exemplary the term 'generally' is missed by the stopword 'general' before lemmatization, but not after.

4.2 Latent Dirichlet Allocation

Following [26, 25, 14], this thesis aims to extract knowledge components contained in patent abstracts. The application of LDA on the patent abstracts, provides the topics found over all documents, and the respective document-topic affiliations. Topics themselves are represented as sequences of terms and their corresponding topic-specific weights. These relative weights indicate how prominent a term is in a topic. The document-topic affiliation connects every document with the respective topics found in it. Additionally, topic weights are provided indicating how prominent the affiliated topics are within a document.

The major advantage of LDA and other probabilistic topic models compared to text classification methods, lies in its unsupervised nature. LDA does not require

prior annotation of data, making it attractive for scaling analyses and the here presented framework. However, this advantage is coupled with a challenging model evaluation.

In the following sections, the model itself and an approach tackling this challenging evaluation is presented.

4.2.1 Model

LDA is a generative probabilistic model aiming to discover topics within a set of documents. It relies on the assumption, that every document consists of a finite mixture of topics and that every topic consists of a finite mixture of words. The generative process assumes that every document arose in three steps. First, for each document a distribution over topics is randomly chosen. Second, for each word to be generated in this document, a topic from this topic distribution is randomly chosen. And third, from this chosen topic, a term of its vocabular distribution is randomly chosen in order to generate the actual word. When utilizing LDA, the only observables in this process are the already finished documents themselves. However, the primary interest in the presented use case are the hidden topic-concerned factors. These variables are usually referred to as hidden or latent variables. By reversing the generative process, LDA tries to infer the nature of these hidden, topic-related variables that generated the observed documents [5]. Inferring these hidden variables is argued to approximate the knowledge components contained in patent abstracts.

In the following paragraphs, the generative process is described more formalized. Thereafter, the variable dependencies are illustrated with a graphical representation. At last, the computational challenges of inferring the hidden variables are addressed.

Let:

- w be a word
- z be the topic of a word
- N be the number words in a document
- M be the number of documents in a corpus
- $\mathbf{w} = (w_1, w_2, ..., w_N)$ be a document
- $D = (\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M)$ be a corpus
- α be a Dirichlet prior
- θ be a multinominal per-document topic distribution
- β be a multinominal per-topic word distribution

Then, each document **w** in a corpus D is assumed to be generated by:

1. Choose $N \sim$ Poisson(ξ)
2. Choose $\theta \sim$ Dir(α)
3. For each of the N words w_n:
   (a) Choose a topic z_n from the multinominal per-document topic distribution θ.
   (b) Choose a word w_n from $p(w_n|z_n, \beta)$, a multinominal probability conditioned on the topic z_N. [5]

Within LDA, the joint distribution of the multinominal per-document topic distribution θ, a set of N topics **z** and words **w** is formulated as:

$$p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta) = p(\theta \mid \alpha) \prod_{n=1}^{N} p(z_n \mid \theta) p(w_n \mid z_n, \beta)$$

When integrating over θ and summing over z, the marginal distribution of a document $p(\mathbf{w} \mid \alpha, \beta)$ is obtained.

$$p(\mathbf{w} \mid \alpha, \beta) = \int p(\theta \mid \alpha) \left(\prod_{n=1}^{N} \sum_{z_n} p(z_n \mid \theta) p(w_n \mid z_n, \beta) \right) d\theta$$

By taking the product of the marginal probabilities of all documents, the probability for the whole corpus equals:

$$p(D \mid \alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn} \mid \theta_d) p(w_{dn} \mid z_{dn}, \beta) \right) d\theta_d$$

A more detailed description of the generative process can be found in [5].

The dependencies between all variables are illustrated in figure 4.1. The coloring of a node indicates whether a variable is observable or hidden. Unshaded nodes represent hidden variables, while shaded nodes represent the contrary. The plate M denotes the documents in a corpus, while N denotes the repeatedly sampled topics and words within a document. The multinominal topic distribution θ of each document is drawn with respect to the Dirichlet prior α. The topic z for each word is drawn with respect to θ, and the words w are drawn with respect to topic z and the word distribution β of each topic [5].

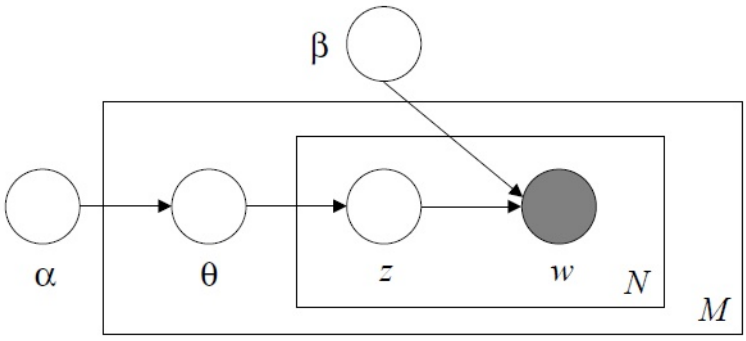

Figure 4.1: Graphical representation of LDA illustrating variable dependencies (source: [5])

In order to infer topics, the posterior distribution of the hidden variables θ and **z** needs to be computed.

$$p(\theta, \mathbf{z} \mid \mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta)}{p(\mathbf{w} \mid \alpha, \beta)}$$

However, [5] reformulate this equation to show that computing this posterior distribution is intractable, due to the coupling of θ and β in the sum of the latent topics.

$$p(\mathbf{w} \mid \alpha, \beta) = \frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) d\theta$$

Common approaches of overcoming this limitation involve approximating the posterior distribution with Variational Inference or Gibbs Sampling. A more detailed description of both can be found in [49] and [21].

4.2.2 Evaluation

When applying LDA, the hyperparameters α, β and the number of topics must be specified in advance. However, the unsupervised nature of topic modeling renders a search for optimal hyperparameter values challenging. Confronting this issue, the notion of Perplexity was proposed to evaluate the quality of the resulting topics [48]. Despite its popularity in other NLP branches, it was shown to be negatively correlated with human judgement of topic quality [7]. In order to overcome this limitation, alternative measurements of topic coherence have been proposed. Coherence refers to the semantic similarity between the top N highest scoring words within each topic. Words are considered semantically similar, if they occur in similar contexts [45]. Of the various existing coherency measures, C_V is reported to achieve high correlation with human topic ranking data [42]. The following description of C_V is based on [42, 45, 30]

C_V combines the notions of cosine similarity, normalized pointwise mutual information (following NPMI) and a boolean sliding window approach [42, 45, 30].

In order to formalize C_V, let:

- N be the number of topics
- t be a topic
- N_t be the number of top words in topic t
- w_i be the ith word in topic t
- W_t be the set of all top word in topic t

For every top word in every topic, the cosine similarity between the context vector w_i and the context vector W_t is calculated.

$$C_V = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{N_t}\sum_{i=1}^{N_t} s_{cos}(\vec{v}_{NPMI}(w_i), \vec{v}_{NPMI}(W_t))$$

The context vector of w_i is characterized by the NPMI score of this word and all other words in W_t.With $w_j \in W_t$ for each dimension j.

$$\vec{v}_{NPMI}(w_i) = \Big\{NPMI(w_i, w_j)\Big\}_{w_j \in W_t}$$

The context vector of W_t is stated in a similar manner. The value of each dimension j is defined as the sum of all NPMI scores calculated between all N_t, and w_j.

$$\vec{v}_{NPMI}(W_t) = \Big\{\sum_{w_i \in W_t} NPMI(w_i, w_j)\Big\}_{w_j \in W_t}$$

These similarity scores are then averaged for each word w_i within a topic. Subsequently, these averages are averaged for every topic [42, 45, 30].
The normalized pointwise mutual information score is calculated as:

$$NPMI(w_i, w_j) = \left(\frac{\log \frac{P(w_i,w_j)+\epsilon}{P(w_i)P(w_j)}}{-\log(P(w_i, w_j)+\epsilon)}\right)$$

The probability of a single word $p(w_i)$ and the joint probability of two words $p(w_i, w_j)$ is estimated using Boolean document calculation. This refers to the number of documents a word w_i or two words w_i, w_j appear in, divided by the total number of documents. This calculation on its own fails to consider how often a term occurs in a document and how distant terms are to each other. In order to overcome this, each document is used to generate a set of virtual documents of smaller size, in a sliding windowed manner. Exemplary, a document **w** would be split into virtual documents $\mathbf{w}_{virtual1} = \{w_1, ...w_s\}$, $\mathbf{w}_{virtual2} = \{w_2, ...w_{s+1}\}, ...$ with s representing the size of the sliding window [42, 45, 30].

The nature of cosine similarity causes the value range of C_V to fall between 0 and 1. Higher values indicate a higher coherence of topics and a higher correlation with human judgement [42, 45, 30].

4.3 Data Transformation

To this point, unfitting patents have been dropped, the patent abstracts were cleaned and preprocessed, and topics were extracted using LDA. For the upcomming measurements and the prediction task to be applicable, the dataset has to be further transformed in two key ways. First, the time dimension in the patent publication dates has to be used to aggregate patents into sliding publication windows. Second, additional network representations for each sliding window have to be created. The following sections describe these transformations in more detail.

4.3.1 Sliding Windows

Grouping patents into publication windows allows for the construction of networks big enough to detect meaningful communities in. When not aggregating, an average of only 4,63 patents are published on every publication date ($\frac{3781}{817}$, see chapter 3). Detecting meaningful communities in networks with roughly five nodes is deemed to fail. Even when sampling significantly more patents, and hence bigger networks, this approach is expected to fail capturing important topical similarities between patents. This is illustrated with an exemplary patent A published at t_i, a patent B at t_{i+1} and a patent C published at t_{i+2}. Without aggregation, these patents are not represented in the same network. Topical similarities between the patent nodes A, B and C in form of connecting edges are not represented, and will be overlooked by a community detection algorithm.

Furthermore, grouping patents into sliding (overlapping) publication windows, instead of disjointed publication windows allows for a more differentiated registration of change in knowledge component appearance. Let three patents be published on t_i and three other patents be published on t_{i+1}. Whenever a disjointed window approach splits the dataset between t_i and t_{i+1}, possibly interesting patterns between these six patents are lost. A brief explanation of the sliding windows approach is given at the end of section 4.2.2, with respect to virtual documents.

The benefits of windowing and sliding were elucidated above in the context of the Community Detection Measurement. However, these advantages carry over to the other two measurements as well. Additionally, applying this procedure for every measurement aligns the data structure and allows for an easier comparison between the three measures.

4.3.2 Network Representation

The network representations of the sliding windows, required for two of the three measurements, are constructed in two steps.

First, for every sliding window, a bipartite patent-topic network is built. In order to draw edges between topic and patent nodes, the computed document-topic affiliation of LDA is utilized. The edges in this bipartite network are weighted by the respective affiliation weights.

Second, the bipartite network is projected into a patent network in which edges indicate topical similarity, and a topic network in which edge indicate patent-similarity. The latter can be thought of as an abstract measurement of topic co-occurrence. The transformation of bipartite edges to unipartite edges within a projection is handled by a projection function. Common functions like [53] struggle to account for bipartite edge weights. Utilizing such projection functions would necessitate the binarization of the bipartite edges along an edge weight threshold. This binarization would result in the loss of significant information. Circumventing this loss, a projection function specific to this framework is proposed and provided below:

$$edge_{vu} = \frac{1}{N} \sum_{i \in N}^{N} (edge_{vi} * edge_{ui})$$

v and u indicate nodes of the same type in a bipartite network. N indicates the number of shared neighbors between nodes v and u. Multiplying $edge_{vi}$ and $edge_{ui}$ instead of adding them up carries the advantage of penalizing skewed edge ratios. When adding up, an $edge_{vi}$ of weight 0,7 and an $edge_{ui}$ of weight 0,1 would result in the same weight for $edge_{vu}$ as a more balanced ratio of 0,4 and 0,4. This is undesired. Arguably, two patents connected to the same topic with 0,7 and 0,1 are expected to be less thematically similar then two patents connected to a topic via 0,4 and 0,4. The same argument is made for two topics connected to a patent.

The projection of a bipartite network into to unipartite networks is illustrated in figure 4.2. In it, an exemplary patent with id 3 exhibits topics A and B with a coverage of 0,5 and 0,2 respectively. After the patent projection, patent 3 is connected to patents 1, 2 and 4, because they share the same topic A or B. In the topic projection, topics A and B share a link, due to their occurrence in patent 3.

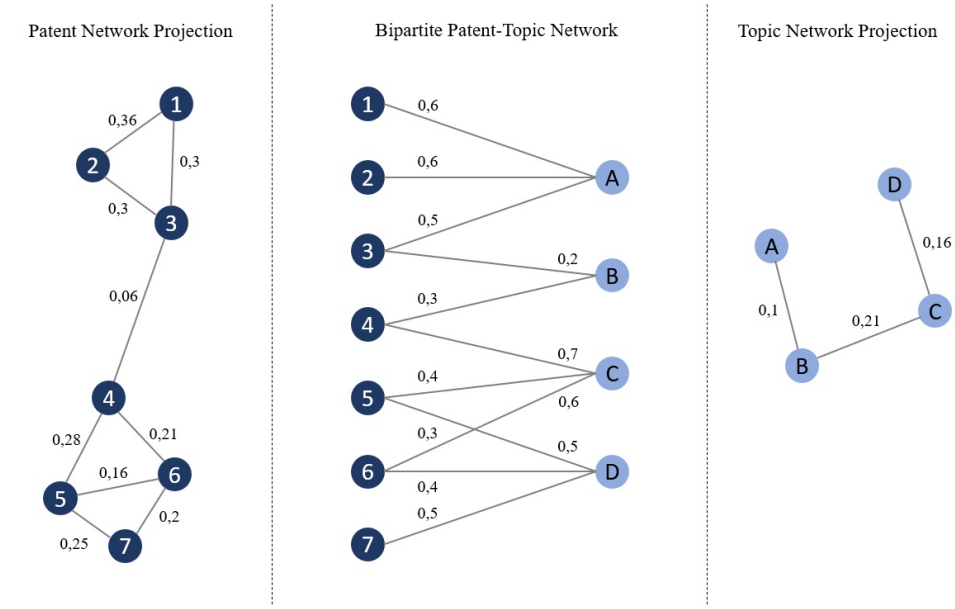

Figure 4.2: Illustration of a bipartite network and its two projections

4.4 Data Structure of Measurements

The explorative nature of this thesis motivates for a comparative evaluation of the proposed measurements. In order to facilitate this comparison between the three measurements, all are designed to produce the same data structures. All measures result in:

- 1. a $SingleComponentMatrix$ (SCM) - a pattern matrix in which the diffusion patterns of (single) knowledge components are represented.
- 2. a $CombinedComponentMatrix$ (CCM) - a pattern matrix in which the recombination and diffusion patterns of knowledge component combinations are represented.

All pattern matrices are characterized by the dimensions i - the number of sliding windows, and j - the number of all knowledge components or combinations observed over all sliding windows within a measure.

4.5 Direct Measurement

Within the Direct Measurement, the knowledge components of a patent are defined as the topics extracted from it, by the previously presented LDA model. Subsequently, the construction of the SCM is rather straightforward. First, all knowledge components (i.e. topics) over all patents and sliding windows are extracted. The number of these components equals the desired number of topics previously given to the LDA as hyperparameter (unless the LDA produced topics that are not covered by any patent). With this extraction the dimensions of the SCM are defined. i equals all sliding windows, sorted ascending by window id. j equals all extracted knowledge components sorted ascending by component id (i.e. topic id). Secondly, the cells SCM_{ij} are filled with the number of times component j occurred in sliding window i.

Knowledge component combinations are defined as topic combinations occurring within a patent. In order to construct the CCM, all component combinations within all patents of all sliding windows are extracted. This process is illustrated by a simplified case of two windows containing two patents each. Let patent 1 in window I cover topics A, B and C and patent 2 in window I cover topics D, E, and F. Additionally, let patent 3 in window II cover W and V and patent 4 in window II cover X, Y and Z. The extracted combinations from window I would be (A,B), (A,C), (B,C), (D,E), (D,F), (E,F) and the combinations from window II would be (W,V), (X,Y), (X,Z), (Y,Z). The overall combinations extracted from the dataset would be (A,B), (A,C), (B,C), (D,E), (D,F), (E,F), (W,V), (X,Y), (X,Z), (Y,Z), if no further windows are present.

With the extraction of all combinations of all windows, the dimensions for CCM are set. Once, again i equals the ascending sorted sliding windows, while j equals the ascending sorted list of extracted component combinations. The population follows in the same manner as in the SCM. Every cell CCM_{ij} is filled with the number of times component combination j is observed in sliding window i.

4.6 The Limited Community Detection Measurement

This measurement was conceptualized to treat knowledge components not to as singular topics, but as unobservables that are approximated via patent communities and their corresponding topic distributions. For this purpose, these patent communities were extracted from the patent network projection of the bipartite patent-topic networks. Subsequently, the diffusion of these unobservables and the

diffusion of combinations of these unobservables were planned to be measured by tracing their approximating patent communities over the sliding windows. This tracing was considered necessary because slight variations in the communities and the corresponding topic distributions over time, were argued to still approximate the same underlying knowledge components. Once these traced community sequences would have been established, component combinations were supposed to be identified via the overlap between communities and community bridging patent nodes.

However, the developed approach supposed to trace the patent communities is flawed and fails to appropriately account for community splits. Despite its flaws, the original procedure and argumentation is described in the following subsections. This is done to inspire future efforts aiming to refine this process.
Subsequently, an adjustment of the argument and a methodological solution circumventing these flaws is presented in section 4.7.

4.6.1 Community Detection

The data transformation in section 4.3 results, in i patent networks (with $i =$ number of sliding windows). For the identification of the described approximations of the knowledge components, communities are extracted in each patent network. In order to find a suitable community detection algorithm (following CDA), two crisp (disjointed) and two overlapping algorithms are employed. The former detects communities in which a node is part of at most one community, while the later identify communities in which a node can contribute to more than one community. Every algorithm applied represents a different school of CDAs. Label Propagation utilizes a threshold-based diffusion dynamic, Greedy Modularity is focused on the measurement of Modularity, K-Clique explores local graph structures and Lais^2 improves upon centrality measures. The square of Lais^2 does not represent a footnote, but is part of the algorithm name.

The Community Detection Measurement is computed for each CDA separately, resulting in four SCMs and four CCMs. Each of the following steps in applied to one of four identical sets of patent networks described in section 4.3.2. The only difference between these sets lies in the CDA applied and the resulting communities.
A detailed breakdown of each employed CDA is given in the subsections concluding section 4.7.

After their detection, communities with less than three member nodes are excluded. These communities are not expected to sufficiently approximate knowledge components.

4.6.2 Identifying Community Cores

All of the presented CDAs are of cross-sectional nature. They take a snapshot of a network (i.e. the representation of one sliding window) and identify communities in it. These methods are powerful, yet not sufficient on their own, for measuring the longitudinal aspect of diffusion and recombination dynamics.

The problem at hand is illustrated with the following example. Let A, B, C, D and E be patents in the dataset and let a CDA identify the communities (A, B, C, D, E), ..., (...), in sliding window t_i and the communities (A, B, C, D), ..., (...) in window t_{i+1}. By applying human judgement it is trivial to connect the two communities (A, B, C, D, E) and (A, B, C, D). However, manually evaluating and connecting all detected communities in all sliding windows over all CDAs seems infeasible, especially if further efforts scale the analysis up to a significantly larger set of patents. The reasons for this are the community dynamics at place. Over the course of the sliding windows, communities grow, shrink, emerge, disappear, merge and split.

The proposed automated approach consists of three steps. First, for each detected community in every window, a community core is identified. Next, these cores are connected between neighboring sliding windows, if appropriate conditions are meet. At last, identified core sequences are labeled with time invariant, static community ids.

Community cores are defined as the set of N patents with the highest centrality in their community. These patents are chosen because they are least likely to leave the community in the next time step (i.e. the next sliding window). In cases in which they are leaving (e.g. by merging into another community), it is most likely that a majority of their community already left or merged into other communities before them.

For overlapping CDAs, this set is additionally restricted to patent nodes that are not part of a community overlap. This restriction is softened for communities resulting from the Lais^2 algorithm, since Lais^2 is prone to identify communities that are complete subsets of other communities, and hence complete overlaps. These cases are handled by allowing the set to contain overlapping patent nodes. This exception is the first of many cases following, in which Lais^2 demands special adjustments, indicating its unsuitability for the proposed framework.

Once all cores are identified, communities with the same core, meaning communities that are complete subsets of other communities or completely contain other communities, are merged.

Due to the relatively small average community size (see section 4.6.1), the set size N is set to 1 and the degree centrality is chosen. [1]

4.6.3 Community Tracing

The tracing of the detected communities relies on the most central patent nodes (i.e. cores) of each community and a tracing matrix TM. In it every row i represents a sliding window analogous to the already introduced SCMs and CCMs. Every column represents a sequence of community cores defining a community over time. In the following, the terms core and core id are used synonymously. In cases in which the core size is set to one, both are referring to the most central, identifying node in a community.

For crisp communities, the TM is constructed as follows: First, TM_0j is initialized with the core ids of all communities detected in the first window (e.g. $TM_{00} \leftarrow$ core id 1; $TM_{01} \leftarrow$ core id 2; $TM_{02} \leftarrow$ core id 3; and so on). Thereafter, by iterating over i, the community cores in sliding window i are connected to community cores in $TM_{i-1,j}$, if they fulfill a matching criterion. The community cores in sliding window i that are not linked to any cores in $TM_{i-1,j}$ are considered to mark new emerging communities. For these cores, new columns (i.e. core sequences) are opened, similar to the initialization step ($TM_{ik} \leftarrow$ new unconnected core, with $k =$ number of already existing community sequences).

The rules by which cores are connected are illustrated with the following prototypical cases.

1. A core in $TM_{i-1,j}$ is part of a community in sliding window i. For this simple case, both core ids are linked, meaning the core id of the community in which $TM_{i-1,j}$ is found is entered below in $TM_{i,j}$.
2. A core in $TM_{i-1,j}$ is not part of a community in sliding window i. In this case, all nodes of the community in window $i-1$ for which $TM_{i-1,j}$ is the core of, are sorted descending by their centrality. Following this order, every node is checked, whether it is part of a community in window i. Once this is the case, the core of this community in window i is linked to the core in $TM_{i-1,j}$ (i.e. the found core is entered below in $TM_{i,j}$). [2] If none of

[1] Future efforts might set N dynamically with respect to the community size and explore different centralities.

[2] Instead of simply taking the community in which the next highest degree node is found, future efforts might retrieve the membership of all sorted nodes and weight them by their centrality. Ulti-

the sorted nodes can be found in a community in window i, then the core sequence j ends. The community later associated with this core sequence is considered dead at time i.

Overlapping communities are handled in the same spirit. However, more edge cases are considered. The initialization of TM, and the opening of new community sequences are identical. Additionally, the rules by which cores are linked, are extended.

3. A core in $TM_{i-1,j}$ is part of more than one community in sliding window i. Here, the core $TM_{i-1,j}$ is not considered to be identifying anymore, so it is disregarded and treated as case 2, meaning all other nodes of its community in window $i-1$ are sorted and checked for memberships.
4. A core in $TM_{i-1,j}$ is not part of a community in sliding window i, but part of more than one community in window $i-1$. In this case simply the biggest of these communities is chosen and case 2 is applied. [3]

Additionally, case 2 is modified in the following manner. Instead of checking whether the descending sorted nodes are part of a community in window i, it is checked whether they are part of one and only one community in window i. Nodes that are part of two or more communities in window i are disregarded and treated as if they would be part of no other community. If none of the sorted nodes can be found in one and only one community in window i, then the core sequence j ends.

This procedure results in a TM in which community cores can identify multiple core sequences within a window i. These cases can be interpreted as merged communities. Once two or more communities, separated in window $i-1$, merge in window i, the core sequences align. The community behind the aligned sequences is later labeled with only one static community id, and hence only associated with one knowledge component approximation. However, the alignment still presents a problem. In cases in which a merge is followed by an analogous split, seceded cores are unable to be linked again to the pre-merge core in TM_{i-1}.

Community splits in general are accounted in the TM by identifying the fragment that contains the most central node that was already present in the previous

mately, the community core with the highest weight sum might be used for linking. Additionally, this process might be tightened by not considering all sorted nodes, but only the h most central nodes. With h being defined dynamically with regard to the community size or degree distribution.
Both approaches are expected to preform similar in the presented use case.

[3] Instead of simply choosing the biggest community, future efforts might choose the community in which $TM_{i-1,j}$ is most rooted in. This evaluation can be done by checking for the fraction of edges or edge weights reaching from $TM_{i-1,j}$ to the respective communities.

community union. In the presented patent use case, this fragment is the biggest one, most of the time. Subsequently, this biggest fragment is linked to the core of the previous union. All other fragments are treated as new knowledge component approximations and new core sequences are opened for each of them.

This is where the proposed approach is limited. Consider two communities merging, and splitting again shortly after. If these two communities are similar to the communities before the split, then they should be connected to their respective sequences before the split. Furthermore, two communities merging should be represented in a new sequence as well, if both communities are of similar size, but dissimilar topic distribution. With the presented, the core sequences are not suited to confidently approximate the described knowledge components.

In the following, the presentation of The Limited Community Detection Measurement is continued, to facilitate future efforts aiming to refine it.

4.6.4 Community Labeling

The linked cores in the TM serve as dynamic ids. For each sliding window in the dataset a community is identified with one unique core id. However, these core ids change over time. This section is concerned with the assignment of time invariant, static community ids to all core sequences. These static ids are referred to as community labels.

As in the previous section, the rules with which static ids are assigned to community sequences are illustrated by the discussion of prototypical cases. Labels are assigned to every unique core id in a window by iterating through all rows i and sequences j in the TM.

The crisp community case:

1. A core id is present in one and only one column j of TM_i. Then the core corresponding community ij is labeled with the corresponding column id j.
2. A core id is present in more than one column $j_{1:k}$ of TM_i, with $k =$ the number of columns the core id appears in.
   (a) And the core id was already present in $TM_{i-1,j_{1:k}}$. Then the corresponding community $i, j_{1:k}$ is labeled with the same label assigned to the core id in TM_{i-1}.
   (b) And the core id was not already present in $TM_{i-1,j_{1:k}}$. Then the communities of all cores $TM_{i-1,j_{1:k}}$ and the current community with core $i, j_{1:k}$ is retrieved. After retrieval, the nodes of the current community are sorted descending by node centrality. By traversing, these ordered nodes are checked for their membership in one and only one of any

of the communities, associated with the cores $i-1, j_{1:k}$. The previously assigned label to the first community found containing one of the sorted nodes, is copied for the current community $i, j_{1:k}$.

For the overlapping case, the same rules apply. Additionally, 2. (b) is modified in the following way. If none of the sorted nodes of community $i, j_{1:k}$ is part of one and only one of the communities associated with $i-1, j_{1:k}$, then the label of the most frequently occurring community over all node checks is used to label the current community $i, j_{1:k}$.

After the establishment of these static labels, the evolving communities can be identified over all sliding windows. In order to measure the diffusion of the knowledge components approximated by these communities, a SCM can be populated. In it every cell SCM_{ij} would be filled with zero or one, indicating whether the approximating community j was active in window i or not. A creation of a SCM based on this community tracing was refrained from for the previously mentioned limitations of community tracing.

4.6.5 Measuring Recombination

The measurement of recombination is dependent on the CDA applied. When retrieving crisp (disjointed) communities, then recombinations are defined by the emergence of brokering patent nodes. When retrieving overlapping communities, they are defined by the overlap between communities. In the next paragraphs, both operationalizations of recombination are presented in more detail.

In the patent network nodes represent patents, edges between nodes represent the topic similarity between the connected patents and communities are considered to approximate knowledge components. Measuring recombination of knowledge components in this setting intuitively revolves around the connection of communities.

In a crisp community setting, patent nodes that exhibit edges connected to nodes in two or more communities are considered to recombine the knowledge components these communities are approximating, see figure 4.3. These brokering patent nodes are identified in two steps. First, all new patent nodes joining the patent networks between two sliding windows $i-1$ and i are gathered. Second, these patents emerging in window i are investigated for their edges to nodes that have been part of communities in $i-1$.

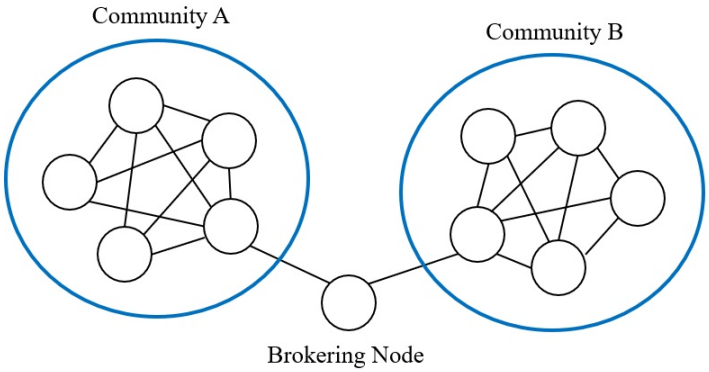

Figure 4.3: Recombination (i.e. a recombining patent node) in a crisp communities setting

Identifying recombining patents in an overlapping community setting is even more straightforward. In every window i all patent nodes that are part of more than one community are considered recombining patents [4] , see figure 4.4.

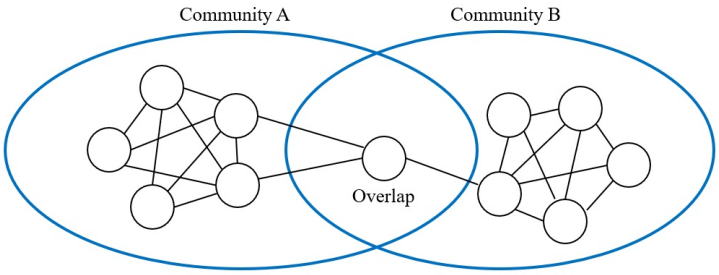

Figure 4.4: Recombination (i.e. a recombining patent node) in an overlapping communities setting

Once all recombinations of knowledge components are identified, a CCM can be populated in the usual manner. In every cell CCM_{ij}, the count of the knowledge component combination (i.e. the count of recombining patents) j in window i can be inserted. However, a construction of a CCM based on the limited community

[4] An adjustment only considering overlapping patent nodes that joined between $i-1$ and i might improve results.

tracing was rejected as well. Alternatively, The Simplified Community Detection Measurement is proposed.

4.7 The Simplified Community Detection Measurement

In order to circumvent the limitations of the previously described approach, the objective of the Community Detection Measurement is adjusted. Instead of measuring knowledge components by their approximating patent network communities and the corresponding topic distribution, the components are measured as the most prominent, singular topics in patent communities. Additionally, communities that could be linked between sliding windows are allowed to represent different knowledge components in different points in time.

This pruning and increased flexibility entails a significant loss of information. However, it allows to avoid complicated and error-prone tracing mechanisms. Another major advantage lies in the enhanced comparability of the results between the Community Detection Measurement and the other two measurements.

The adjusted measurement is based on the same CDAs, described at the end of this section. Subsequently, communities of size three are excluded as well. Also, the merging of completely overlapping communities is adopted.

4.7.1 Topic Distribution Pruning

Once these preprocessing steps are applied, the topic distributions of the detected communities are extracted. This is done by aggregating the topic distributions of all patents within a community. The aggregation is illustrated in table 4.1. The first three rows represent a community consisting of three patent nodes. Each column represents a topic. The value in each cell portrays to what extent a patent covers a topic. The fourth row represents the aggregated community distribution, calculated by taking the column (i.e. topics) sum.

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Patent A	0,2	0,4	0,3	0	0
Patent B	0,2	0,1	0	0,2	0,1
Patent C	0,1	0,7	0	0,1	0
Community	0,5	1,2	0,3	0,3	0,1

Table 4.1: Illustration of a community with three patents and the described topic distribution pruning process

After identifying the topic distribution of the community, it is pruned by simply reducing it to the most prominent topic. Accordingly, the most prominent topic of the previous example would be topic 2.

The pruning of the topic distributions is introduced along a measurement of confidence. It indicates how well the most prominent topic represents the whole topic distribution, and thereby how much information is lost during pruning. Let K be the number of topics, then:

$$Confidence = \frac{max(TopicCoverage)}{\sum_j^K TopicCoverage_j}$$

4.7.2 Simplified Recombination and Diffusion

By identifying communities not with static labels, but with their most prominent topics, the construction of the SCMs is analogous to the one of the Direct Measurement. First, all knowledge components (i.e. most prominent topics) present in at least one sliding window are extracted to fix dimension j. Thereafter, every cell SCM_{ij} is filled with the number of times, component j is observed in window i.

The measurement of recombination is adopted from section 4.6.5. Recombinations of two topics are observed, if a node brokers between multiple communities with different most prominent topics; Or, in the overlapping case, if communities with different most prominent topics overlap. In order to populate the CCMs, all recombinations over all sliding windows are gathered to fix dimension j. Subsequently, the cells CCM_{ij} are valued with the number of times recombination j is observed in window i.

This simplified Community Detection Measurement circumvents the problems associated with the limited measurement. However, this adjusted measurement suffers from its own drawbacks. Communities that are mostly, but not completely overlapping are likely to exhibit the same most prominent topic. During the population of the SCM, this characteristic leads to a slight overestimation of diffusion. The only SCM suffering from this overestimation is the one associated with Lais[2]. However, yet unlikely, it is possible that in future other overlapping CDAs might suffer from this problem as well. With the presented patent dataset, K-Clique is not affected. This limitation can be resolved by introducing a more sophisticated merging approach. Exemplary, communities might not only be merged if they are complete subsets, or contain complete subsets of other communities, but also if 80% of their members overlap and if they share the same most prominent topic.

4.7.3 Community Detection Algorithms

This section provides detailed descriptions of the employed CDAs. This descriptions were withheld until now to facilitate a continuous flow of argumentation with regard to the limited and simplified Community Detection Measurement.

The implementation of Label Propagation is accessed via the Python library *NetworkX* (version 2.5). All other algorithms are employed by utilizing implementations of the Python library *CDlib - Community Discovery Library* (version 0.2.1)

Label Propagation

The basic concepts behind this algorithm are labels and propagation. A label is held by at least one node and at most N nodes, with N being the number of nodes in the graph. Nodes themselves always hold exactly one label.

Propagation describes the process by which nodes adapt to the label that the majority of their neighbors exhibit. Once the propagation process terminates, communities are extracted with regard to the label affiliation of nodes [40].

A more detailed description of synchronous Label Propagation is given by:

(i) Initialize all nodes with a unique label. For node x, $C_x(0) = x$.

(ii) Set $t = 1$.

(iii) Order all nodes in a random fashion X.

(iv) Following the order in X, for every node $x \in X$ compute the label of node x via $C_x(t) = f(C_{x_1}(t-1), ..., C_{x_k}(t-1))$. $C_x(t-1)$ represents the label x is holding at time $t-1$. f takes the labels of all neighbors of x at time $t-1$ and returns the neighbor label that is most common. Cases in which label frequencies are tied are decided by randomly picking one of the tying labels.

(v) Check whether all nodes display a label shared by a majority of their neighbors. More formally, let $d_x^{C_j}$ be the number of neighbors that node x has with label C_j. Let C_m be the current label of x. Check if $\forall j : d_x^{C_m} \geq d_x^{C_j}$ holds true for all nodes x (m may be a subset of j). If the condition is met for all nodes, then the algorithm terminates and communities are identified by their label affiliation. Otherwise, t is incremented by 1 and the execution jumps back to step (iii). [40]

In order to prevent the potential oscillation of labels in bipartite- or star-like structures, an extension of the synchronous Label Propagation algorithm was chosen. The asynchronous approach mimics to the synchronous one. The only adaption is the updating function f in step (iv). With
$C_x(t) = f(C_{x_{i1}}(t), ..., C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), ..., C_{x_{ik}}(t-1))$ the label calculation still takes the labels of all neighbors $x_1, ..., x_k$ into account, but if some neighbors $x_{i1}, ..., x_{im}$ were already updated in the current t, then their current labels $C_{x_{i1}}(t), ..., C_{x_{im}}(t)$ are used instead of their labels in $t-1$ [40].

The chosen NetworkX implementation takes edge weights into account by adjusting the updating function f further. In the unadjusted version, the simple counting of neighbor labels $C_x(t)$ or $C_x(t-1)$ corresponds to weighting them with one. This value is adjusted with edge weights. Thereby, neighboring labels connected with a high edge weight are more likely to be chosen as label for x, than neighboring labels connected with a low edge weight. [40]

By applying this community detection, the network is partitioned into disjointed communities, in which every node has at least as many neighbors within its community as it has to any other community [40].

Greedy Modularity

Representing Modularity based community detection algorithms, [10] provides another approach of identifying crisp communities. Before diving into the algorithm itself, a brief description of Modularity is provided. In contrast to local network measurement like centralities, Modularity is a measurement on graph level.

Let:

- m be the number of edges in the graph
- A_{vw} be the element of a corresponding adjacency matrix
- $\delta(i, j)$ be a function returning 1 if $i = j$ and 0 otherwise
- C_v be the community, that node v is a member of

With this notion Modularity [10] is defined as:

$$\frac{1}{2m} \sum_{vw} A_{vw} \delta(C_v, C_w)$$

Unadjusted, it represents the ratio of edges falling within communities to all edges in the graph. High modularity values, thereby indicate a good network partition in the sense of having many edges within communities, and few edges between them. However, this notion of modularity might be misleading for partitioning algorithms. Cases in which all nodes of a graph belong to one giant community are rewarded with a perfect score of 1 in this unadjusted case. This is addressed by considering how edges would fall in a randomized network [10].

Let:

- k_v be the degree of node v
- $\frac{k_v k_w}{2m}$ be the probability of an edge existing between two nodes v and w, if connections are made at random, but respecting node degrees

By subtracting this probability $\frac{k_v k_w}{2m}$ from the fraction of within-community edges $\sum_{vw} A_{vw}\delta(C_v, C_w)$, we get an adjusted Modularity Q, in which nonzero values represent deviations from randomness [10].

$$Q = \frac{1}{2m}\sum_{vw}\left[A_{vw} - \frac{k_v k_w}{2m}\right]\delta(c_v, c_w)$$

Positive values represent favorable network partitions. In order to illustrate how the CDA utilizes Modularity, Q is rewritten as:

$$Q = \sum_i (e_{ii} - a_i^2)$$

With:

$$e_{ij} = \frac{1}{2m}\sum_{vw} A_{vw}\delta(c_v, i)\delta(c_v, j)$$

$$a_i = \frac{1}{2m}\sum_{v} k_v\delta(c_v, i)$$

Here, e_{ij} is defined as the fraction of edges that join nodes of community i and nodes of community j. a_i is defined as the fraction of edge ends linking to nodes in community i. A more detailed breakdown can be found in [10].

The CDA utilizes the modified Modularity by calculating Q for all possible partitions. Subsequently, the partition maximizing Q is chosen. However, computing the Q for all possible partitions within big networks is unfeasible. Tackling this limitation, [10] introduce a greedy approximation:

(i) Initialize all nodes with their own unique community.

(ii) Compute the change in Modularity for every community pair, when joining the pair.

$$\Delta Q = e_{ij} + e_{ji} - 2a_i a_j = 2(e_{ij} - a_i a_j)$$

Joining communities with no edges between them will not result in an increase of Q and are omitted.

(iii) Join the pair that increases Q most, or decreases Q the least.

(iv) Update e_{ij}. If more than two communities are left after the merge, then jump back to (ii).

(v) Chose the partition with highest Q.

This results in a dendrogram in which the leaves represent the nodes and the root represents a giant community including all nodes. Cutting this dendrogram at different levels results in partitions with varying number of communities and community sizes.

The variation of the algorithm implemented in CDlib is further optimized and introduces more sophisticated data structures in order to shrink the complexity from $O((m+n)n)$ (or $O(n^2)$ on a sparse graph) to $O(md \log n)$, where d represents the depth of the dendogram. Both versions result in an identical community detection. For a more detailed description, see [10].

K-clique

The K-Clique CDA concerns itself with cliques and K-cliques. Both are defined as maximally connected subgraphs. The former refers exclusively to cliques that are not part of bigger cliques. The latter to cliques that may be subsets of bigger cliques. Additionally, the latter are specified by a size of K nodes.
This CDA considers a community to be a union of all K-cliques, in which every node can reach every other node through a series of adjacent K-cliques. K-cliques are considered adjacent, if they share at least K-1 nodes [38].

The algorithm consists of four parts. First cliques are identified. Afterwards a Clique-Clique Overlap Matrix is computed. Next, this matrix is modified and used for a concluding component analysis [38]. In the following paragraphs, each part is described in more detail.

Identifying Cliques:

The first step of clique identification relies on the degree sequence of a graph. This sequence represents the descending ordered list of node degrees present in a graph. Illustrating this, the degree sequence of figure 4.5 is:

$$8, 7, 5, 5, 5, 5, 5, 3, 3, 2$$

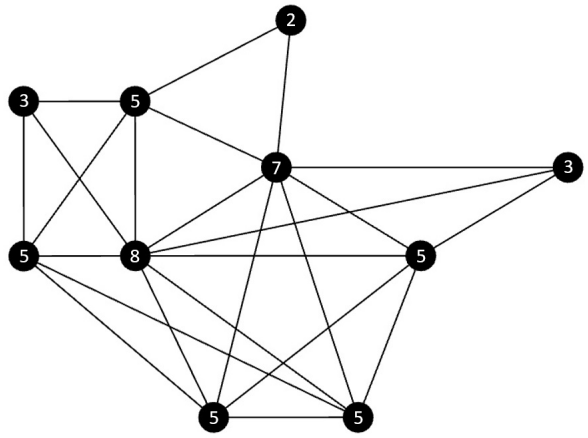

Figure 4.5: Illustration of the concept of degree sequence with an exemplary graph of ten nodes. The numbers inside the nodes represent their degree centrality.

Utilizing this sequence, the size s of the biggest possible clique within a graph is extracted. Nodes are then iteratively chosen and checked for their membership in cliques of size s. Once all cliques of that size s that include the iterated node v are collected, v and its edges are deleted and the next node is iterated through. This deletion prevents multiple collections of the same clique. When no further cliques can be collected, the size s of the searched cliques is decreased by one and the original graph is restored. The cliques found in previous rounds are used to limit the search space of the upcoming rounds, since a clique must not be part of larger cliques [38].

In order to find the cliques of size s for which v is a member of, the algorithm explores v's neighbors. For this purpose two disjunct sets are constructed. Set A is defined by containing only nodes that are maximally connected to each other. Set B, by containing only nodes that are connected to every node in A. The nodes in B don't have to be maximally connected within themselves. In the beginning, set A contains only v and set B contains all neighbors of v. Recusively, a node w is transferred from B to A. In order to conserve the set properties, all nodes that are not neighbors of w are deleted from B. If A reaches size s, a new clique is found and the algorithm resumes to explore the remaining neighbors for further cliques. If B runs out of nodes, or if both sets are subsets of an already found bigger clique, the recursion is stepped back to explore other possibilities. The transfer of nodes from B to A is ordered by their node index, to prevent the finding of the same clique multiple times [38].

Computing Clique-Clique Overlap Matrix:

The overlap matrix A in figure 4.6 represents every previously found clique as row i and column j index. Every cell A_{ij} contains the number of nodes both cliques are sharing. This means the diagonal values represent the size of the respective cliques. Following this rationale, the size of the red clique is identified $A_{red,red} = 4$. The off-diagonal values represent fully connected subgraphs as well. The size of the intersection of the blue and yellow clique is identified in $A_{yellow,blue} = A_{blue,yellow} = 3$ [38].

Matrix Manipulation:

The resulting overlap matrix is adjusted in the following way. Every diagonal value smaller than k, and every off-diagonal value smaller than $k - 1$ is replaced with a zero. The remaining non-zero values are set to 1 [38].

Component Analysis:

At last, the K-clique communities are extracted via a component analysis. The example provided by the authors reveals two $k = 4$ -clique communities. For this illustration, the removal of all rows and columns with only zeros (green and pink) might ease the intuitive detection of the components (i.e. communities) [38].

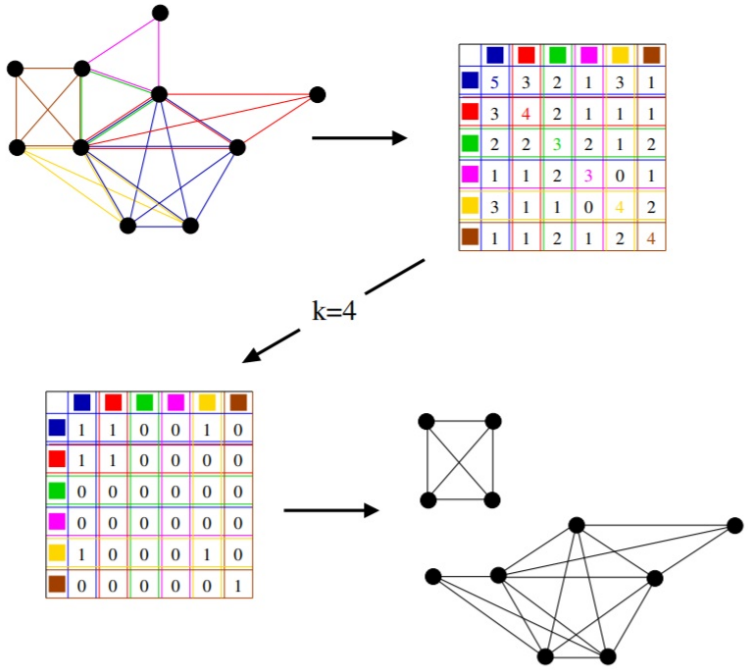

Figure 4.6: Illustration of the Clique-Clique Overlap Matrix, its modification and the component analysis. Each colour represents a clique. (source: [38])

Lais^2

Lais^2 consists of two algorithms. At first the Link Aggregate algorithm (LA) initializes seed clusters. Subsequently, these clusters are then updated during the Improved Iterative Scan algorithm (IS^2). In the latter, nodes are added and deleted from clusters until the density of the clusters stops improving. The resulting node clusters are the reported communities [4].

Link Aggregation:
During the link aggregation phase, community cluster seed are distributed with regard to Page Rank centrality and density [4]. In the following paragraphs the LA algorithm is formalized.

Let:

- G be the graph LA is operating on
- V be the vertices of G
- E be the edges of G
- C be the communities of G
- D_j be community $j \in C$
- W be a density function

Algorithm 1 Lais^2 - Link Aggregate Algorithm

Link Aggregate ($G = (V, E), W$)

```
1: C ← ∅;
2: Order the vertices v_1, v_2, ..., v_|V|;
3: for i = 1 to |V| do
4:     added ← false;
5:     for all D_j ∈ C do
6:         if W(D_j ∪ v_i) > W(D_J) then
7:             D_j ← D_j ∪ v_i; added ← true;
8:     if added = false then
9:         C ← C ∪ {{v_i}};
10: return C;
```

After initializing C as empty, all vertices V are ordered descending by their Page Rank centrality. Following this order, for all sorted nodes v_i it is checked whether adding v_i to community D_j increases the density of D_j. If this is the

case, v_i is added to D_j. If node v_i does not improve the density of any community $D_j \in C$, then v_i establishes its own community. After termination every node is part of at least one cluster [4].

Improved Iterative Scanning:
Note that LA is applied once on graph level and that C in LA refers to the set of all communities in G. Contrary, IS^2 is applied on cluster level, hence C now refers to one community fed to IS^2.

Let:

- C be the community IS^2 is operating on
- N be community C extended by all adjacent nodes

Algorithm 2 Lais^2 - Improved Iterative Scan Algorithm

Iterative Scan (seed, G, W)

```
1:  C ← seed; w ← W(C);
2:  increased ← true;
3:  while increased do
4:      N ← C;
5:      for all v ∈ C do
6:          N ← N ∪ adj(v);
7:      for all v ∈ N do
8:          if v ∈ C then
9:              C' ← C \ {v};
10:         else
11:             C' ← C ∪ {v};
12:         if W(C') > W(C) then
13:             C ← C';
14:     if W(C) = w then
15:         increased ← false;
16:     else
17:         w = W(C);
18: return C;
```

For every iterative scan denoted by the while-loop, the extended community N comprises all nodes in C and their neighbors. Subsequently, for every node v in N it is checked, whether adding or deleting this node from the cluster would improve its density. If this is the case, the node is added or deleted respectively. This

iteration is repeated until no further improvement in the density can be made. After refining all cluster in this manner, the detected communities process terminates [4].

4.8 The Edge Weight Measurement

The Edge Weight Measurement utilized the other network projection of the bipartite patent-topic network. The nodes in this projected topic network represent the topics in the sliding window. The edges connecting topics are considered to represent the patent similarity between them. Patent similarity can be thought of as an abstract measurement for topic co-occurrence.

Knowledge components are defined as node activity. Combined knowledge components are defined as edge activity. Topic node activity is measured by how often a node is involved in an edge. The activity of patent similarity edges is measured by their weight, resulting from the projection function in section 4.3.2.

In contrast to the Community Detection Measurement, no further methods are applied on the topic network before constructing the SCM and CCM. The dimensions of the former are established similarly to previous measurement. i is set by the number of sliding windows, and j by the number of topic nodes represented over all windows. Once again, this number is expected to equal the respective hyperparameter given to the LDA. The cells SCM_{ij} are filled with the number of times a node j is engaged in an edge in window i.

The j dimension of the CCM is fixed by the number of unique edges observed over all windows. The uniqueness of an edge is defined by the node pair it is connecting (the weight is neglected). The cells CCM_{ij} are filled with the weight of edge j in window i. If no edge is present, the value of CCM_{ij} is defined as 0.

4.9 Comparative Analysis

The most suitable evaluation for not only the results of the LDA, but also for the performance of the three proposed measurements involves experts in patent administration and other related fields. Incorporating this expertise allows for judgments about how well recombining patents, and recombination and diffusion periods are identified. While such resources might be employed in future efforts, this thesis has to compromise and evaluate the fitness of the proposed approaches comparative to each other.

SCM	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Window i	0	0	6	1	0
Window i+1	1	0	5	2	0
Window i	0	0	0,86	0,14	0
Window i+1	0,13	0	0,63	0,25	0
Window i	0	0	1	0	0
Window i+1	0	0	1	1	0

Table 4.2: Illustration of the normalization and binarization of SCMs. The same process is applied on CCMs as well. The exemplary numbers are rounded after the second decimal.

For this comparative analysis, the identification of singular knowledge component diffusion in SCMs, and the identification of recombinations of knowledge components and their diffusion in CCMs are compared respectively.

4.9.1 Comparing SCMs

Before comparing the measurements in the former sense, the SCMs of the three approaches are aligned. For this purpose, the entries SCM_{ij} are normalized with respect to their sliding window i. Afterwards the SCMs are binarized with respect to a threshold value. All cells exceeding this threshold are converted to 1, all other to 0. The values intuitively indicate whether a knowledge component j is considered diffusing in window i, or not.

These matrix modifications are illustrated with an exemplary SCM of five components (i.e. topics) j and two sliding windows i, presented in the first two rows of table 4.2. Rows three and four exhibit the window normalized values for the same SCM and rows five and six the binarized values for the same SCM.

In a last alignment step, small leeway is introduced into the diffusion patterns of the SCMs, by bridging disconnected diffusion cycles within a component. Let 011101110 be an exemplary diffusion pattern in column of a SCM with size of $i = 9$. By introducing leeway of size 1, the sequence is transformed to 011111110. The introduction of this leeway is not necessary, but tries to account for irregularities in the data. If deemed unfitting, this leeway can be excluded with no effort.

After retrieving key statistics regarding the average number and length of diffusion patterns in the SCMs, four similarities are computed.

1. The SCM similarity between every pair of measurements
2. The SCM similarity between all measurements
3. The similarity in measuring the diffusion of every knowledge components j between all measurements
4. The similarity in measuring the diffusion of all knowledge components between all measurements

For this purpose, the concepts of Cosine and Manhattan similarity are utilized in a modified way.

The common notion of Cosine similarity reaches its limit, if one of the vectors compared, consists of zeros only. This case would require to divide by zero. If translated to the diffusion patterns of knowledge components in the SCMs, then cases in which a component j is measured to have no diffusion at all ($\sum SCM_j = 0$), would also require dividing by zero. This limitation is addressed by modifying the cosine similarity to equal 0 in such cases.

$$CosineSimilarity_{mod} = \begin{cases} CosineSimilarity & \text{if one pattern is empty} \\ 0 & \text{otherwise} \end{cases}$$

In the binary case, the Manhattan Similarity returns the number of equal positions in two vectors of the same length. In order to arrive at a value range between 0 and 1, with 0 indicating no similarity and 1 indicating complete similarity, the result is normalized (i.e. divided) by the length of the vectors.

The SCM similarity between every pair of measurements

The similarity between every SCM pairs is retrieved in three steps. First, all six SCMs are paired against each other, resulting in 15 pairs. Next, for every SCM pair, the similarity between all K column pairs is calculated using the modified Cosine and Manhattan similarity. At last, these K similarity scores are averaged within every SCM pair, resulting in 15 similarities. With $K =$ column dimension of the SCMs.

The SCM similarity between all measurements

The overall similarity between all SCM pairs equals the average of these 15 pair similarities.

The similarity in measuring the diffusion of every knowledge components j between all measurements

For every knowledge component j, the similarity between all 15 SCM_j pairs is calculated using the modified Cosine and Manhattan similarity. Subsequently, these 15 similarity scores are averaged within every knowledge component j, resulting in K similarities.

The similarity in measuring the diffusion of all knowledge components between all measurements

The overall diffusion similarity between all measures equals the average of these K component similarities.

4.9.2 Comparing CCMs

The comparison of the CCMs requires preprocessing as well. The first issue tackled is the difference in the diffusion measuring between the Community Detection Measurements, and the other measurements. In the Direct and Edge Weight Measurement, a singular recombination spreads through all sliding windows containing the date the recombination was measured. On the contrary, the crisp and overlapping Community Measurements account for a recombination just once. When a patent node emerges between the sliding windows $i-1$ and i, and happens to recombine knowledge components, then it is registered in i only, and not in i and all other sliding windows overlapping with i. In order to align this discrepancy, the recombination entries in the respective CCMs are extended. Consider an exemplary case in which the window size and the sliding interval equals 12 months and 1 month, and in which $i-1$ equals the time span January to December 2005 and i equals February 2005 to January 2006. When a recombination emerges in i, the recombination date necessarily falls on January 2006. In the original Community Detection Measurement CCMs, $i+1$ to $i+k$ would not represent this recombination, even though including January 2006 ($k =$ window size $- 1$). Therefore, every entry CCM_{ij} is extended by adding the value of CCM_{ij} onto the following cells $CCM_{i:i+k,j}$.
After this diffusion alignment, all CCMs are normalized and binarized in the same manner as the SCMs. Additionally, leeway is introduced in the same manner as well.
At last, the column span of the CCMs is aligned. Every CCM only accounts for knowledge component combinations which they measure at least once. This means, that not only the size of the j dimension varies, but also their order. An

exemplary diffusion pattern $j = 15$ in one CCM is unlikely to represent the same component combination as a diffusion pattern $j = 15$ in a different CCM.
This last column alignment is done by extracting all component combinations over all measurements that occurred at least once. Afterwards, this extracted list of recombinations is sorted in an ascending order, similarly to the recombinations extracted during the construction of the original CCMs. Next, for every CCM the missing columns are identified and inserted. If the diffusion pattern for a component recombination has to be inserted after the construction, then the respective recombination was not measured once within the respective Measurement. For this reason, the inserted columns represent no diffusion patterns. All cells of these columns equal zero.

Once, the CCMs are aligned. The same descriptive statistics and similarity scores as for the SCMs are computed.

Additionally, a method finding customized patterns in the aligned and unaligned SCMs and CCMs is provided in the attached code.

4.10 Predicting Recombination

After presenting several approaches for measuring past recombination of knowledge components, a short introduction into the prediction of future recombination is given.

When revisiting the Edge Weight Measurement in section 4.8, the prediction of recombination can be defined as a problem of link prediction within the topic network projections.

The approach presented in this section is highly limited and merely supposed to serve as a proof of concept. Link prediction tasks on their own are already complex ventures, when applied on clearly defined concepts, established measurements and far bigger data samples.

In order to predict recombining edges in the topic network, two topic network representations of sliding windows are chosen. Subsequently, the older one is used to train Node2Vec [22] node embeddings for all topic nodes in the network. These node embeddings are then utilized to compute the edge embeddings of the edges to be predicted in the younger sliding window. Both network representations only contain true, actually existing edges. When predicting the edge weights via these edge embeddings, then the existence of recombination is not predicted, but presupposed. The only thing predicted would be the strength of the presupposed recom-

bination. Tackling this, the younger network representation is appended with fake edges of weight zero.

When predicting edge weights in the described manner, it is deemed appropriate to only consider recombination predicted, if the predicted edge weight exceeds a set threshold.

4.10.1 Node2Vec

Node2Vec is an embedding framework that extracts continuous feature representations of nodes. Inspired by word embedding algorithms found in NLP, Node2Vec utilizes the context in which nodes are found to infer similarities between them. In this framework, contexts are thought to be the neighborhoods of nodes. Contrary to most network related ideas, Node2Vec uses the term neighborhood not to refer to nodes adjacent to one focal node, but rather to a collection of nodes that were sampled with biased random walks starting in one focal node. The overall feature learning process is treated as a maximum likelihood optimization problem, incorporating negative sampling [22].

In order to extract meaningful context of nodes, a flexible sampling strategy is introduced that combines aspects of Breadth-first and Depth-first Sampling. The former refers to a sampling of directly adjacent neighbors of a focal node. The later to a sampling of walks (i.e. node sequences) starting from the focal node, and increasing in distance to the focal node with every step. Breath-first is suited to capture structural equivalencies between nodes, meaning capturing similar roles nodes take on in a network. Exemplary, two hub-nodes with many neighbors might be located in different parts of the network, yet still may fulfil similar functions in the network.

On the contrary, Depth-first is geared to explore larger parts of the graph, hence a more macro-view of the neighborhood. This macro-view facilitates inference about neighborhood communities based on homophily. For a more detailed description, see [22].

Node2Vec accounts for both concepts with a biased random walk sampling including a Return parameter p and an In-out parameter q. Let:

- c_0 be the source node of a random walk with fixed length
- c_i be the ith node in the walk
- π_{vx} be the transition probability between nodes v and x
- Z be a normalization constant [22]

With this notion, the generation of an upcoming node c_i in a random walk is denoted as:

$$Pr(c_i = x | c_{i-1} = v) = \begin{cases} \frac{\pi_{vx}}{Z} & \text{if } (v, x) \in E \\ 0 & \text{otherwise} \end{cases}$$

The random walk is biased to stay local to the focal node, or to explore nodes more distanced, by adjusting the transition probability π_{vx} [22]. Let:

- v be the node, the random walk is currently residing on
- t be the node, the random walk visited before traversing to v
- d_{tx} be the shortest path distance between nodes t and x
- w_{vx} be the edge weight between node v and x (with $w_{vx} = 1$ in an unweighted graph)

Then the transition probability for a biased random walk in Node2Vec is $\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx}$ [22], where:

$$\alpha_{p,q}(t, x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases}$$

With this bias, high values in the Return parameter p (i.e. $p > max(q, 1)$) encourage a moderate explorative random walk, while low values (i.e. $p < min(q, 1)$) lead to frequent backtracks, keeping the walk local, near the starting node of the walk.

High values in the In-out Parameter q (i.e. $q > 1$) approximate Breadth-first Sampling by biasing the random walk towards nodes close to t, thus creating a local view of the source node. Lower values of q (i.e. $q > 1$) emulate Depth-first Sampling by inclining the walk to favor nodes further away from t.

With this definition of node neighborhoods (i.e. contexts), the maximum likelihood optimization is addressed. Let:

- $G = (V, E)$ be a given graph
- $f : V \rightarrow \mathbb{R}^d$ be the mapping function from nodes to feature representations
- d be the number of dimensions of the feature representations
- S be a sampling strategy for network neighborhoods
- $N_S(u) \subset V$ be a network neighborhood of node u generated through S
- $n_i \in N_S(u)$ be a node in the sampled neighborhood of node u

Building on top of the Skip-gram concept [23], Word2Vec aims to optimize the following function, maximizing the log-probability of observing a network neighborhood $N_S(u)$ for node u given the feature representation $f(u)$ for node u.

$$\max_f \sum_{u \in V} \log Pr(N_S(u)|f(u))$$

The assumption that the likelihood of observing a neighborhood node $n_i \in N_S(u)$ is independent of observing any other neighborhood node given the feature representation $f(u)$ (i.e. assuming Conditional Independence), allows for:

$$Pr(N_S(u)|f(u)) = \prod_{n_i \in N_S(u)} Pr(n_i|f(u))$$

By additionally assuming symmetry in feature space, the conditional likelihood of every source-neighborhood node pair can be modeled as a softmax unit parametrized by a dot product of its features:

$$Pr(n_i|f(u)) = \frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in V} \exp(f(v) \cdot f(u))}$$

These assumptions restate the original function maximizing log-probability to:

$$\max_f \sum_{u \in V} \left[-\log Z_u + \sum_{n_i \in N_S(u)} f(n_i) \cdot f(u) \right]$$

With:

$$Z_u = \sum_{u \in V} \exp(f(u) \cdot f(v))$$

Z_u represents a per-node partition function. For large networks, Z_u becomes expensive to compute, since the embedding of the source node $f(u)$ has to be multiplied with the embedding of every other node $f(v)$. Addressing this complexity, Word2Vec relies on an approximation via negative sampling. Also originating in NLP, negative sampling simplifies the calculation of Z_u by reducing the number of embeddings, that $f(u)$ has to be multiplied with a hyperparameter given to the algorithm. A more detailed description can be found in [19]. The optimization is tackled by employing stochastic gradient descent over the model parameters defining the features f [22].

The Node2Vec algorithm consists of three phases. First the transition probabilities are computed in row 1, then the biased random walks are simulated in rows 6. At last Stochastic Gradient Descent is employed to optimize the feature representation in row 8.

Algorithm 3 Node2Vec

LearnFeatures(Graph $G = (V, E, W)$, Dimensions d, Walks per node r, Walk length l, Context size k, Return p, In-out q)

1: $\pi =$ PreprocessingModifiedWeights(G, p, q)
2: $G' = (V, E, \pi)$
3: Initialize $walks$ to Empty
4: **for** $iter = 1$ to r **do**
5: **for all** nodes $u \in V$ **do**
6: $walk =$ node2vecWalk(G', u, l)
7: append $walk$ to $walks$
8: $f =$ StochasticGradientDescent($k, d, walks$)
9: **return** f

The authors provide an approach of deriving edge embeddings from the computed node embeddings. Given nodes u and v and their respective embeddings $f(u)$ and $f(v)$. Then an edge embedding $g(u, v)$ can be generated by applying a binary operator on both node embeddings. The binary operator chosen for this thesis is the average of both feature representations.

Algorithm 4 Node2Vec - node2vecWalk

node2vecWalk
(Graph $G' = (V, E, \pi)$, Start node u, Length l)

initialize $walk$ to $[u]$
for $walk\_iter = 1$ to l **do**
    $curr = walk[-1]$
    V_{curr} = GetNeighbors $(curr, G')$
    $s = AliasSample(V_{curr}, \pi)$
    Append s to $walk$
return $walk$

Chapter 5

Results

In this chapter, the intermediate and final results of the presented methodology are documented. The structure of the Results chapter follows the structure of the Methodology chapter.

5.1 Patent Abstract Preprocessing

Before applying any preprocessing, the 3.781 patent abstracts exhibit a vocabulary of 14.324 terms. After removing all non-alphabetic characters and single letter terms, and after lowercasing all terms, the vocabulary shrinks to 9.936 terms. After this first preprocessing step, 436.944 terms are distributed over all abstracts, leading to a average abstract size of 115,56 terms. A more detailed breakdown of these statistics can be found in table 5.1.

In order to evaluate the usefulness of the smallest abstracts, all abstracts with a term count of 20 or lower were reviewed manually. Following a conservative approach, all five identified patents are kept for the following analysis. With more domain knowledge, future efforts might handle these outliers differently.

In a next step, the abstracts are tokenized and stop words are filtered out, to facilitate the forming of only relevant bigrams. The attached code provides several filters with varying levels of exclusion confidence. Here as well, more expertise might suggest a different filtering. The filter used for the following analysis involves most of the default stopwords provided by the python *NLTK* library (version 3.5), English and Roman numbers and a custom selection of stop words like 'therefor' and 'additionally'.

A detailed list of filtered stopwords can be found in the Appendix A or in the attached code. The applied filter contains 304 stopwords. After the stopword filtering, the vocabulary shrinks to 9.714 words.

After removing non-alphabetics and single letters and lowercasing

	Terms in Abstracts	Unique Terms in Abstracts
Mean	115,56	55,08
Median	113	53
Mode	99	50
Max	492	147
Min	8	8
Total	436.944	9.936

After complete Preprocessing

	Terms in Abstracts	Unique Terms in Abstracts
Mean	56,02	30,01
Median	54	29
Mode	55	26
Max	233	101
Min	4	4
Total	211.799	7.291

Table 5.1: Number of terms and unique terms per abstract after the first, and all preprocessing steps.

Fitting bigrams are formed using the 'Phrases' function of the *Gensim* Python library (version 3.8.3). Adjusting the 'min_count' and 'threshold' hyperparameters to 10 and 100, while keeping the default values of the remaining hyperparameters results in 76 bigrams. Examplary formed bigrams are 'autonomous_coverage' and 'closed_loop'. A more detailed list can be found in the Appendix A or in the attached code.

The following lemmatization of the vocabulary is preformed using the *en_core_web_sm* pipeline, provided by the python library *Spacy* (version 3.0.6). This step shrinks the vocabulary further to 7.348 tokens.

At last, the described stopword filter is applied once again, to erase irrelevant variations of stopwords previously missed. This removes another 57 tokens from the vocabulary.

The resulting preprocessed vocabulary consists of 7.291 tokens. Over all abstracts, 211.799 tokens remain, leading to an average abstract size of 56,02. A more detailed breakdown can be found in table 5.1 as well.

5.2 Probabilistic Topic Modeling

Some of the most popular Latent Dirichlet Allocation (LDA) algorithms used for topic modeling are *Gensim* [41] and *Mallet* [34]. The major difference between both algorithms lies in their estimation of the latent variables introduced in section 4.2.1. *Gensim* utilizes Variational Bayes (see [24]) for its estimation, while *Mallet* makes use of a Gibbs Sampling approach (see [52]).

In the following subsection, both are tested for their suitability in identifying knowledge components. The *Gensim* LDA is accessed via the Python library of the same name (version 3.8.3). The *Mallet* LDA is implemented in Java and accessed via a wrapper function of the *Gensim* Python library.

5.2.1 Hyperparameter Estimation

In order to estimate the desired, hidden, topic related variables, fitting hyperparameters need to be specified beforehand. However, identifying optimal LDA hyperparameters is coupled with major challenges. With a lack of performance measures like accuracy or precision, the unsupervised nature of topic modelling used to require the manual evaluation for every hyperparameter setting each. To overcome this, researchers introduced several coherency scores. The one utilized here is C_V, described in section 4.2.2.

In order to shrink the search space for suitable hyperparameter settings, patent topic modeling literature presented in chapter 2 is briefly revisited. [26], [25] and [14] report to achieve desirable patent topics with a β value of 0.01. Following their approach, this value is chosen as a first point of orientation.

With regard to further hyperparameter settings, the presented literature depicts a more diverse picture. [26] works with 2.826 patents and 100 topics, leading to a patent-topic ratio of 28,26. The ratios of [25] and [14] fall on 6,82 and 37,8. Guided by this, the search space for a suitable number of topics is orientated around

a lower bound of 100 ($\frac{3.781}{37,8}$) and an upper bound of 550 ($\frac{3.781}{6,82}$).

Additionally, the α values of the described works are reported as 0.1, 0.5 and 5. Due to α influencing the assumed number of topics used to compose an abstract, smaller values appear more suitable for the proposed framework. Low values guide the modelling to identify fewer, more dominant topics in an abstract, while higher values tend to identify more, but less dominant topics. In order to measure meaningful and clear diffusion and recombination patterns, concise knowledge components seem favorable. Following this rationale, the search space for α values was confined to a lower bound of 0.1 and a higher bound of 0.5. As last hyperparameter of interest for this proposed explorative framework, suitable values for the 'optimization_interval' were investigated. This parameter is supported by *Mallet* and fits an internal hyperparameter optimization, allowing topics within a model to be more prominent than others [33]. None of the presented studies states the use of this hyperparameter, so its range is explored more freely.

Instead of feeding the derived value ranges into a grid search, their separated, independent influence on the coherency score and on the topic quality (judged by the author) is investigated beforehand. For this purpose, multiple LDA models are computed, with only one hyperparameter varying.
First, the effect of the number of topics within *Gensim* is analyzed. Figure 5.1 presents the coherency score C_V for all models with a value interval of 50 to 550, and a step size of 50. β is fixed to 0.01 and α to 0.1. A small upward trend with an increase in the number of topics is observable.
This trend is exacerbated in Mallet, displayed in figure 5.2. For Mallet, additionally, the 'optimization_interval' parameter was fixed at the default value of 0. The substantially bigger coherency scores in the *Mallet* models are the first signs for their better suitability, compared to the *Gensim* models.
The trend of C_V increasing with the number of topics can be found in the nature of C_V. The more topics a model is allowed to use when clustering words, the easier it is for the model to only group similar, co-occurring words together into one topic. Hence, the easier coherent topics tend to form.

With this in mind, simply choosing the number of topics with the highest C_V seems ill-advised. Alternatively, models with topic number values of local maxima (for figure 5.1) and topic number values for which the slope begins to decrease (for figure 5.2) are investigated more thoroughly. The coherency scales of both figures are not aligned in order to not disguise the patterns in figure 5.1. The Appendix B provides a visualization with an aligned coherency scale and a broader topics number range. In it, the continuation of this trend is observable.

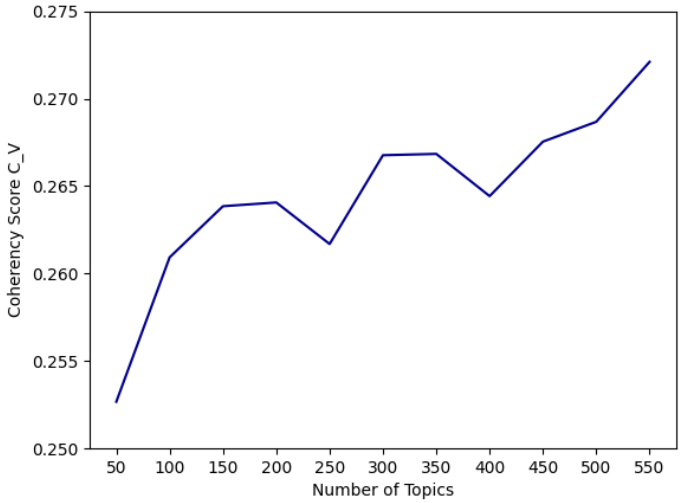

Figure 5.1: Coherency scores of *Gensim* models with varying number of topics ($\beta = 0,01$, $\alpha = 0,1$, rest = default)

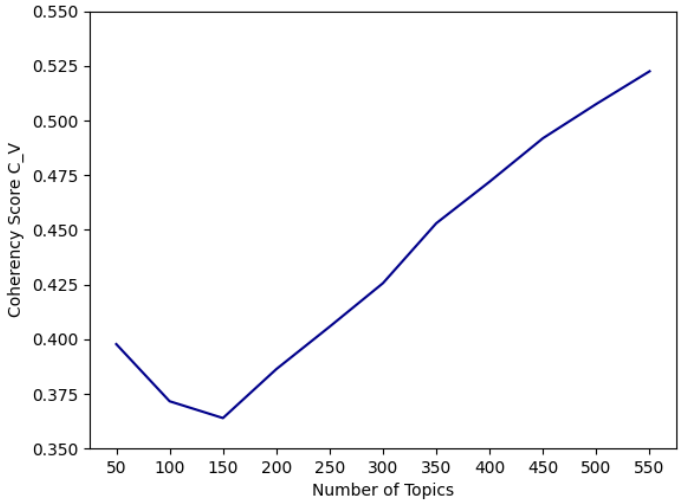

Figure 5.2: Coherency scores of *Mallet* models with varying number of topics. ($\beta = 0,01$, $\alpha = 0,1$, optimization interval = 0, rest = default)

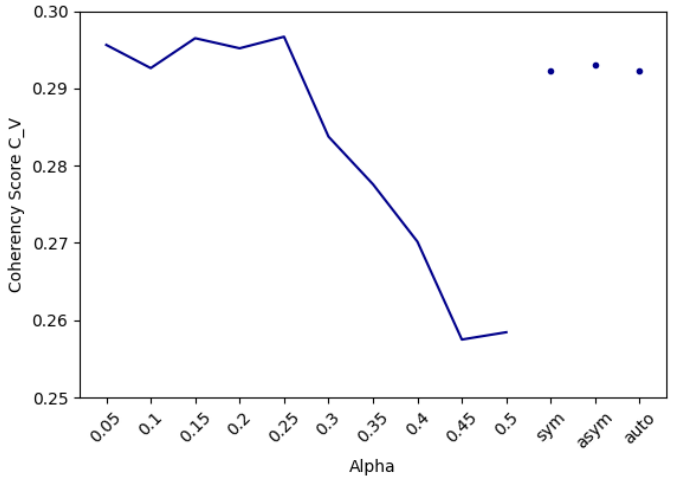

Figure 5.3: Coherency scores of *Gensim* models with varying α values. ($\beta = 0,01$, number of topics $= 330$, rest $=$ default)

By investigating the resulting topics of these models, and by investigating the topics of models within an additional, more fine-grained search, a number of 330 topics appears most appropriate.

Favorable α values were searched for in a similar manner. Figure 5.3 and 5.4 present the coherency scores of *Gensim* and *Mallet* models with 330 topics and the same fixed non-α hyperparameters.

While once again, C_V tends to increase with bigger α values in the *Mallet* models, the *Gensim* models paint the opposite picture. With the same heuristic applied in the search for a suitable number of topics, the resulting topics corresponding to models with α values at local maxima and slope decreases were revisited. An α-value of 0.15 appears suitable.

At last, fitting values for the *Mallet* optimization interval are searched. Figure 5.5 shows that the default value of 0 and values of 1000 and above preform best, and equally well. Values between 0 and 1000 consistently underpreform. With regard to the resulting topic quality, the value of 1000 is deemed appropriate.

After identifying these hyperparameters, broader grid searches were preformed to make sure, that no unexpected patters were hidden within unaccounted hyperparameter interactions. In over 5.000 computed LDAs no such patterns were observed. Subsequently, the topic quality of randomly sampled models and hyperpa-

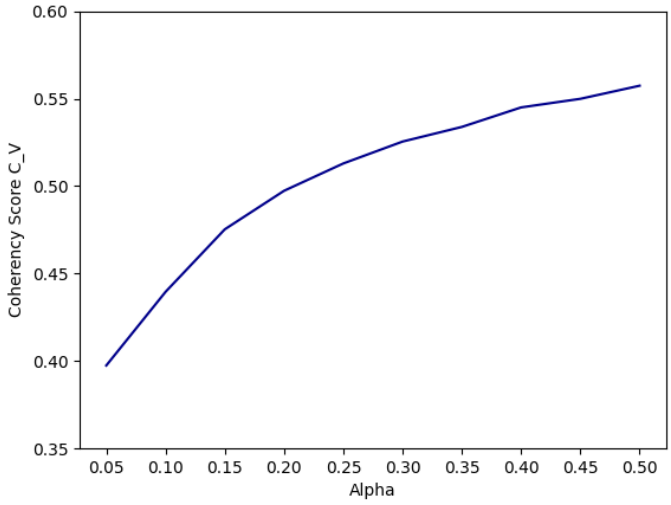

Figure 5.4: Coherency scores of *Mallet* models with varying α values. ($\beta = 0,01$, number of topics $= 330$, optimization interval $= 0$, rest $=$ default)

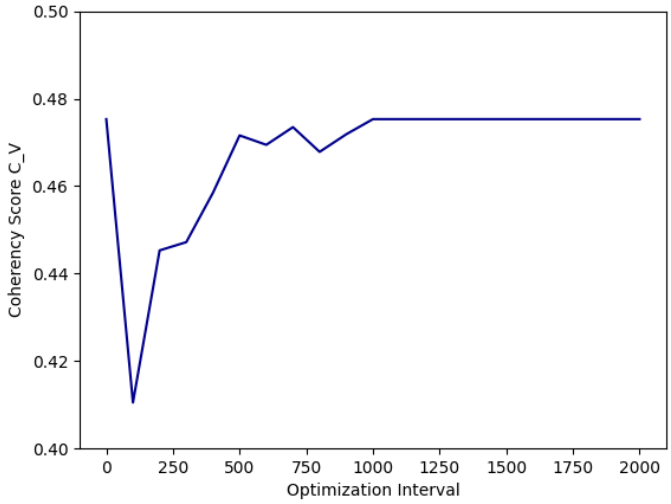

Figure 5.5: Coherency scores of *Mallet* models with varying optimization intervals. ($\beta = 0,01$, $\alpha = 0,15$, number of topics $= 330$, rest $=$ default)

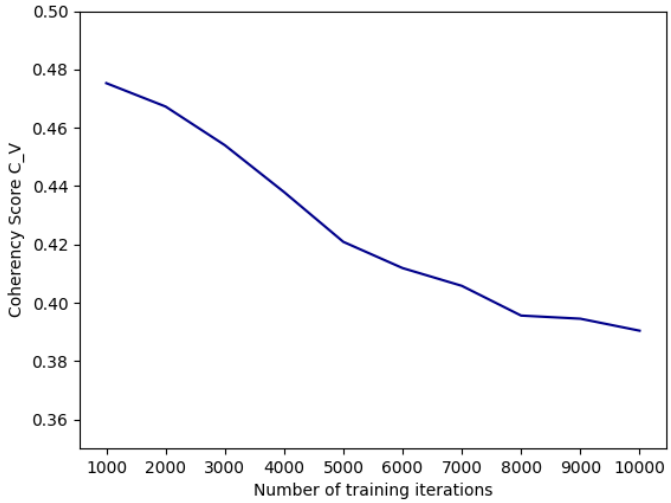

Figure 5.6: Coherency scores of *Mallet* models with varying number of training iterations. ($\beta = 0,01$, $\alpha = 0,15$, number of topics $= 330$, optimization interval $= 1000$, rest = default)

rameter settings were evaluated. Over all, the quality of the resulting *Mallet* topics appeared more useful and clearer that the *Gensim* ones.

A *Mallet* model resulting from the settings $\beta = 0,01$, $\alpha = 0,15$, number of topics $= 330$, and optimization interval $= 1.000$ was concluded to be most fitting for further analyses. In an attempt to boost topic quality and coherency further, the number of training interactions was raised from the default value of 1.000 to 10.000 with a step size of 1.000. Unfortunately, topic quality did not improve, neither did coherency (see figure 5.6)

5.2.2 Topic Descriptives

Most abstracts contain around three to five topics, with an average number of 2,91 topics. Table 5.2 and figure 5.7 provide a detailed view of the distribution of topic possessions. A full representation of the abstract-topic affiliation is generated by running the provided code.

LDA does not only provide a list of extracted topics and their document affiliation, but also the percentage with which a document is considered to cover a topic.

	Topics in Abstracts	Topic Coverage	Abstracts in Topics
Mean	2,91	8,82	33,32
Median	3	7,22	28
Mode	3	6,94	24
Max	8	62,46	178
Min	0	5,00	6
Magnitude	10^0	10^2	10^0

Table 5.2: Number of topics extracted from abstracts, the topic coverages in Percent, and the number of abstracts a Topic is affiliated with.

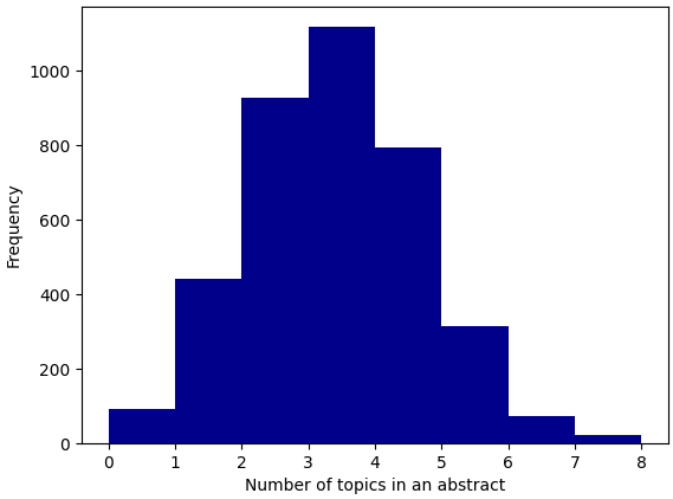

Figure 5.7: Distribution of topics possession within patent abstracts

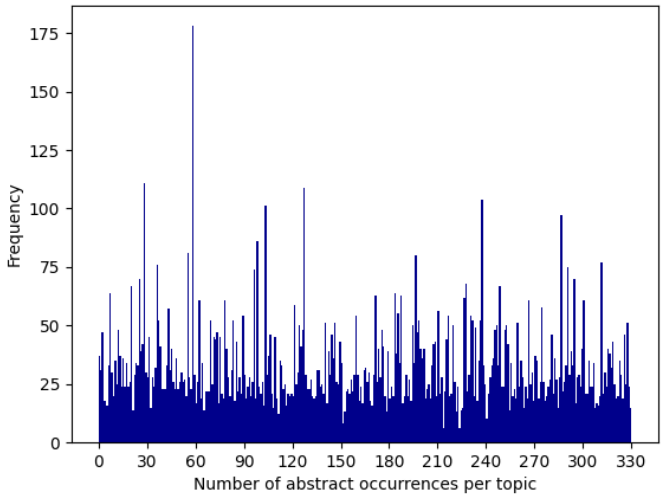

Figure 5.8: Number of abstracts occurrences per topic. Values on the x-axis are not continuous. The distance between topic names (e.g. '0' and '30') is not connected to a thematic or semantic similarity.

The average coverage over all topic affiliations falls on 8,82%. When considering the average number of topics within each abstract, then this score appears rather low. Figure 5.8 provides an overview over how often each topic was covered by a patent abstract. The most common topic carries the id 58 and appears 178 times. The least covered topics are 213 and 223 with a frequency of 6. Once again, table 5.2 provides key statistics.

The topics provided by LDA carry generic names, in form of numbers reaching from zero to the respective hyperparameter given, minus one. For this analysis, these names are kept. However, future efforts might assign names reflecting the topic content, to facilitate a more intuitive evaluation in later steps of this framework. In order to grant a more intuitive understanding of how LDA topics may look like, the term composition of topic 109 is provided below. A full topic representation is generated by running the provided code.

$$\text{Topic } 109 = \{pool, water, clean, tank, underwater, discharge, jet, filter, ...\}$$

	Patents in Windows	Unique Topics in Windows
Mean	222,78	247,17
Median	211	252
Mode	204	254
Max	432	312
Min	53	109
Total	3.781	330

Table 5.3: Number of patents and unique topics per sliding window.

5.3 Data Transformation

In this section, key statistics of the sliding window and network transformation are presented.

5.3.1 Sliding Windows

The challenge associated with the introduction of a sliding window approach lies in picking a suitable window size and sliding interval (see subsection 4.3.1). By consulting the University of Mannheim's chair of Organization and Innovation, a window size of 360 days and a sliding interval of 30 days was deemed appropriate for explorative analyses. Picking a more intuitive window size of 365 would lead to difficulties in the detection of diffusion patterns described in chapter 5.

With this approach, 189 windows are created, spanning over 6.000 days of the total publication time span of 6027 days. The last 27 days are discarded in order to keep the size of all sliding windows identical. Opening an additional sliding window with 357 days was concluded to introduce avoidable heterogeneity to the data structure. Excluding the last 27 days from the analysis changes the date for the last observed publication from 31. January 2018 to 3. January 2018. The reason why the last publication is not dating to 4. January 2018 is found in the publication interval (see chapter 3); The dataset does not contain any patents published on the 4. January 2018.

After slicing the dataset with the described window size and interval, the average number of patents contained in a sliding window equals 222,78. The average number of unique topics within a window equals 247,17. Table 5.3 provides more details regarding the window descriptives.

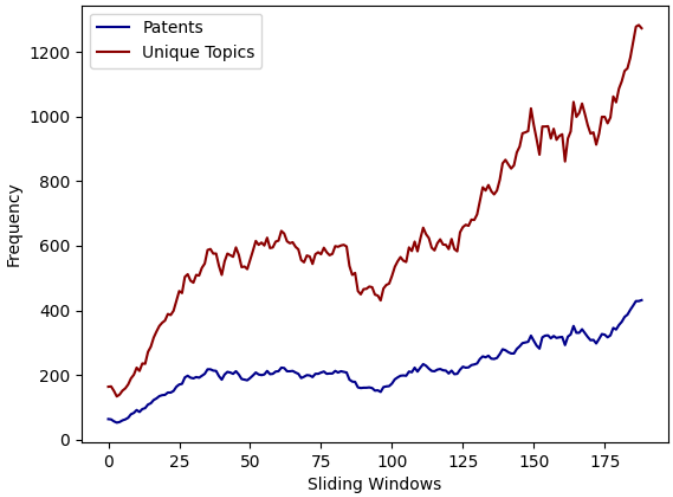

Figure 5.9: Number of patents and unique topics within sliding window. Each value on the x-axis represents a sliding window. Hence, the 0 value on the x-axis represents the first sliding window and the value 189 represents the last sliding window in the dataset.

When regarding the time scale, the number of patents and unique topics within the sliding windows tends to increase, flatten out, dip and increase again. The flattening and dip might indicate a stagnation in robot innovation, followed by one or many breakthrough innovations. The topic distribution mirrors this development on a higher scale, suggesting that this supposed breakthrough did not only influence the quantity of future patent publications, but also their variety. Figure 5.9 displays the complete sliding window span.

5.3.2 Network Representation

The bipartite patent-topic networks are based upon the sliding windows. This leads to the patent node distribution equaling the patent distribution over the windows. The same equivalency holds for topic nodes and unique topic within the sliding windows. The bipartite edges and edge weights are spanned with regard to the patent-topic affiliation and the topic coverage within each patent. However, in order to reduce the noise in the SCMs and CCMs (introduced in section 4.4), not all affiliations are considered, when spanning edges. For each patent within a

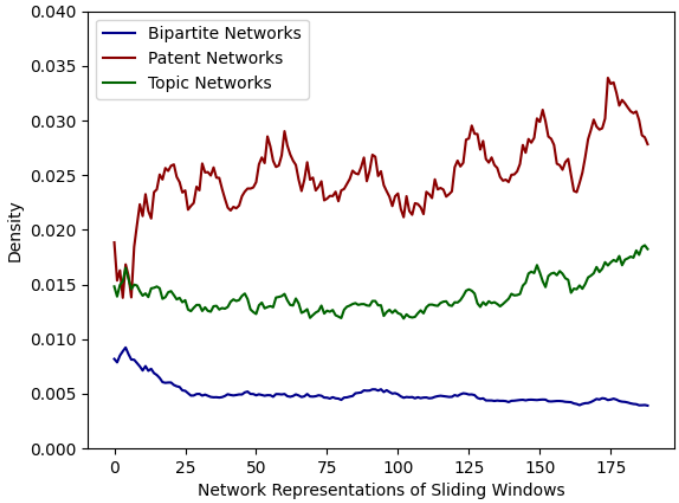

Figure 5.10: Density per network representation. Each value on the x-axis represents a graph constructed from a sliding window. Hence, the 0 value on the x-axis represents the first graph and the value 189 represents the last graph.

window, only the three topics with the highest coverages are represented as edges. This decision as well, is prone to be changed in future efforts[1].

The averaged edge weight average within all bipartite networks falls on 0,09. The averaged degree centrality averages of the patent and topic nodes on 2,42 and 2,13, respectively. The average network density is considered particularly sparse, with a value of $5 \cdot 10^3$. Table 5.4 reveals more bipartite network details. Additionally, figure 5.10 displays the densities of each bipartite network. A short increase in the beginning is followed by a steep decline in network density. After the first 25 sliding windows, the density levels out around $5 \cdot 10^3$, slightly continuing to decrease.

The distribution of nodes within both bipartite network projections equals the respective node distributions in the bipartite network. Tables 5.4 grants more insight.

[1] A better approach would be filtering out the patent topics below a certain quantile, specified over all topic coverages.

Bipartite Network	Edge Weight	Patent Node Centrality	Topic Node Centrality	Density
Mean	9,35	2,42	2,13	5,00
Median	9,32	2,44	2,03	4,76
Mode	9,35	2,27	1,89	8,19
Max	10,35	2,58	3,50	9,24
Min	8,75	2,15	1,06	3,93
Magnitude	10^2	10^0	10^0	10^3

Patent Network	Edge Weight	Centrality	Density
Mean	9,35	5,77	2,52
Median	8,96	5,10	2,50
Mode	8,96	1,19	2,33
Max	13,70	12,43	3,39
Min	7,45	0,72	1,38
Magnitude	10^3	10^0	10^2

Topic Network	Edge Weight	Centrality	Density
Mean	8,07	3,47	1,40
Median	7,98	3,26	1,35
Mode	7,85	2,91	1,32
Max	9,47	5,76	1,86
Min	7,16	1,65	1,19
Magnitude	10^3	10^0	10^2

Table 5.4: Edge Weight averages, Degree Centrality averages and Densities aggregated over all sliding windows of the respective network representations.

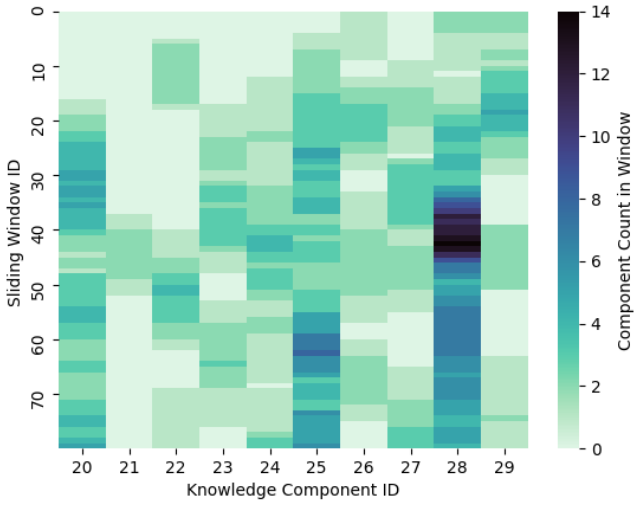

Figure 5.11: Excerpt of the Direct Measurement SCM illustrating the representation of diffusion patterns.

5.4 The Direct Measurement

SCM

The dimensions of the build SCM span over 189 rows and 330 columns. Every row represents a sliding window sorted from the oldest sliding window to the youngest. The 330 columns represent the topics (i.e. knowledge components) extracted from the patent abstracts. As demonstrated in previous sections, every topic is at least once represented in a patent abstract, hence at least once extracted.

Figure 5.11 represents a sample of the Direct Measurement SCM at hand. It is clearly visible, that topic 28 exhibits the strongest diffusion pattern. A diffusion pattern in the SCMs and CCMs is defined as a column sequence containing non-zero elements, encapsulated by zeros. Over the complete SCM 1.568 diffusion patterns are observed. When differentiating between the topics, the average number of diffusion patterns is 4,75.
However, the construction of the SCM causes a diffusion pattern to not automatically represent a diffusion cycle. An exemplary diffusion pattern of

$$1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1$$

is caused by only one patent, diffusing over all twelve sliding windows containing its publication date. In order to account for that, a notion counting diffusion steps needs to be introduced. A diffusion step accurately measures within a diffusion pattern, in how many windows the knowledge component is referred to by at least one new patent joining the diffusion pattern. Excluded from this are the first patents starting the pattern. Otherwise the exemplary diffusion pattern above would count one step, and counter-intuitively indication a diffusion. If a patent contains a knowledge component, but thereafter no other following patents exhibit this knowledge component as well, then it is argued that no knowledge diffusion took place.
An exemplary diffusion pattern of

$$2, 2, 2, 3, 4, 4, 4, 4, 4, 4, 4, 4, 2, 2, 2, 1$$

would exhibit diffusion steps at position 4 and 5 (position count starting from 1)[2]. The algorithm developed to count these steps id not refined enough yet, to be applied. Confronting this issue, the length of the diffusion patterns, approximating this step size is presented. The average pattern length of the direct SCM equals 24,35 sliding windows.

CCM

The CCM spans over 189 sliding windows and 10.030 knowledge component combinations (i.e. columns). Figure 5.12 illustrates diffusion patterns of recombinations. The attached code provides insights into which topic combinations are hidden behind the recombination ids. Displaying all 10.030 recombinations in this chapter is deemed more confusing than helpful.

The same notions used to describe the content of the SCM are facilitated for the CCM description. The presented CCM contains 12.526 diffusion patterns over all windows and recombinations. Broken down on recombination level, the average number of diffusion patterns for each knowledge component combination falls on 1,25. The average pattern length is computed as 11,59.

[2]The extracting algorithm would need to account for disguising cases in which patents leave and join in the same sliding window

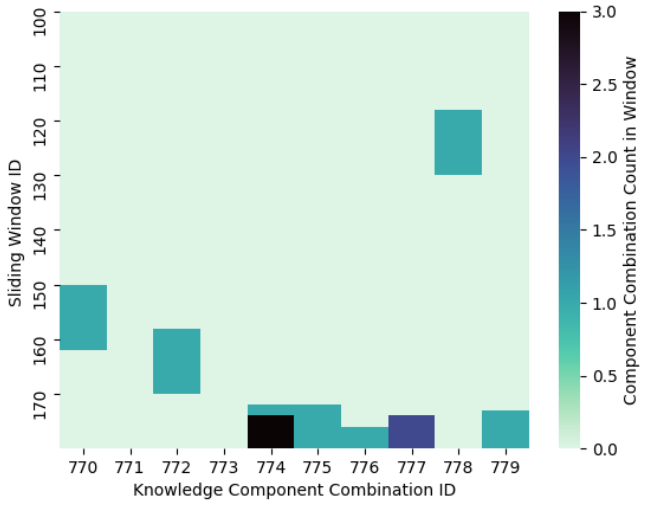

Figure 5.12: Excerpt of the Direct Measurement CCM illustrating the representation of diffusion patterns.

These numbers observed for the diffusions of recombinations are far smaller than the numbers observed for the diffusion of singular knowledge components. Most knowledge component combinations are considered to barely diffuse. This might either indicate that the LDA extraction of topics can be optimized, or that the patent sample drawn from the PATSTAT database exhibits few diffusion of recombination, indeed.

5.5 Community Detection Measurement

Preprocessing

Extracting communities with the Label Propagation algorithm (following LP) resulted in an average Modularity (over all sliding window representations) of 0,67. The next best preforming CDA is the Lais[2] algorithm (following L2) with 0,37, shortly followed by the K-Clique algorithm (following KC) with 0,32 (K = 3). Greedy Modularity (following GM) is preforming worst with a average of 0,05.

A more detailed view is granted in figure 5.13 [3].

[3]The graph displays a y-scale exceeding 1, however Modularity scores over 1 are impossible. The

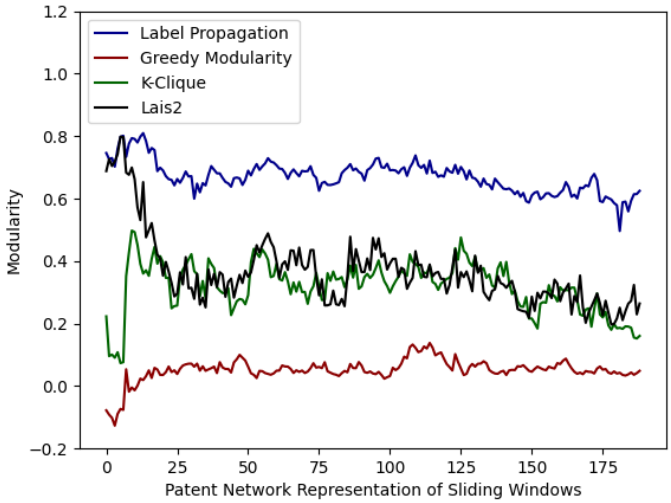

Figure 5.13: Computed Modularity per sliding window for the four CDAs. Each value on the x-axis represents a graph constructed from a sliding window. Hence, the 0 value on the x-axis represents the first graph and the value 189 represents the last graph.

After filtering out all communities with a size of two or less, the LP networks contain on average 26,26 communities within the sliding windows. GM, KC, and L2 average around 4,65, 3,89 and 7,88 communities, respectively. Communities of size one or two are argued to not represent knowledge components. However, this might be adjusted in future efforts as well. The average number of detected communities filtered out falls on 20,08 for LP, 183,61 for GM and 0 for KC and L2. The average community size of all communities remaining is computed as 6,37 (LP), 3,08 (GM), 16 (KC) and 14,47 (L2) The low Modularity and high number of small sized communities for GM, is a first indicator, for its unsuitability.

While pruning the topic distributions of the communities to their most prominent topic, a confidence score, described in the methodology is computed. For LP the average confidence score within topics, averaged over all sliding windows is reported as 0,33. The score in GM, KC and L2 is averaged to 0,4, 0,34 and 0,15. These mediocre scores indicate, that the majority of information contained in the

scale is extended to avoid an overlap between the legend and the curves.

topic distributions of communities is lost, during pruning. This emphasizes the benefit expected, for implementing of a non-flawed community tracing approach. As described in the Methodology, a non-flawed tracing would render the pruning of the topic distributions redundant.

SCM

Identical to all other SCMs, the Community Detection SCMs obtain the dimensions of 189 rows and 330 columns. Similarly to the result description of Direct Measurement, an excerpt of the four SCMs is giving in figure 5.14 and 5.15. The depicted sample of knowledge component IDs and sliding window IDs equals the representation of the Direct SCM in the previous subsection. It becomes apparent, that both crisp and overlapping CDAs find far less diffusion patterns. Whether this is an improvement or a deterioration in finding knowledge diffusion in patent data has to be investigated with more patent innovation expertise.
The number of diffusion patterns observed in the SCMs add up to 1.307 for LP, 390 for GM, 255 for KC and 480 for L2. Their average lengths fall on 3,8, 2,25, 2,89 and 1,98 respectively.

The described notions of diffusion steps is not applicable to the community detection SCMs. The elements building the diffusion patterns in the Direct Measurements are patents. These patents have a static id, invariant over the sliding windows, and always contribute to a pattern for twelve sliding windows (neglecting the patents close to the borders of the publication time span). In this context, the event of patents joining or leaving a diffusion pattern is simple to observe. In the Community Detection Measurement, the elements building SCM diffusion patterns are communities. These communities are aggregations of patents with no static id. There is no fixed period of time a community can or has to participate in the diffusion. Surveying when which communities contributes to a pattern is thereby more challenging and might be of interest, when establishing an approach proving static community labels.

Combating this limitation, the average pattern length for the CDAs is computed as well. LP repots 3,8, GM 2,25, KC 2,88 and L2 1,98. The fact that the community detection SCMs measure far less diffusion of single knowledge components, then the SCM of the Direct Measurement emphasize their different nature. As mentioned above, knowledge about patent innovation might help to identify which measurement results are more desirable.

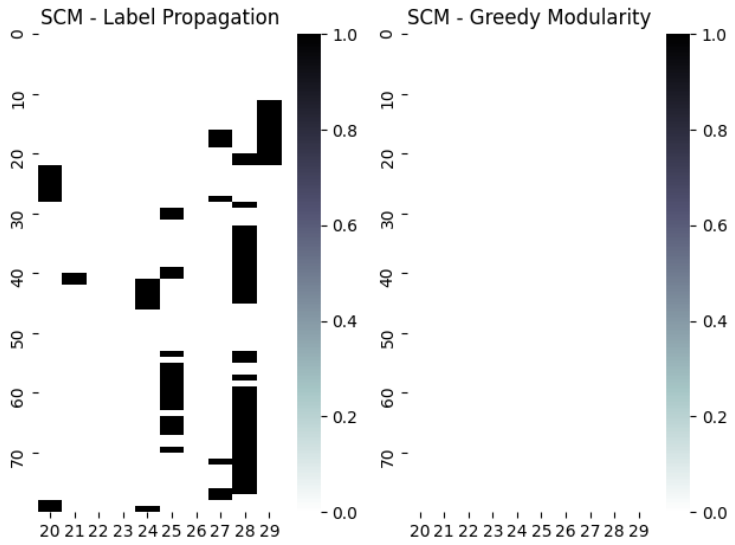

Figure 5.14: Excerpts of the crisp Community Detection SCMs illustrating the representation of diffusion patterns. The X-axis represents knowledge component IDs, the Y-axis Sliding window IDs and the legend the number of components found within a window.

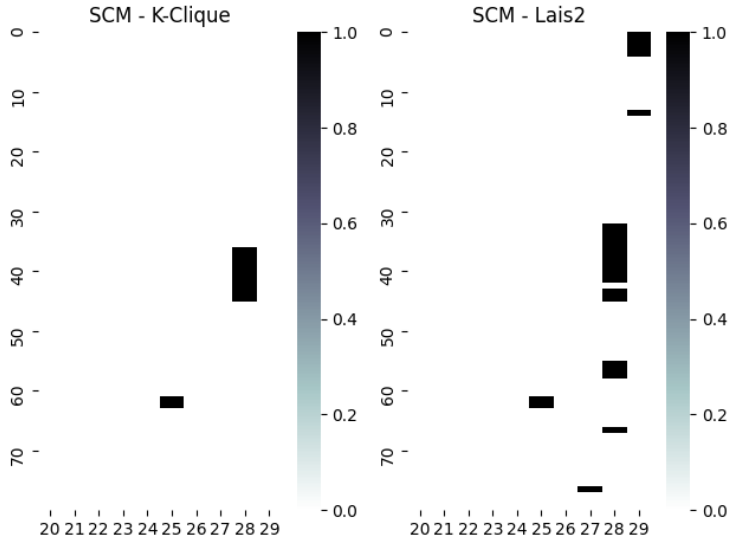

Figure 5.15: Excerpts of the overlapping Community Detection SCMs illustrating the representation of diffusion patterns. The X-axis represents knowledge component IDs, the Y-axis Sliding window IDs and the legend the number of components found within a window.

CCM

The dimensions of the CCMs span over 189 rows and a varying number of columns. This is caused by every CDA identifying different recombinations. This also implies that the recombination ids between the CCMs are not yet comparable. A recombination with id 10 is likely to identify different underlying knowledge component pairs, between the CCMs. LP spans the column dimension of its respective CCM by finding 5.127 recombinations at least once. GM, KC and L2 find 106, 229 and 625. The number of patterns within each SCM is reported as 8.444, 107, 284 and 905 respectively. This results in an average number of patterns per recombination of 1,65, 1,01, 1,24 and 1,45. Similar to the community detection SCMs, the CCMs of GM, KC and L2 tend to find less recombinations and diffusion pattern that the CCM of the Direct measurement. This is illustrated by figure 5.16 and 5.17. Regarding the total number of diffusion patterns and recombinations found, LP shares more similarity with the Direct Measurement, then the other community detection CCMs. This might indicate the forming of two classes of measurements. One inhabited by the Direct Measurement and one inhabited by GM, KC and L2. LP seems to reside in between these classes.

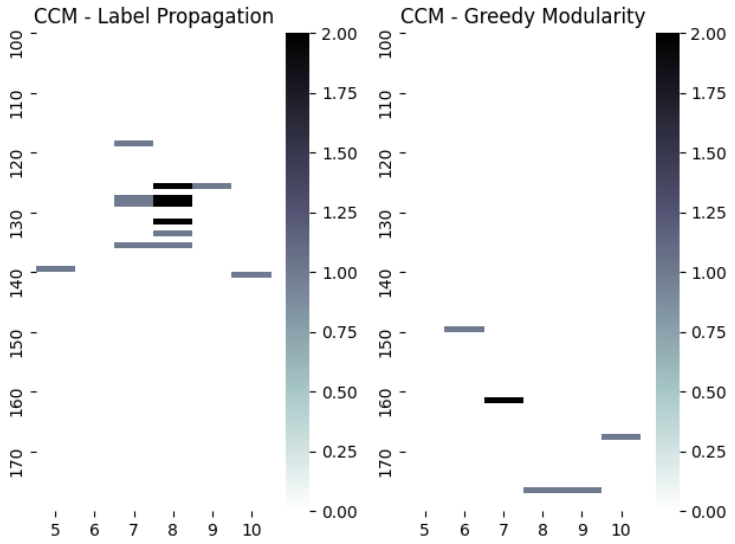

Figure 5.16: Excerpts of the crisp community detection CCMs illustrating the representation of diffusion patterns of recombinations. The x-axis represents the IDs of knowledge component combinations, the y-axis Sliding window IDs and the legend the number of recombinations found within a window.

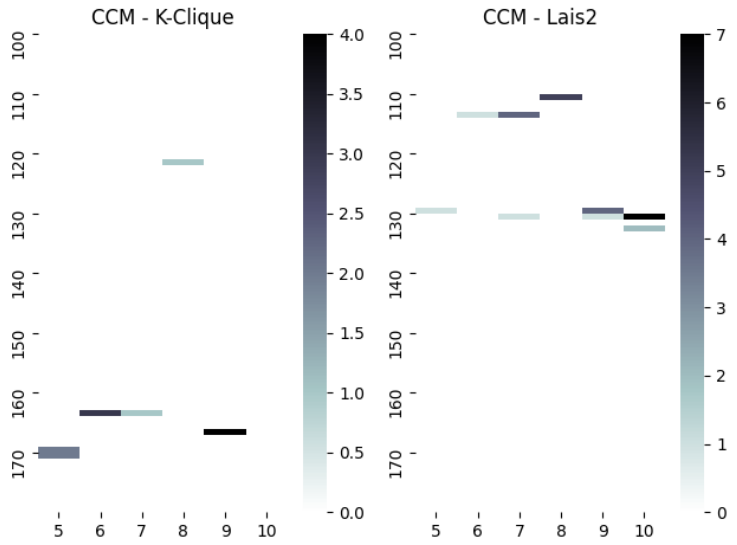

Figure 5.17: Excerpts of the overlapping community detection CCMs illustrating the representation of diffusion patterns of recombinations. The x-axis represents the IDs of knowledge component combinations, the y-axis Sliding window IDs and the legend the number of recombinations found within a window.

For the CCMs, the notions of step size is applicable. The knowledge components themselvess are still communities, however the units recombining the components are patents. This makes the elements of the diffusion patterns patents as well. In contrast to the Direct Measurement, these patents only contribute to recombinations once. This causes the notion of step size to equal the pattern length minus one, and the notion of engaged patents to equal the sum of the diffusion pattern sequence.
The average length of patterns within the LP CCM is computed as 1,29. For GM, KC and L2 this value equals 1, 1,56 and 1,18.

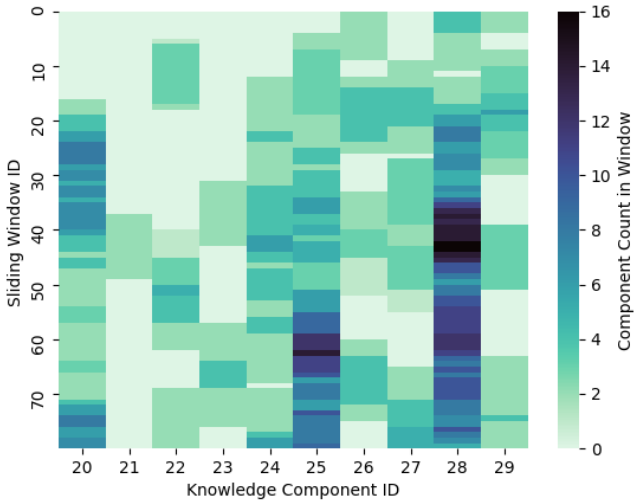

Figure 5.18: Excerpts of the Edge Weight Measurement SCM illustrating the of diffusion patterns of knowledge components. The x-axis represents the IDs of knowledge component, the y-axis Sliding window IDs and the legend the number of edges a topic node is engaged with, within a window.

5.6 Edge Weight Measurement

SCM

At last, the characteristics of the Edge Weight Measurement are reported. The dimensions of the SCM equals 189 windows and 330 topics, once again. The excerpt in figure 5.18 resembles the one of the Direct Measurement. This is caused by both relying on similar grounds. This is elaborated below. Within the SCM, 1.659 diffusion patterns are found, resulting in an average number of 5,14 patterns per topic. The length of the diffusion patterns averages to 21,6.

As for the community detection SCMs the diffusion steps and number of engaged patents is convoluted. The unit contributing to diffusion patterns in the SCM are active topic nodes. Nodes are considered active, if they engage in at least one edge. Edges between topics are aggregations of patents, hence active topic nodes are aggregations of patents as well, yet differently arranged. If this measurement is deemed insightful later on, future efforts might include an extraction of the diffusion step size. However, the similarity between this measurement

and the Direct Measurement, and the more intuitive nature of the latter, might advise focusing on the Direct Measurement.

The differences between the two measurements lies in an additional filtering of patents by the Edge Weight Measurement. While the direct one considers all patents exhibiting at least one topic, the edge weight one disregards patents with only one topic, due to their lack of topic recombination. When not displaying recombinations, patents can not engage in the edges of the topic network projection and thereby can not engage in the activation of nodes. Additionally, attenuation between the SCM and CCM of both measurements is cause by the network construction of the topic network. Within the proposed construction, patents can only contribute to the edges corresponding to their three most prominent topics (or less). Despite this extra filters, the values of the edge weight SCM are greater. This is caused by it counting the topics that remain, twice. A patent with topics A, B and C would contribute to every topic once in the direct SCM. The edge weight SCM would count the edge nodes of (A, B), (A, C) and (B, C).

CCM

The CCM of the Edge Weight Measurement paints a similar picture. Compared to the direct CCM, the diffusion patterns of the excerpt displayed in figure 5.19, are scattered more sparsely. However, it still holds true, that the knowledge component pairs hidden behind the recombination ids are likely to differ. Thus, simply overlapping the CCMs without further alignment steps and drawing inference upon this overlap is not recommended. As first and only component matrix, the entries do not resemble discrete units (i.e. patents or communities), but continuous edge weights of the topic network.

The CCM spans over 189 sliding windows and 6.229 recombinations that are observed at least once. The number of diffusion patterns contained in the CCM falls on 7.304. The average number of diffusion patterns per recombination is computed as 1,17. Diffusion patterns within the Edge Weight Measurement averages to a length of 11,53.

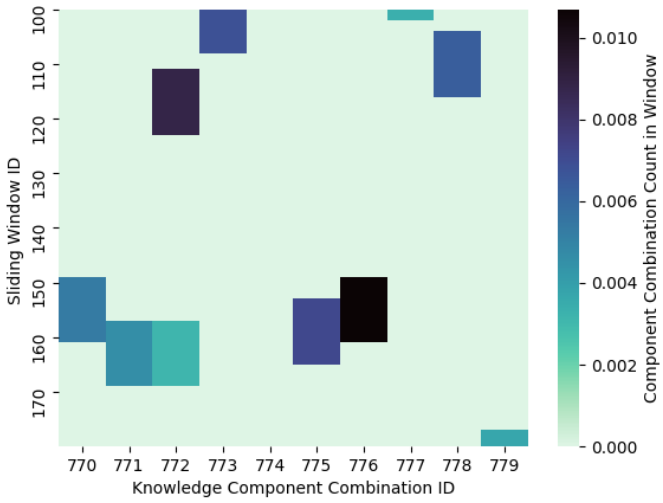

Figure 5.19: Excerpts of the Edge Weight Measurement CCM illustrating the representation of diffusion patterns of recombinations. The x-axis represents the IDs of knowledge component combinations, the y-axis Sliding window IDs and the legend the weight of the topic connecting edge found within a window.

5.7 Comparative Analysis

SCM

After aligning the SCMs as described in the methodology, the number of diffusion patterns over all measurements averages to 472,17. The averaged average length of the patterns falls on 4,23. Table 5.5 provides a more detailed view of the number of patterns and their length after alignment, broken down to the respective SCMs. The threshold used to binarize the SCM is 0,01, meaning that a knowledge component has to account for at least 1% of a windows overall diffusion to not be discarded.

It is observable, that all Community Detection Measurements apart from Label Propagation find considerably shorter diffusion pattern lengths than the Direct and the Edge Weight Measurement. The observed values are a further indicator for Label Propagation to be classified somewhere between the other Community Detection Measurement and the Direct and Edge Weight Measurement.

Aligned SCM	Number of Patterns	Average Pattern Length
Direct	376	5,87
Label Propagation	1047	4,99
Greedy Modularity	347	2,65
K-Clique	221	3,48
Lais^2	375	2,81
Edge Weight	467	5,60

Aligned CCM	Number of Patterns	Average Pattern Length
Direct	4	1,00
Label Propagation	277	6,24
Greedy Modularity	106	9,35
K-Clique	228	12,60
Lais^2	394	10,82
Edge Weight	270	5,40

Table 5.5: Number of diffusion patterns an their length, averaged over the SCMs and CCMs respectively.

Table 5.6 represents the modified Cosine similarities between each aligned SCM pair. In this context, the Cosine similarity between matrices expresses the similarity in diffusion entries (i.e. entries with value 1) only.
The Greedy Modularity approach consistently preforms worst. The only exception is the similarity with Label Propagation. The high score between the Direct and the Edge Weight Measurements emphasizes the common ground they are sharing, once again.

The modified Manhattan similarities in table 5.6 give insight into the similarity between the aligned SCM pairs, by regarding non-diffusion entries (i.e. entries with value 0) as well. However, most of the topics do not diffuse most of the time. The primary interest of this framework lies in finding similarities in diffusion patterns, and not in finding similarities in diffusion and non-diffusion patterns. For this reason, the modified Manhattan similarities are reported but not further described in the following paragraphs.

The average modified Cosine similarity between all pairs is computed as 0,148. The average modified Manhattan similarity as 0,952. The former suggests that the proposed measurements quantify the diffusion of single knowledge components differently. Evaluating which approach is most accurate in its quantification, is reserved for future efforts.

Figure 5.20 [4]. displays the distribution of similarity scores measured for the first 100 knowledge component, between each SCM. The high variety in the Cosine similarities indicates that some topics are measured more similarly between the approaches, then other topics. This suggests, that finding one measurement appropriate for all kinds of topics will be challenging. The average modified Cosine and Manhattan similarity between these 330 knowledge components 0,107 and 0,952

CCM

The alignment of the CCM results in a uniform dimension span of 189 rows and 967 recombinations. The average number of diffusion pattern over all CCMs computes to 213,17. The averaged average pattern length to 7,57. During alignment, the CCM are binarized with respect to a threshold of 0,01 as well. Table 5.5 grants further insights into the diffusion pattern distribution of the CCMs. On average, the averaged pattern lengths in the aligned CCMs are considerable larger than the length in the SCMs. This is caused by the alignment process reducing the number of recombinations to 967. With this reduction, only the strongest (and often longest) patterns survive, leading to an increase in average pattern length.

[4]The graph displays a y-scale exceeding 1, however similarity scores over 1 are impossible. The scale is extended to avoid an overlap between the legend and the curves.

SCM: Cosine Similarity	Direct	Label Propagation	Greedy Modularity	K-Clique	Lais^2	Edge Weight
Direct	1					
Label Propagation	0,154	1				
Greedy Modularity	0,000	0,189	1			
K-Clique	0,157	0,133	0,008	1		
Lais^2	0,212	0,166	0,002	0,239	1	
Edge Weight	0,444	0,164	0,000	0,145	0,208	1

SCM: Manhattan Similarity	Direct	Label Propagation	Greedy Modularity	K-Clique	Lais^2	Edge Weight
Direct	1					
Label Propagation	0,920	1				
Greedy Modularity	0,950	0,919	1			
K-Clique	0,968	0,926	0,973	1		
Lais^2	0,970	0,928	0,968	0,984	1	
Edge Weight	0,976	0,918	0,943	0,962	0,965	1

Table 5.6: Modified Cosine and modified Manhattan similarity for each aligned SCM pair.

CCM: Cosine Similarity	Direct	Label Propagation	Greedy Modularity	K-Clique	Lais^2	Edge Weight
Direct	1					
Label Propagation	0	1				
Greedy Modularity	0	0	1			
K-Clique	0	0,065	0	1		
Lais^2	0	0,086	0	0,092	1	
Edge Weight	0,008	0,001	0	0,002	0,002	1

CCM: Manhattan Similarity	Direct	Label Propagation	Greedy Modularity	K-Clique	Lais^2	Edge Weight
Direct	1					
Label Propagation	0,991	1				
Greedy Modularity	0,995	0,985	1			
K-Clique	0,984	0,979	0,979	1		
Lais^2	0,977	0,974	0,971	0,968	1	
Edge Weight	0,992	0,983	0,987	0,976	0,969	1

Table 5.7: Modified Cosine and modified Manhattan similarity for each aligned CCM pair.

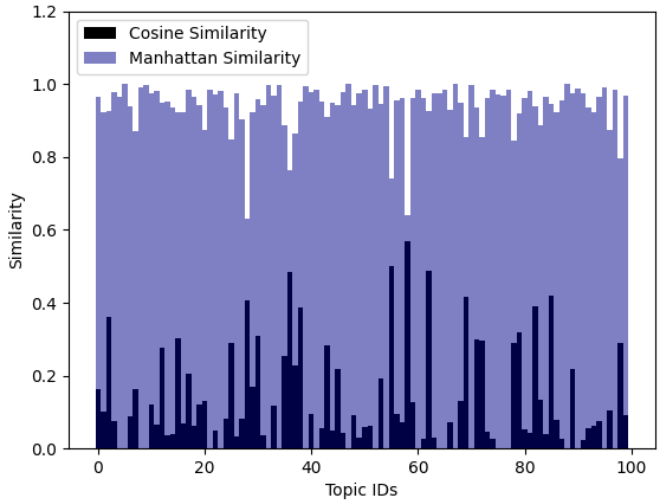

Figure 5.20: Measured similarities between the approaches, broken down on topic level. Only the first 100 topics are displayed.

Two highest average pattern lengths in the two overlapping Community Detection Measurements are most notable. Their singularly measured diffusion events (see chapter 4.6.5) are extend before the normalization and binarization. Nonetheless, this extension significantly increases the average length. In order to omit this introduced bias, future efforts might explore other ways of aligning singularly and plurally measured recombinations. The Direct Measurement shows especially small results. This is caused by the binarization threshold. The small values indicate, that the LP CCM losses most of its diffusion entries because their normalized score is below 0,01.

The measured modified Cosine and Manhattan similarities between the CCMs is reported analogous to the similarities between the SCMs. Table 5.7 shows that the recombination similarities are even lower. Once again, Greedy Modularity seems to differ the most from all other CCMs with a consistent value of 0. And once again, the only considerable similarity is found between the Direct and the Edge Weight Measurement, emphasizing their common ground further. The low similarity of the Direct Measurement is caused by most of its normalized values not meeting the aligning threshold. The Cosine similarity averaged over all pairs scores particularly low with a value of 0,017. The averaged Manhattan value falls on 0,981.

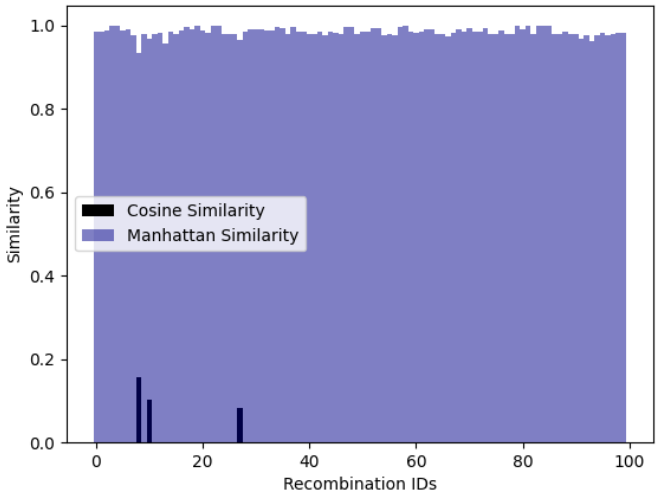

Figure 5.21: Measured similarities between the approaches, broken down on recombination level. Only the first 100 recombinations are displayed.

In figure 5.21 the distribution of similarity scores measured for the first 100 knowledge component combinations, between each CCM is depicted. Most component combinations share a similarity of 0. Few cross a similarity score of 0,1. This indicates that the observation made in the respective SCM similarity is even more drastic, when it comes to the diffusion of recombinations. Most recombinations are uniquely measured within one approach. The Cosine average falls on 0.011 and the Manhattan one on 0.981

5.8 Predicting Recombination

As discussed in the Methodology, this section is merely an introduction to Link prediction. It is highly limited and suffers many shortcommings. A proper approach involves far greater data quantity and quality. Additionally, more sophisticated approaches, then the here presented are recommended.

In the following paragraphs, sliding windows are referred to by their id. This means, that window 0 refers to the first sliding window in the publication span, and window 188 refers to the last, 189th sliding window.

The topic network chosen for the computation of the edge embeddings corresponds to window 150. In this later stages of the publication span, more patents and hence more recombinations are identified, providing a bigger data sample. The second sliding window chosen to predict edges in, is window 162.
As discussed in section 5.6, edges represent aggregations of recombining patents. These patents contribute to the networks for twelve sliding windows.
When choosing a window closer to 150 (i.e. window 151), the set of links present in window 150 would resemble the set of links present in the closer window. Predicting edges in this closer window, with the structural information of window 150 would only partly correspond to predicting future recombinations. Predicting in a completely overlapping window would correspond to the prediction of edges, supposedly missed by the sampling process described in chapter 3.
When trying to circumvent this problem by simply focusing only on the edges joining the graphs between the two (almost) adjacent windows, then the resulting sample size would be to small too train meaningful models on.
Window 162 was chosen, because it is distant enough to have all contributing patents in window 150, leaving the graph. Additionally, it is close enough to hopefully benefit from the structural information embedded in window 150.

Both windows contain 330 topic nodes. Window 150 compresses 665 recombining edges and window 162 compresses 634. Simply predicting the edge weights of these 634 recombinations, would presuppose the existence of these future recombinations. Combating this, fake edges with weight 0 are appended to window 162. Within a network of 330 nodes, the number of all possible, unidirectional edges expands to 108.570 (excluding self-loops). However, appending 107.936 fake edges to the sparse network would skew the dataset heavily. Alternatively, an ambiguous number of 1.902 ($634 \cdot 2$) fake edges is chosen.

After preparing the topic network corresponding to window 150, it is fed to a Node2Vec model, accessed via a *PyTroch Geometric* implementation (Version 2.0.1). The hyperparameters used to specify the model are 128 embedding dimensions, a walk length of 20, a context size of 10, 10 walks per node, a Return parameter of 1 and an In-Out parameter of 1 as well. Additionally, the number of negative samples used for each positive one is set to 1 and the data is declared sparse. The optimizer utilized is Sparse Adam, a variation of Gradient Decent. Adam stands for Adaptive Moment Estimation and additionally stores the exponentially decaying average of past squared gradients and non-squared gradients. [29, 2]. The model was trained for 300 epochs. Figure 5.22 depicts the reported Loss over epochs.

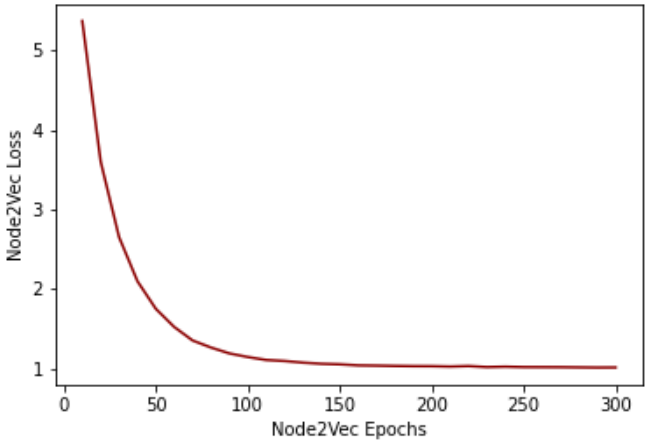

Figure 5.22: The Loss of the described Node2Vec model over 300 epochs.

After retrieving the node embeddings for window 150, the embeddings for the edges of window 162 are derived. This is done, by multiplying the embeddings of two nodes spanning an edge. At last, the edge embeddings and edge weights of window 162 are fed to a XGBoost Regressor model with default hyperparameter settings, provided by the Python library *XGBoost* (Version 0.9), The model was trained using the 10-fold cross validation implementation of the *scikit-learn* Python library (version 0.22.2).
The model reports a RMSE of $4,80 \cdot 10^3$. This value is averaged over all cross validations. With a standard deviation of the edge weight distribution of $4,81 \cdot 10^3$, the model preforms almost as bad as guessing the mean edge weight for every edge to be predicted.
The mean edge weight falls on $2,65 \cdot 10^3$ and the value range is reported between $46,08 \cdot 10^3$ and 0. The grave limitations of this introduction are discussed in chapter 7.

Chapter 6

Conclusion

This thesis presents an explorative framework, quantifying the recombination and diffusion of knowledge in patent data. For this purpose, Latent Dirichlet Allocation was utilized to extract meaningful topics (i.e. knowledge components) from patent abstracts concerned with robot innovation. The first contribution of this thesis lies in verifying previous academic publications, introducing this methodology.

Furthermore, three approaches are introduced, measuring the diffusion of single and combined knowledge components. These approaches can be categorized into two classes, based on their performance. The first class is made up of a Direct Measurement, accessing the extracted knowledge components directly, and an Edge Weight Measurement, relying on a topic network representation of the extracted topics. The second class inhabits a set of Community Detection Measurements. These measurements utilize two crisp and two overlapping detection algorithms to identify knowledge components in a patent network representation of the sampled dataset.
The results suggest that the crisp approach relying on a Label Propagation algorithm might be placed somewhere in between these two classes as well.
The proposed measurements clearly quantify the diffusion of single and combined knowledge components differently. An evaluation of which approach is most suitable, had to be postponed for future efforts and is discussed in more detail in the next chapter.

Tackling this limitation, this thesis presents a method comparing the proposed approaches against each other, without relying on patent innovation expertise. New ideas quantifying knowledge recombination and diffusion can easily be implemented in this comparative method, and the proposed framework in general.

At last, a limited introduction into predicting knowledge recombination is presented. Yet merely a rough sketch, it still hopes to spark more sophisticated ideas for prediction of knowledge recombination in the readers mind.

With an incorporation of the respective expertise, this thesis aims to contribute not only to research concerned with knowledge recombination and diffusion in patent data, but also to research concerned with recombination and diffusion in academic publications. Moreover, the presented framework can easily be extended to cover other written records of knowledge as well. Exemplary use cases involve the identification of diffusion and recombination in transcripted speeches associated with national and international institutions. Moreover, more informal documents like social media posts might reveal interesting patterns as well, when applying the proposed framework.

Chapter 7

Limitations & Future Work

The biggest limitation of this thesis lies in the lack of domain knowledge incorporated. Every intermediate and final result reported is evaluated without expertise in patent innovation. This renders the exploration of the proposed measurements and optimal LDA hyperparameters challenging, at least. Incorporating the evaluation of experts working in patent administration or other related fields, and establishing baselines for how suitable the extracted topics are, and how suitable the measurements of recombination and diffusion is, would constitute the biggest improvement within future efforts. Apart from this, the following limitations and potential improvements are reported.

Reiterating through a LDA grid search with more focus on the alpha parameters might help to improve the quality of the measurements. Especially the Community Detection approaches suffer from the sparse identification of recombination. Higher α values and more expertise might identify a fitting configuration resulting in more topics per patent and a higher average topic coverage, with similar or better topic quality.
Furthermore, a variation of the projection function for the construction of the network projections might represent the underlying recombinations more accurately. Illustrating this, squaring the bipartite edge weights of the current function before multiplying them would penalize skewed edge ratios even more.

When digging into the proposed measurements themselvess, the Community Detection approachs might benefit significantly from a non-simplified measurement. By refining the limited community tracing algorithm, the pruning of the community topic distributions might be omitted, leading to a more differentiated picture of recombination and diffusion.
This refinement could rely on the notions of popular similarity measurements. Communities in adjacent windows might be linked when expressing a high cosine

similarity in topic distribution and a high Jaccard similarity in node membership. Disregarding the nodes leaving and joining the sliding windows, in between t_i and t_{i+1} is deemed promising. For the community detection in general, higher k values for K-Clique are expected to produce more, smaller communities, and hence better results. Furthermore, the highly overlapping nature of the Lais^2 algorithm leads to a slight overestimation of single knowledge component diffusion in the community detection approaches. Additionally, exploring whether merging communities can represent recombination in the community detection approaches might be of interest.

Regarding the comparison of measurements, developing ways to extract the diffusion step size and number of engaged patents form the SCM of the Community Detection Measurements and for the SCMs and CCMs from the other measurement is expected to improve comparability.

Lastly, the prediction of recombination suffers heavily from limitations as well, due to its introductory nature. With adequate resources and enough data of sufficient quality, the employment of deep learning models is assumed to deliver more usable results. In its current form, the proposed introduction is not recommended for adequately predicting recombination. However, when sticking to the proposed node and edge embeddings, a Node2Vec model considering edge weights is expected to improve the resulting embeddings significantly. Additionally, these embeddings might be enriched by document embeddings extracted from the abstracts concerned with a topic node. Finally, the structural and perhaps semantic information of more than one sliding window is expected to improve the prediction.

Bibliography

[1] European patent office; guidelines for examination. https://www.epo.org/law-practice/legal-texts/html/guidelines/e/f_vi_2_1.htm. Accessed: 02.10.2021.

[2] Pytorch - sparseadam. https://pytorch.org/docs/1.9.0/generated/torch.optim.SparseAdam.html. Accessed: 9.10.2021.

[3] Juan Alcacer and Michelle Gittelman. How do i know what you know? patent examiners and the generation of patent citations. *Patent Examiners and the Generation of Patent Citations (August 2004)*, 2004.

[4] Jeffrey Baumes, Mark Goldberg, and Malik Magdon-Ismail. Efficient identification of overlapping communities. In *International Conference on Intelligence and Security Informatics*, pages 27–36. Springer, 2005.

[5] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[6] Eugenio Cavallo, Ester Ferrari, and Mario Coccia. Likely technological trajectories in agricultural tractors by analysing innovative attitudes of farmers. *International Journal of Technology, Policy and Management*, 15(2):158–177, 2015.

[7] Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L Boyd-Graber, and David M Blei. Reading tea leaves: How humans interpret topic models. In *Advances in neural information processing systems*, pages 288–296, 2009.

[8] Ta-Shun Cho and Hsin-Yu Shih. Patent citation network analysis of core and emerging technologies in taiwan: 1997–2008. *Scientometrics*, 89(3):795–811, 2011.

[9] Jinho Choi and Yong-Sik Hwang. Patent keyword network analysis for improving technology development efficiency. *Technological Forecasting and Social Change*, 83:170–182, 2014.

[10] Aaron Clauset, Mark EJ Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical review E*, 70(6):066111, 2004.

[11] Wilfred Dolfsma and Loet Leydesdorff. Lock-in and break-out from technological trajectories: Modeling and policy implications. *Technological forecasting and social change*, 76(7):932–941, 2009.

[12] Giovanni Dosi. Technological paradigms and technological trajectories: a suggested interpretation of the determinants and directions of technical change. *Research policy*, 11(3):147–162, 1982.

[13] Péter Érdi, Kinga Makovi, Zoltán Somogyvári, Katherine Strandburg, Jan Tobochnik, Péter Volf, and László Zalányi. Prediction of emerging technologies based on analysis of the us patent citation network. *Scientometrics*, 95(1):225–242, 2013.

[14] Lijie Feng, Yilang Li, Zhenfeng Liu, and Jinfeng Wang. Idea generation and new direction for exploitation technologies of coal-seam gas through recombinative innovation and patent analysis. *International journal of environmental research and public health*, 17(8):2928, 2020.

[15] Vinicius Eduardo Ferrari, José Maria Ferreira Jardim da Silveira, and Maria Ester Soares Dal-Poz. Patent network analysis in agriculture: a case study of the development and protection of biotechnologies. *Economics of innovation and new technology*, 30(2):111–133, 2021.

[16] Lee Fleming. Recombinant uncertainty in technological search. *Management science*, 47(1):117–132, 2001.

[17] Lee Fleming and Olav Sorenson. Technology as a complex adaptive system: evidence from patent data. *Research policy*, 30(7):1019–1039, 2001.

[18] Roberto Fontana, Alessandro Nuvolari, and Bart Verspagen. Mapping technological trajectories as patent citation networks. an application to data communication standards. *Economics of Innovation and New Technology*, 18(4):311–336, 2009.

[19] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[20] Kenneth Green, Andrew McMeekin, and Alan Irwin. Technological trajectories and r&d for environmental innovation in uk firms. *Futures*, 26(10):1047–1059, 1994.

[21] Thomas L Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235, 2004.

[22] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[23] David Guthrie, Ben Allison, Wei Liu, Louise Guthrie, and Yorick Wilks. A closer look at skip-gram modelling. In *LREC*, volume 6, pages 1222–1225. Citeseer, 2006.

[24] Matthew Hoffman, Francis Bach, and David Blei. Online learning for latent dirichlet allocation. *advances in neural information processing systems*, 23:856–864, 2010.

[25] Zhengyin Hu, Shu Fang, and Tian Liang. Empirical study of constructing a knowledge organization system of patent documents using topic modeling. *Scientometrics*, 100(3):787–799, 2014.

[26] Sarah Kaplan and Keyvan Vakili. The double-edged sword of recombination in breakthrough innovation. *Strategic Management Journal*, 36(10):1435–1457, 2015.

[27] Riitta Katila and Gautam Ahuja. Something old, something new: A longitudinal study of search behavior and new product introduction. *Academy of management journal*, 45(6):1183–1194, 2002.

[28] Dong-hyu Kim, Heejin Lee, and Jooyoung Kwak. Standards as a driving force that influences emerging technological trajectories in the converging world of the internet and things: An investigation of the m2m/iot patent network. *Research Policy*, 46(7):1234–1254, 2017.

[29] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[30] Katsiaryna Krasnashchok and Salim Jouili. Improving topic quality by promoting named entities in topic modeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 247–253, 2018.

[31] Xin Li, Hsinchun Chen, Zan Huang, and Mihail C Roco. Patent citation network in nanotechnology (1976–2004). *Journal of Nanoparticle Research*, 9(3):337–352, 2007.

[32] Farshad Madani and Charles Weber. The evolution of patent mining: Applying bibliometrics analysis and keyword network analysis. *World Patent Information*, 46:32–48, 2016.

[33] Andrew Kachites McCallum. Mallet - topic modeling. http://mallet.cs.umass.edu/topics.php. Accessed: 5.10.2021.

[34] Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[35] Atul Nerkar. Old is gold? the value of temporal exploration in the creation of new knowledge. *Management science*, 49(2):211–229, 2003.

[36] European Patent Office. Patstat. https://www.epo.org/searching-for-patents/business/patstat.html. Accessed: 14.09.2021.

[37] World Intellectual Property Organization. International patent classification (version 2020) guide. https://www.wipo.int/edocs/pubdocs/en/wipo_guide_ipc_2020.pdf. Accessed: 14.09.2021.

[38] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *nature*, 435(7043):814–818, 2005.

[39] Govindan Parayil. Mapping technological trajectories of the green revolution and the gene revolution from modernization to globalization. *Research policy*, 32(6):971–990, 2003.

[40] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[41] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. http://is.muni.cz/publication/884893/en.

[42] Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408, 2015.

[43] Jasjit Singh. Collaborative networks as determinants of knowledge diffusion patterns. *Management science*, 51(5):756–770, 2005.

[44] Vangelis Souitaris. Technological trajectories as moderators of firm-level determinants of innovation. *Research policy*, 31(6):877–898, 2002.

[45] Shaheen Syed and Marco Spruit. Full-text or abstract? examining topic coherence scores using latent dirichlet allocation. In *2017 IEEE International conference on data science and advanced analytics (DSAA)*, pages 165–174. IEEE, 2017.

[46] Peter Thompson. Patent citations and the geography of knowledge spillovers: evidence from inventor-and examiner-added citations. *The Review of Economics and Statistics*, 88(2):383–388, 2006.

[47] Bart Verspagen. Mapping technological trajectories as patent citation networks: A study on the history of fuel cell research. *Advances in complex systems*, 10(01):93–115, 2007.

[48] Hanna M Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation methods for topic models. In *Proceedings of the 26th annual international conference on machine learning*, pages 1105–1112, 2009.

[49] Chong Wang, John Paisley, and David Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 752–760. JMLR Workshop and Conference Proceedings, 2011.

[50] Chao-Chan Wu. Constructing a weighted keyword-based patent network approach to identify technological trends and evolution in a field of green energy: a case of biofuels. *Quality & quantity*, 50(1):213–235, 2016.

[51] Guan-Can Yang, Gang Li, Chun-Ya Li, Yun-Hua Zhao, Jing Zhang, Tong Liu, Dar-Zen Chen, and Mu-Hsuan Huang. Using the comprehensive patent citation network (cpc) to evaluate patent value. *Scientometrics*, 105(3):1319–1346, 2015.

[52] Limin Yao, David Mimno, and Andrew McCallum. Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 937–946, 2009.

[53] Tao Zhou, Jie Ren, Matúš Medo, and Yi-Cheng Zhang. Bipartite network projection and personal recommendation. *Physical review E*, 76(4):046115, 2007.

Appendix A

Additional Preprocessing Information

List of filtered stopwords:
'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselvess', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'to', 'from', 'in', 'out', 'on', 'off', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", "won't", 'wouldn', "wouldn't", 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'ten', 'eleven', 'twelve', 'first', 'second', 'third', 'fourth', 'fifth', 'sixth', 'seventh', 'eighth', 'ninth','tenth', 'eleventh', 'twelfth', 'i', 'ii', 'iii', 'iv', 'v', 'vi', 'vii', 'viii', 'ix', 'x', 'xi','xii', 'also', 'therefor', 'thereto', 'additionally', 'thereof', 'minimum', 'maximum', 'multiple', 'thereon', 'pre', 'kind', 'extra', 'double', 'manner', 'general', 'previously', 'exist', 'respective', 'end', 'central', 'indirectly', 'expect', 'include', 'main', 'relate', 'type', 'couple', 'plurality', 'common', 'properly', 'entire', 'pos-

sible', 'multi', 'would', 'could', 'good', 'done', 'many', 'much', 'rather', 'right', 'even', 'may', 'some', 'preferably', 'input', 'output', 'base', 'basic', 'directly', 'time', 'item', 'work', 'number', 'information', 'make', 'set', 'sequentially', 'subject', 'object', 'define', 'reference', 'give', 'feature', 'determine', 'workpiece', 'action', 'mode', 'function', 'relative', 'reference', 'application', 'describe', 'invention', 'represent', 'task', 'form', 'approach', 'independent', 'independently', 'advance', 'becomes', 'preform', 'parallel', 'get', 'try', 'easily', 'use', 'know', 'think', 'want', 'seem', 'robot', 'robotic', 'robotically', 'robotize'

List of Bigrams formed during abstract preprocessing:
'ac_ac', 'adhesive_feet', 'angular_velocity', 'artificial_landmark', 'aspect_embodiment', 'autonomous_coverage', 'bag_like', 'bevel_gears', 'biped_walking', 'bipedal_walking', 'boundary_wire', 'care_receiver', 'ccd_camera', 'center_gravity', 'claim_included', 'claims_included', 'closed_loop', 'collision_avoidance', 'compressed_air', 'computer_readable', 'degree_freedom', 'degrees_freedom', 'disclosed_herein', 'docking_station', 'dust_collector', 'electricity_meter', 'emergency_stop', 'energy_consumption', 'extension_retraction', 'fabric_product', 'flat_glass', 'flexure_amount', 'floor_reaction', 'following_steps', 'foreign_substances', 'glass_sheets', 'glass_wiping', 'graphical_representation', 'kinematic_chain', 'laser_beam', 'lawn_mower', 'legged_mobile', 'light_emitting', 'loading_unloading', 'magnetic_field', 'master_slave', 'minimally_invasive', 'opening_closing', 'opposite_sides', 'pick_up', 'picking_up', 'picks_up', 'piece_string', 'positional_relationship', 'power_supply', 'present_disclosure', 'prior_art', 'reaction_force', 'real_world', 'self_propelled', 'self_propelling', 'spaced_apart', 'speed_reducer', 'standing_up', 'suction_cup', 'surgical_instrument', 'systems_methods', 'taking_account', 'taught_points', 'teach_pendant', 'teat_cup', 'touch_sensitive', 'umbilical_member', 'up_down', 'welding_torch', 'wind_turbine'

Appendix B

Additional LDA Grid Search Results

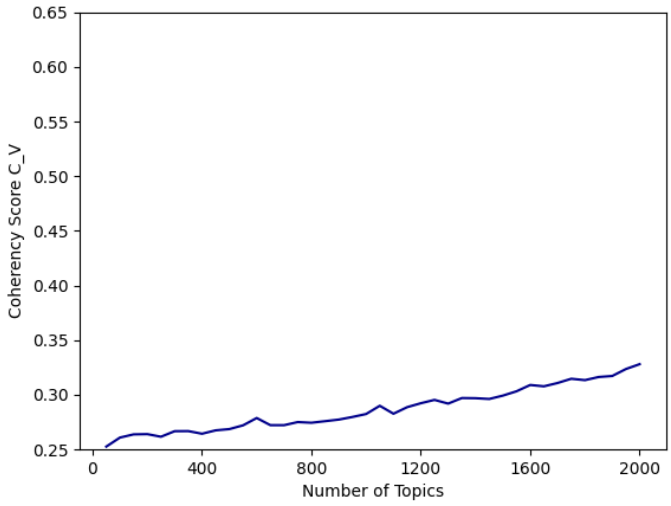

Figure B.1: Extended grid search: Coherency scores of *Gensim* models with varying number of topics, with a fixed β value of 0.01, and a fixed α value of 0.1

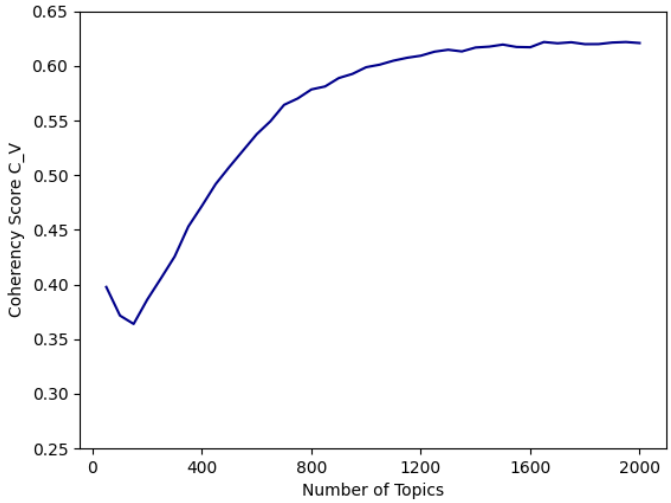

Figure B.2: Extended grid search:Coherency scores of *Mallet* models with varying number of topics, with a fixed β value of 0.01, a fixed α value of 0.1 and a fixed optimization interval of 0.

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass diese Arbeit von mir persönlich verfasst wurde und dass ich keinerlei fremde Hilfe in Anspruch genommen habe. Ebenso versichere ich, dass diese Arbeit oder Teile daraus weder von mir selbst noch von anderen als Leistungsnachweise andernorts eingereicht wurden. Wörtliche oder sinngemäße Übernahmen aus anderen Schriften und Veröffentlichungen in gedruckter oder elektronischer Form sind gekennzeichnet. Sämtliche Sekundärliteratur und sonstige Quellen sind nachgewiesen und in der Bibliographie aufgeführt. Das Gleiche gilt für graphische Darstellungen und Bilder sowie für alle Internet-Quellen. Ich bin ferner damit einverstanden, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs in elektronischer Form anonymisiert versendet und gespeichert werden kann. Mir ist bekannt, dass von der Korrektur der Arbeit abgesehen werden kann, wenn diese Erklärung nicht erteilt wird.

Mannheim, den 13.10.2021 Unterschrift