Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Lise Presterud Hove

Mining Periodic Topic Trajectory Patterns in Spatiotemporal Textual Data

Master's thesis in Computer Science
Supervisor: Kjetil Nørvåg
June 2021

NTNU
Kunnskap for ei betre verd

Lise Presterud Hove

Mining Periodic Topic Trajectory Patterns in Spatiotemporal Textual Data

Master's thesis in Computer Science
Supervisor: Kjetil Nørvåg
June 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Abstract

This thesis explores use-cases in data mining of the newly emerged spatiotemporal textual data type from social media. We define a new kind of pattern in this data that we call a *Periodic Topic Trajectory Pattern* (PTTP). The pattern describes a textual topic that occurs periodically in the same geographical trajectory. As this is a previously undefined pattern, preliminary research is conducted to develop algorithms to identify PTTPs in geotagged social media data. We present similar works that mine lower-dimensional patterns and investigate how we can expand these works to find PTTPs accurately and efficiently. Finally, we implement the expanded algorithms and compare their results and runtimes.

Our findings indicate that spatiotemporal periodic pattern mining extended with a state-of-the-art topic model is the best approach in finding PTTPs, as it scales well and returns accurate results. On the other hand, direct topic modeling approaches are concluded to be problematic as they are challenging to implement efficiently and scales poorly.

We are the first to define a Periodic Topic Trajectory Pattern and explore how to find such patterns. The geographical aspect is challenging to work with, resulting in the approach that handles the locations specifically performing better. We recommend further research in this direction and incorporating word embeddings for more accurate text analysis.

Sammendrag

Denne oppgaven utforsker anvendelser innenfor datagruvedrift av en ny type data, rom-temporal tekstdata, generert fra sosiale medier. Vi definerer et nytt mønster fra slik data som vi kaller periodisk emnebanemønster, på engelsk *Periodic Topic Trajectory Pattern* (PTTP). Mønsteret beskriver et tekstlig tema som gjentar seg periodisk med samme geografiske bevegelse. Da dette er et hittil udefinert mønster, forsker vi innledende på temaet for å utvikle algoritmer for å identifisere PTTP-er i geotagget data fra sosiale medier. Vi presenterer liknende arbeider av lavere dimensjoner og undersøker om og hvordan vi kan utvide disse til å finne PTTP-er presist og effektivt. Til slutt implementerer vi disse algoritmene og sammenlikner dem basert på resultater og kjøretider.

Våre funn indikerer at rom-temporal periodisk mønsterutvinning utvidet med en toppmoderne temautvinningsalgoritme er den beste tilnærmingen for å finne PTTP-er, da den skalerer godt og gir nøyaktige resultater. Derimot, å utvide temautvinningsalgoritmer viser seg å være problematisk, da slike modeller er utfordrende å implementere effektivt, og skalerer dårlig.

Vi er de første som definerer et periodisk emnebanemønster og utforsker hvordan vi finner slike mønstre. Det geografiske aspektet er utfordrende å jobbe med, noe som resulterer i at tilnærmingen som håndterer dette spesifikt yter best resultater. Vi anbefaler videre forskning i denne retningen og å innlemme word embeddings for mer nøyaktig tekstanalyse.

Preface

This thesis is a master thesis completing a five-year Master of Science degree in Computer Science at the Norwegian University of Science and Technology (NTNU). The research was conducted under the supervision of Professor Kjetil Nørvåg at the Department of Computer Science, to whom I would like to extend my gratitude. His guidance and feedback have been of great value to this thesis, and his support and enthusiasm have been a great encouragement to me.

The goal of the thesis was to find new and interesting approaches in mining spatiotextual trajectories from geotagged social media data. As this is a relatively new type of data, we wanted to explore what kind of information is possible to extract and how to extract it. In the autumn of 2020, a specialization project was conducted together with Ingrid Domben to explore the possibilities of spatiotextual trajectories. I found periodicity to have high potential in both usefulness and an exciting research area and decided to research this for my master thesis. I want to thank Ingrid Domben for her insightful discussions during the specialization project.

Finally, I would like to thank my friends and family for their support during my time at NTNU, especially throughout this project.

Lise Presterud Hove
Trondheim, June 19, 2021

Contents

Abstract i

Sammendrag iii

Preface v

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Periodic Information Diffusion . . . 2
- 1.3 Research Questions . . . 2
- 1.4 Contributions . . . 3
- 1.5 Structure . . . 3

2 Problem Definition 5
- 2.1 Data Definitions . . . 5
- 2.2 Subdefinitions . . . 6
- 2.3 Main Definitions . . . 7
- 2.4 Summary . . . 9

3 Related Work 11
- 3.1 Periodic Pattern Mining . . . 11
- 3.2 Periodic Trajectory Pattern Mining . . . 12
- 3.3 Periodic Topic Pattern Mining . . . 13
- 3.4 Geographic Topic Discovery . . . 13
- 3.5 Summary . . . 14

4 Background 15
- 4.1 Periodic Pattern Mining in Time Series . . . 15
  - 4.1.1 Periodograms and Autocorrelation . . . 15
  - 4.1.2 Mathematical Process . . . 16
- 4.2 Spatiotemporal Periodic Pattern Mining . . . 17
  - 4.2.1 Problem Definition . . . 17
  - 4.2.2 Stage 1: Period Detection . . . 18
  - 4.2.3 Stage 2: Mining Periodic Behaviors . . . 19
- 4.3 Topic Modeling . . . 21
  - 4.3.1 Unigram Models . . . 22
  - 4.3.2 Probabilistic Latent Semantic Analysis . . . 22
  - 4.3.3 Latent Dirichlet Allocation . . . 24
- 4.4 Latent Periodic Topic Analysis . . . 25
  - 4.4.1 Generative Process . . . 26
  - 4.4.2 Inferring the Unknown Variables . . . 27
- 4.5 Spatiotemporal Theme Pattern Mining . . . 29

4.5.1 Definitions . . . 29
4.5.2 Generative Process . . . 30
4.5.3 Inferring the Unknown Variables . . . 30
4.5.4 Spatiotemporal Analysis . . . 32
4.6 Summary . . . 33

5 Mining Periodic Topic Trajectory Patterns **35**
5.1 PSTA with Additional Analysis . . . 35
5.1.1 Model . . . 35
5.1.2 Complexity Analysis . . . 36
5.2 LPTA with Geographical Information . . . 37
5.2.1 Model . . . 37
5.2.2 Complexity Analysis . . . 39
5.3 Periodica with Topics . . . 39
5.3.1 Model . . . 39
5.3.2 Complexity Analysis . . . 40
5.4 Summary . . . 41

6 Evaluation Methodology **43**
6.1 Datasets . . . 43
6.1.1 Synthetic Dataset . . . 43
6.1.2 Real Datasets . . . 44
6.2 Preprocessing . . . 45
6.3 Performance Measures . . . 46
6.4 Implementation . . . 47
6.5 Environment . . . 48
6.6 Parameter Estimation . . . 48
6.6.1 PSTA+ . . . 48
6.6.2 GeoLPTA . . . 49
6.6.3 TopicPeriodica . . . 49
6.6.4 Deciding Time and Location Granularity . . . 50
6.7 Summary . . . 51

7 Experimental Results and Evaluations **53**
7.1 Qualitative Results and Evaluations . . . 53
7.1.1 Patterns Discovered by PSTA+ . . . 53
7.1.2 Patterns Discovered by GeoLPTA . . . 55
7.1.3 Patterns Discovered by TopicPeriodica . . . 57
7.1.4 Patterns in the Second Dataset . . . 59
7.1.5 Qualitative Comparison . . . 62
7.2 Quantitative Results and Evaluations . . . 63
7.2.1 Effects of Varying the Dataset Size . . . 63
7.2.2 Effects of Varying the Number of Topics . . . 65
7.2.3 Revised Time Complexity Analysis . . . 67
7.2.4 Memory Usage . . . 68
7.3 Overall Performance . . . 69
7.4 Summary . . . 70

8 Conclusions and Future Work **71**
8.1 Conclusions . . . 71
8.2 Future Work . . . 72

A Complete Outputs **75**
A.1 PSTA+ 75
A.2 GeoLPTA 75

Bibliography **81**

Chapter 1

Introduction

In this chapter, we introduce the thesis topic and stake out the remaining course. This is done by first outlining the motivation and main idea behind this thesis before defining three research questions that set the scope. Further, we present our contributions to data mining research. Lastly, we summarize the structure of the rest of the chapters, organized by the research questions.

1.1 Motivation

Mobile devices have become a natural part of our everyday lives and follow us where ever we go. They track our movements and web interactions, resulting in massive amounts of multidimensional data. With a continuous expansion in movement tracking and mobile web usage, we see a new type of data submerging: *spatiotemporal textual data*. This high-dimensional, partly unstructured data introduces new possibilities and challenges in data mining.

A typical example is geotagged *tweets* from Twitter[1]. A tweet consists of maximum 140[2] characters of text and a timestamp. Additionally, it includes the possibility of tagging the location of the user with a longitude and latitude value, or a specific name of a location. Unlike standard geographical data, the GPS data is not the main element in spatiotemporal textual data but a supplement to a short unstructured text. An example of a tweet is

> **@UserName:** Italy should be the one hosting FIFAAAA world cup next year. Quality football combined with pizza and wine = <3. #worldcup #fifa #italy2022ftw
>
> *11:48AM, May 25, 2021 — Oslo, Norway*

With such a data source, we have access to unlimited data as new tweets are generated faster than we can collect at all times across the globe. These data points allow tracking of how information spreads through interactions, also known as *information diffusion*. Patterns in spatiotemporal textual data reveal the general population's geographical movement of opinions, events, and developments, some of which we might not be aware of.

[1] `https://twitter.com`.
[2] Expanded to 280 in 2017.

Tweets
01.01.15 03:56 (-95.7129, 37.0902)
"I love #chocolate"
01.01.15 03:58 (-72.7295, -65.8627)
"Feelin feverish and tired #ugh"
01.01.15 03:59 (-129.1014, 25.6301)
"BRING ON THE BEATS!!!"
...
31.12.20 23:59 (92.9163, -30.9461)
"HAPPY NEW YEARS!!!"
31.12.20 23:59 (64.7009, 37.3645)
"Yey #new #years coming up"
31.12.20 23:59 (-119.5736, 37.0005)
"10, 9, 8, 7, 6, 5, 4, 3, 2, 1 !!!"

(A) Input.

Patterns	
STTP 1:	Period: 7 days
	Starts on day: 5
	Topic: chocolate, candy, chips
	Movement: (Australia, day 5),
	(India, day 6), (US, day 7)
...	
STTP 8:	Period: 365 days
	Starts on day: 273
	Topic: fever, tired, cough
	Movement: (Chile day 273),
	(US, day 289), (Spain, day 301),
	(Russia, day 344)

(B) Output.

FIGURE 1.1: Input and output of a Periodic Topic Trajectory Pattern mining algorithm.

1.2 Periodic Information Diffusion

When it comes to human- or animal-generated data, an interesting phenomenon is periodic patterns. Humans (and animals) tend to move and live in periodically recurring patterns. Examples are yearly music festivals and award shows, sports seasons, presidential elections, and the weekly periodic behaviors of the weekend and the weekdays. Numerous algorithms identify thematic patterns [1–4]. However, no known algorithm explores the geographical movement, or trajectory, per cycle. Such information can be thought of as *periodic* information diffusion.

A simple example is the yearly flu season. We can say from personal experience that tweeting about flu symptoms will periodically appear every year in the fall. However, if we discover that, e.g., the periodic reporting on flu symptoms most commonly arises in South America before moving to North America and then Europe and Asia, we could not only get a deeper understanding of the origin and spread of the seasonal flu, but we could also detect anomalies and faster identify abnormal flu patterns that might indicate a new disease.

Periodic information diffusion is inherent in spatiotemporal textual data. We wish to find an effective method to transform an input Twitter dataset to a set of patterns describing the periodic information diffusion of the dataset, so-called *Periodic Topic Trajectory Patterns*. Figure 1.1 shows an example input and output of such a method. The transformation requires handling large datasets and possibly numerous patterns to discover, meaning a possible algorithm must be scalable. It is not obvious how to develop such an algorithm to be both efficient and accurate, which is what we wish to discover.

1.3 Research Questions

We define three research questions that define the scope of this thesis. They will guide us in our work, and we will answer the questions in the conclusion of this

paper. They are ordered to work incrementally towards developing possibly multiple pattern mining algorithms while keeping in mind their different strengths and weaknesses. The research questions are:

RQ1: How can a Periodic Topic Trajectory Pattern be formally defined?

RQ2: What algorithms exist that partly solve the problem? Can we expand them, so they fully solve the problem?

RQ3: If yes, what is the performance?

We point out in the relevant chapters when a research question is being answered.

1.4 Contributions

The contributions of this thesis are two-fold. Firstly, we introduce *Periodic Topic Trajectory Patterns* (PTTP). This is a new type of previously unexplored pattern, so part of this thesis defines what a PTTP entails.

Secondly, PTTP mining is a concatenation of textual categorization and periodic pattern analysis both geographically and thematically. These research areas define the starting points for three possible approaches in developing a PTTP algorithm. We find that the current best approach is to expand geographical periodic pattern mining with text analysis.

1.5 Structure

The rest of the thesis is organized as follows. We formally define the thesis problem of finding Periodic Topic Trajectory Patterns in Chapter 2, answering RQ1. Next, Chapter 3 presents related work to this thesis to get a sense of where to start looking for algorithms that partly solve the problem. Furthermore, we present details on the most relevant algorithms discussed that solve parts of our problem in Chapter 4. These algorithms are expanded in Chapter 5, so they fully solve our problem. Consequently, the chapter provides an answer to RQ2. Chapter 6 presents the methodology used during the implementation and evaluation of these algorithms. The experimental results are presented and discussed in Chapter 7, answering RQ3 as we evaluate the performance of each algorithm. We conclude our thesis in Chapter 8 with a summary of what we discovered regarding the three research questions and some additional input for future work.

Chapter 2

Problem Definition

In this chapter, we define the problem of mining Periodic Topic Trajectory Patterns (PTTPs), and we present the notation used in the rest of this thesis. The PTTP Problem is defined incrementally. First, a document and a document collection or dataset are described formally. Next, we then define a topic and definitions related to topic modeling and periodic topics. Furthermore, the geographic elements of a PTTP are defined before we combine them all to define a PTTP and state the thesis problem. The chapter is finalized with a summary.

In addition to these definitions, Table 2.1 summarizes some of the most frequent variables of the thesis and their meanings. To standardize the notation across this thesis, note that some of the definitions and variables defined here are different from the notations used by the papers we present in Chapter 4. From this point, we use the definitions and variables defined in this chapter, and all variables discussed in Chapter 4 are converted to use our notation for consistency.

2.1 Data Definitions

We first define what constitutes a document. Further, we define a dataset containing such documents and describe how we cluster documents within a dataset.

Definition 2.1.1 (Document)**.** Let $d = (W_d, loc_d, time_d)$ be a document where W_d consists of words from the vocabulary V, loc_d is the tagged location of the document so that $loc = (loc_d.x, loc_d.y)$ are respectively the longitude and latitude values of the document, and $time_d$ is the timestamp of the document.

	Definition
$\mathcal{T}$	A set of equally spaced timestamps, $\mathcal{T} = \{t_1, ..., t_{\lvert\mathcal{T}\rvert}\}$.
D	The document collection.
n	The number of documents in D.
V	The vocabulary of D.
N_d	The number of words in a document d.
W_d	Words in document d, $W_d = \{w_1, ..., w_{N_d}\}$ where $w_i \in V$.
K	The total number of topics.
Z	The topic set $Z = \{z_1, ..., z_K\}$.
θ_i	The topic distribution for document $d_i \in D, i \in 1, 2, ..., n$.
φ_{z_k}	The word distribution for topic $k \in 1, 2, ..., K$.

TABLE 2.1: Definitions.

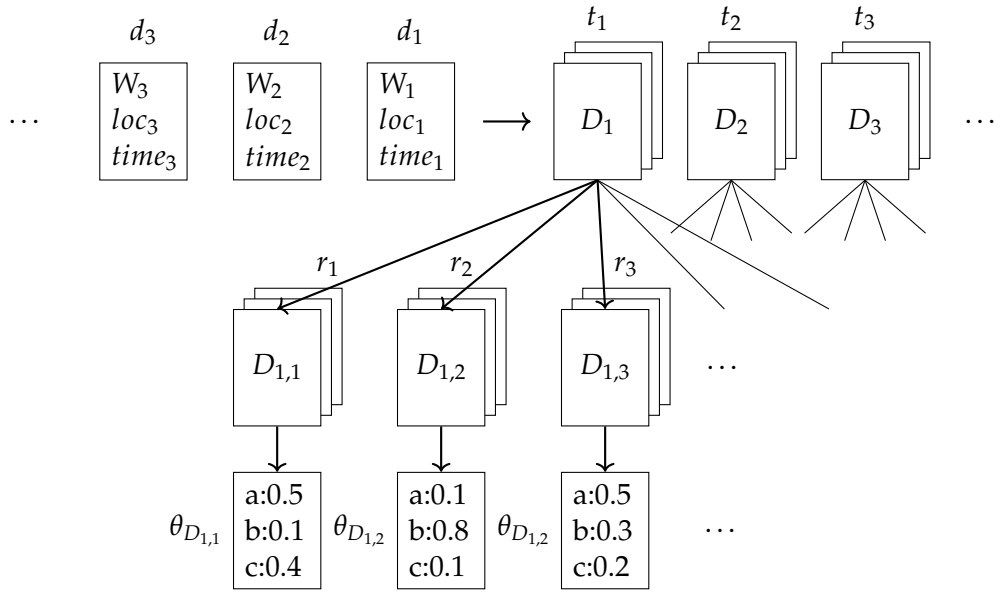


FIGURE 2.1: Data visualization of an input flow categorized and analyzed per time period t_i and geographical region r_j.

Definition 2.1.2 (Dataset)**.** Define D to be the dataset of all documents, i.e. $D = \{d_1, ..., d_n\}$ for $n > 0$. The documents in D are sorted by their timestamp. For a document $d_i \in D$, we simplify the notation so that $d_i = (W_i, loc_i, time_i)$.

Definition 2.1.3 (Timeline)**.** Define a set of equally spaced timestamps $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$, so that $t_{i+1} - t_i = A$ for all $t_i, t_{i+1} \in \mathcal{T}$ and some constant $A > 0$. We say a document d occurs in time $t_i \in \mathcal{T}$ if $time_d \in [t_i, t_{i+1}\rangle$. The time granularity A is set so there is at least one document in D per timestamp in $\mathcal{T}$.

Definition 2.1.4 (Region)**.** A region is a defined geographical area. It can be defined by a set of coordinates, a location name like a city or country, or a set of city or country names. A document d is said to belong to a region r if loc_d is within the region, i.e. if $loc_d \in r$. We define the set of all regions as $R = \{r_1, ..., r_{|R|}\}$.

Definition 2.1.5 (Regional Clustered Dataset)**.** We define a regional clustered dataset, $D_{j,k}$, as the set of all documents in D that occur in time $t_j \in \mathcal{T}$ with location within region $r_k \in R$.

Figure 2.1 illustrates the creation of a regional clustered dataset. The unprocessed dataset, D, is a set of single documents consisting of a set of words, a location, and a timestamp. We divide the timestamps into a discrete timeline and cluster the documents based on their timestamp. This results in those documents within the same timestamp slot being clustered together. The next transformation again subdivides each document set per timestamp by regions. Finally, each $D_{j,k}$ only contains documents with timestamp $time \in [t_j, t_{j+1}\rangle$ and location $loc \in r_k$. From the dataset, we can extract topics, and from each regional clustered dataset, we can extract a topic distribution, as displayed in Figure 2.1. We next define these concepts.

2.2 Subdefinitions

The following definitions are concepts related to the final problem, so-called subdefinitions. We define these concepts separately for clarity before combining them

when we state the thesis problem.

Definition 2.2.1 (Topic)**.** A topic z is defined as a semantically coherent theme, described by a word distribution $\varphi_z = \{p(w|z)\}_{w \in V}$ where $\sum_{w \in V} p(w|z) = 1$. The distribution denotes the probability of each word in the vocabulary V to describe the topic. A set of topics $Z = \{z_1, ..., z_K\}$ has a corresponding set of word distributions $\Phi = \{\varphi_1, ..., \varphi_K\}$ for some number of topics K.

Definition 2.2.2 (Topic Distribution)**.** Given a document $d \in \mathcal{D}$, we define a topic distribution $\theta_d = \{p(z|d)\}_{z \in Z}$ where $\sum_{z \in Z} p(z|d) = 1$. It defines the distribution of all topics $z \in Z$ in d. The probability for a specific topic z in a document d is given by $\theta_d(z)$, and the topic is said to be present in the document if $\theta_d(z) > \epsilon$ for some threshold $\epsilon > 0$.

Definition 2.2.3 (Periodic Pattern)**.** A periodic pattern is a pattern repeating in regular intervals of T in a sequence $S = \{s_0, s_1, \ldots, s_{|\mathcal{T}|}\}$. More specifically, if $s_\tau = s_{\tau+iT}$ for $0 \leq \tau < T$ and $i = 1, 2, \ldots, \lfloor |\mathcal{T}|/T \rfloor$, we say we have a periodic pattern $P_{per} = \langle T, \tau \rangle$, where T is the period and τ the offset within T.

Definition 2.2.4 (Periodic Topic Pattern)**.** Given a document set D, a periodic topic pattern P_{top} is a periodic pattern of topic presences. We define $P_{\text{top}} = \langle T, \tau, \varphi \rangle$ where T denotes the period, $\tau \in \mathcal{T}$ the offset and φ the word distribution representing the theme of the pattern.

Definition 2.2.5 (Trajectory)**.** A trajectory describes a geographical movement, and is defined as a sequence of ordered locations, represented as points of longitude and latitude values or regions, as defined by Definition 2.1.4. We get $Tra = \{l_1, l_2, \ldots\}$ where $l_i = (l_i.x, l_i.y)$ or $l_i \in R$.

Definition 2.2.6 (Spatiotemporal Trajectory)**.** A spatiotemporal trajectory is a trajectory with corresponding timestamps per location. The timestamps can be implicit, i.e. location l_i happens in time $t_i \in \mathcal{T}$, or explicit so that $Tra = \{(l_1, time_1), ..., (l_n, time_n)\}$.

2.3 Main Definitions

We have now defined all the concepts we need to define the Periodic Topic Trajectory Pattern and the problem of finding such patterns. These definitions answer RQ1 (see Chapter 1).

Definition 2.3.1 (Periodic Topic Trajectory Pattern)**.** We define the *Periodic Topic Trajectory Pattern* (PTTP), P_{PTTP}, as a periodic topic pattern that is also periodic in its geographic movement. We write $P_{\text{PTTP}} = \langle T, \tau, \varphi, \boldsymbol{\delta} \rangle$, where

- T denotes the period of the pattern,
- $\tau \in \mathcal{T}$ is the initial offset from the beginning of the timeline to the first occurrence of the pattern,
- φ is the word distribution describing the periodic topic,
- $\boldsymbol{\delta} = \{(l_1, \tau_1), ..., (l_r, \tau_r)\}$ for $r < T$ and $\tau \in \mathcal{T}$ is the explicit spatiotemporal trajectory describing the geographical topic movement per cycle. The timestamp offsets $\tau_i < T$ symbolize the relative timestamps within T when the topic is present in each respective l_i.

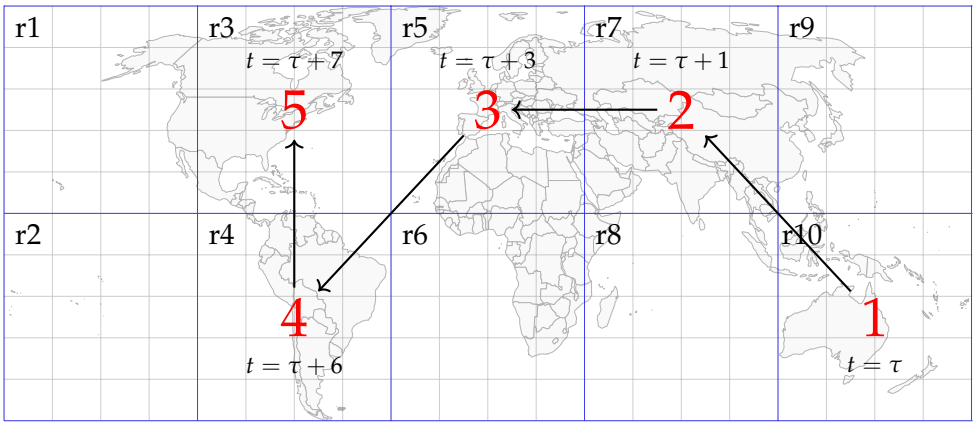


FIGURE 2.2: A global periodic topic trajectory pattern for some topic A with period T, beginning at time τ. The regions are defined by a simple grid structure.

We will also refer to this pattern as a *periodic topic behavior*, as a spatiotemporal trajectory can also be viewed as the behavior of the topic when it repeats itself. A set of PTTPs describes the periodic information diffusion of the dataset.

Definition 2.3.2 (The Periodic Topic Trajectory Pattern Problem)**.** Given a dataset D, find all Periodic Topic Trajectory Patterns, as defined in Definition 2.3.1.

Example

Figure 2.2 illustrates a periodic topic trajectory pattern. The pattern illustrates the topic "New Years", with a period $T = 8760h$ (365 days) and initial offset $\tau = 0$ from the first timestamp of the document collection, in this case, Jan 1, 00:00 AM 2015. The map is divided into regions $R = \{r1, r2, ..., r10\}$ in a grid-based manner for illustrative purposes. We see that the topic starts in Australia (r10) at $\tau_1 = 0h$, i.e., the first hour of the new year. The topic moves to Asia (r7) after one hour. We then get a jump in time to the third hour of the new year, where the topic has moved to Europe and Northern Africa (r5). Note that the jump in time does not mean that the movement stays in region r7 in this second hour but that the topic is not particularly present in any of the regions. We get another jump in time as the topic moves to South America (r4) at $\tau_4 = 6h$ and North America (r3) at $\tau_5 = 7h$. Using the notation from Definition 2.3.1, the pattern is represented as

$$\begin{aligned} P_{\text{PTTP}} = \langle & T : 8760h, \\ & \tau : 0h, \\ & \varphi : \{\text{happy} : 0.171, \text{new} : 0.167, \text{year} : 0.144, \text{years} : 0.142, \text{fireworks} : 0.120, \dots\}, \\ & \boldsymbol{\delta} : \{(r10, 0h), (r7, 1h), (r5, 3h), (r4, 6h), (r3, 7h)\}\rangle \end{aligned}$$

and the time granularity is one hour, i.e. $A = 1h$.

Note that the initial offset τ and the first relative offset of the movement, τ_1, are often the same. However, if the pattern only occurs in a subset of the timeline, the initial offset can be larger. In any case, we have $\tau \mod T = \tau_1$.

2.4 Summary

This chapter answered RQ1 when it formally defined Periodic Topic Trajectory Patterns and the PTTP Problem. Additionally, general notations used in the subsequent chapters were presented, and additional terms were defined. These terms are essential to understand as we continue to the following chapters, which provide a literature review of related works, serving as a basis for solving the newly defined PTTP Problem.

Chapter 3

Related Work

After gaining a thorough understanding of the thesis problem, we delve into related works. This is an initial exploration in finding relevant algorithms that partly solve our problem. First, Section 3.1 presents periodic pattern mining. We then explore extensions of periodic pattern mining in Sections 3.2 and 3.3, discussing periodic trajectory pattern mining and periodic topic pattern mining, respectively. Finally, we look into geographic topic discovery in Section 3.4. Figure 3.1 illustrates how these works are relevant to our work. To our knowledge, there is no prior work on discovering Periodic Topic Trajectory Patterns (PTTPs).

3.1 Periodic Pattern Mining

Periodic data mining was first introduced in a periodicity search by Loether and McTavish in 1993, where they employed statistical methods like Discrete Fourier Transform (DFT) to find cyclic behaviors in time series [5]. The next significant development was introduced by Han et al., who mined *segment-wise* periodic patterns, i.e., patterns where only some of the points in the period repeat [6]. Using a fixed-length user-defined period, they utilized the Apriori principle [7] to mine periodic patterns. Shortly after, they presented an improved method, where they defined the *max-subpattern hit set property*, materialized in a max-subpattern tree [8]. Yang et al. [9–11] defined an information gain metric, later expanded into a *generalized information gain (GIG)*, that penalizes gaps between patterns, resulting in consecutive repeats of the periodic pattern being considered more significant than scattered patterns.

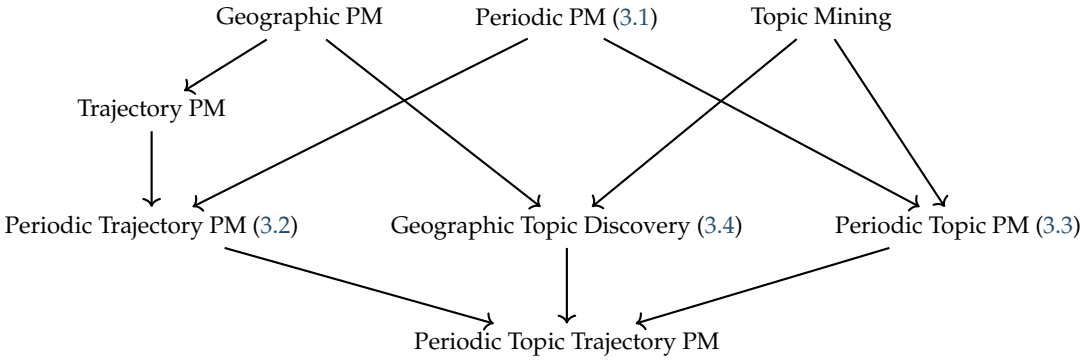


FIGURE 3.1: Related works and how they are connected. Each level adds a dimension, and PM = Pattern Mining.

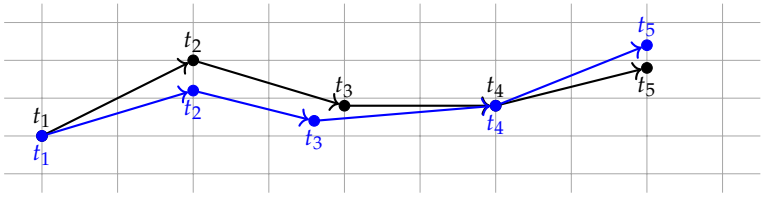


FIGURE 3.2: Illustration of two trajectories consisting of five points with corresponding timestamps.

Ma and Hellerstein [12] criticized the DFT method introduced by Loether and McTavish for not handling noisy data and for having a considerable computational complexity with regards to the number of time units. In agreement with Ma and Hellerstein, Berberidis et al. [13] instead used the autocorrelation function to detect periods and then verified/rejected these periodic tendencies by looking for circular patterns in the data using the max-subpattern tree algorithm by Han et al. [8]. They applied the autocorrelation function to a defined binary vector per event. A binary vector for an event is a vector where value i is set to 1 if the event happens in time $t_i \in \mathcal{T}$ and 0 otherwise. They later expanded the algorithm to handle approximate periods, where they captured "shifted" occurrences by changing the values around the 1's in the binary vector into numbers between 0 and 1, signifying the possibility of a shifted event [14]. Finally, Vlachos et al. [15] found a way to compensate for the problems DFT has with noise by combining periodograms with the autocorrelation function to estimate the most dominant periods in their `AUTOPERIOD` algorithm.

3.2 Periodic Trajectory Pattern Mining

The works mentioned above all focus on time-series data or symbol periodicity, i.e., each data point is assigned a distinct symbol. We now turn our attention to periodic pattern mining on spatiotemporal trajectories. Recall, a spatiotemporal trajectory is a sequence of locations with a timestamp, denoting a movement in the geographical space. Cao et al. [16] argued that locations should not be treated as discrete categorical values as it is likely that coordinates representing the same location will differ slightly due to measuring errors and minor deviations in movement. An example is provided in Figure 3.2, illustrating how two trajectories with the same movement might differ. As a solution, the authors clustered the location coordinates into dense regions and then proposed two distinct methods to find periodic trajectory patterns. The first method identifies patterns directly from the clusters. The second transforms the spatiotemporal datapoints into symbols representing the region to which they belong, enabling the use of an existing categorical periodic pattern mining algorithm [16]. Further, Bar-David et al. [17] analyzed the movement of African buffalo, looking for circular movement patterns. They discovered the periods automatically by transforming location coordinates onto the complex plane and then directly applying DFT to detect periods.

However, as previously mentioned, directly applying DFT is sensitive to noise. Therefore, Li et al. [18] proposed an algorithm, `Periodica`, that detects the period automatically from noisy data using the previously mentioned noise-resistant `AUTOPERIOD` algorithm *per region* and subsequently mines periodic patterns. The geographical coordinates of a moving object are clustered into dense regions, and each such region is, similar to the algorithm by Berberidis et al. [13], related to a binary vector of presence/non-presence in the given region. Next, the binary vector is applied to

the AUTOPERIOD algorithm. Once the periods are detected, a hierarchical clustering method is applied to group segments together to form periodic behaviors [18]. The algorithm handles noisy and complicated data well. However, it is not designed for sparse and inconsistently sampled data.

3.3 Periodic Topic Pattern Mining

Thematic periodic pattern mining concerns itself with finding periodic patterns in document collections. The interest in textual pattern mining started in finding periodic queries in search logs to improve prediction rates. In 2004, Vlachos et al. [1] proposed to convert each query into a time series representing the query demand per day. By applying DFT and then calculating the periodogram, a threshold on the power spectrum was utilized to detect dominant periods in the "signal" [1]. Preoțiuc-Pietro et al. [2] proposed using Gaussian Process models of regression to discover cycles in Twitter hashtags, represented by their frequency in a time series. Gaussian Processes is a probabilistic machine learning framework where defined kernels represent similarities between points, and by defining a periodic kernel, we can identify periodic patterns in the time series [2].

These methods do, however, not extract the *topics* of the query texts, which adds another dimension to the problem. Recall, a topic is a word distribution where high probabilities indicate the term is representative of the topic. Yin et al. [3] developed the mixture model Latent Periodic Topic Analysis (LPTA), which mines periodic latent topics [3]. The model is a variant of latent topic modeling, where a document is considered probabilistically generated by a model of several latent, or hidden, topics, and the goal is to find this model. The difference between LPTA and traditional latent topic models such as Latent Dirichlet Allocation (LDA) [19] is the additional step of not only identifying the topics but also the temporal periodic time distribution of each topic.

A significant downside of LPTA is that it requires the user to input the total number of periodic topics and the period lengths. Wang et al. [4] addressed this issue and proposed a different system, Torpedo. This model defines a time-dependent latent topic model in the same way as LPTA [3]. However, Torpedo models the time distribution per topic directly without any assumptions of periodicity or burstiness. An additional analysis step is applied to the inferred time distribution by inputting it to the AUTOPERIOD algorithm [15] to extract periodic information. The authors of Torpedo compared their model to LPTA, with the conclusion that Torpedo performs better for datasets with a higher degree of deviations, in addition to its ability to automatically configure the period information.

3.4 Geographic Topic Discovery

In (non-periodic) geographic topic discovery, the goal is to identify geographical hotspots for a set of topics. Figure 3.3 illustrates an example of a geographic topic analysis, where four different regions are presented with their inferred topic distributions. This can be at a given timestamp or overall. One such algorithm was proposed by Mei et al. [20], who presented a probabilistic approach to finding spatiotemporal theme patterns in weblog data. They incorporated location and time into a latent topic model to perform Probabilistic Spatiotemporal Theme Analysis (PSTA). A word in this model is dependent on the document's topic distribution

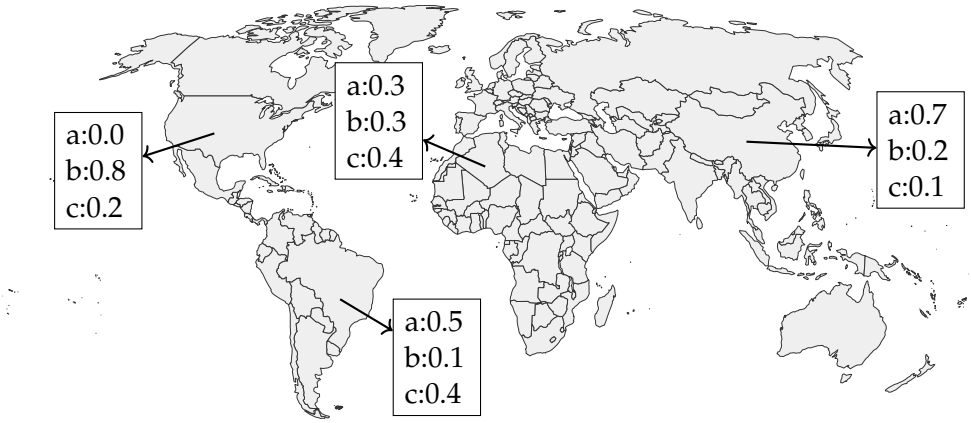


FIGURE 3.3: A geographic topic distribution of three topics a,b and c in four locations.

and the joint location and time topic distribution. The model outputs (1) common themes, (2) theme life cycles for each location, and (3) theme snapshots for each period [20]. This information can describe how each theme evolves geographically. In visualizing the theme life cycles and theme snapshots, it is in theory possible to extract periodic topic trajectory patterns directly, which is the goal of this thesis. This would, however, be manual work, and we aim for automatic extraction.

In general, geographic topic mining research tends to result in LDA-based mixture models with different approaches to incorporate spatial (and temporal) dimensions. For example, Wang et al. [21] integrated geographic information into an LDA-like model they called Location Aware Topic Model (LATM). This model connects topics to geographic multinomial distributions, indicating correlations between topics and geographic locations. However, it does not manage the temporal aspect as the PSTA model does. The authors claim their model solves some of the problems of overfitting and labeling new words in which the PSTA model suffers from [21]. Further, Sizov [22] proposed a model similar to LATM, where the location is drawn from two Gaussian distributions for the longitude and latitude values instead of one multinomial distribution that models a set of regions like in LATM. Yin et al. [23] proposed the model Latent Geographical Topic Analysis (LGTA) based on the assumption that the topics are generated from region distributions only, as opposed to document distributions in standard LDA. Guo and Gong [24] presented a model for spatiotemporal event discovery from social media data, where they demonstrated how to automatically determine the number of topics and regions instead of providing them as user input. Utilizing a Dirichlet Process, they used the Chinese Restaurant Franchise scenario (CRF) [25] to delegate each incoming topic/region to an existing topic/region or a new topic/region in a "rich get richer" approach [24].

3.5 Summary

This chapter presented related work to this thesis. As there is no prior research on PTTP mining, we focused on works closely associated with the thesis problem. The research areas presented were periodic pattern mining, periodic trajectory pattern mining, periodic topic pattern mining, and geographic topic analysis. In the next chapter, we present in further detail some of the works just presented to expand into a PTTP mining algorithm.

Chapter 4

Background

In this chapter, we select the most promising algorithms discussed in the previous chapter to find building blocks to a solution of the Periodic Topic Trajectory Pattern (PTTP) Problem of Definition 2.3.2. We start with periodic pattern mining in time series in Section 4.1. Next, Section 4.2 presents spatiotemporal periodic pattern mining. Further, we explore topic modeling in Section 4.3 and some extensions of topic modeling in Sections 4.4 and 4.5. We conclude the chapter with a summary. The algorithms provided in this chapter serve as a basis when we develop models to solve the PTTP Problem in Chapter 5.

4.1 Periodic Pattern Mining in Time Series

A time series is a series of data points ordered by their occurrence in time. It often represents a signal, and so we use these two terms interchangeably. As discussed in Chapter 3, there have been multiple attempts to use signal processing and statistical methods to extract periodicities in time series data. Unfortunately, most of these methods are noise-sensitive. The `AUTOPERIOD` algorithm by Vlachos et al. [15] uses the strengths of two different functions to alleviate the weaknesses of the same functions, making it resilient to noise. In this section, we present the details of this approach.

4.1.1 Periodograms and Autocorrelation

The algorithm utilizes the periodogram and autocorrelation functions. A periodogram of a time series estimates the power spectral density of the signal per frequency. The frequency is the inverse of the period, and so a high expected signal power indicates a dominant period. However, a periodogram will provide a window of possible periods, and this window can become considerable for long sequences and periods, making it difficult to establish the exact period. The autocorrelation function calculates the correlation of the signal with a delayed copy of itself. It is much more fine-grained than periodograms and can detect specific periods. However, it returns many false alarms and gives less importance to low amplitude events of high frequency than high amplitude events of lower frequency [15].

With these shortcomings in mind, `AUTOPERIOD` combines the periodogram and autocorrelation functions in a two-tier approach. In general, the algorithm uses the periodogram to extract period candidates and then verifies/rejects the candidates based on the autocorrelation function. Vlachos et al. include an illustration of the process in their paper [15]. We simplify and redraw this diagram in Figure 4.1. If the candidate periods lie on a hill in the autocorrelation function, we find the hill's peak

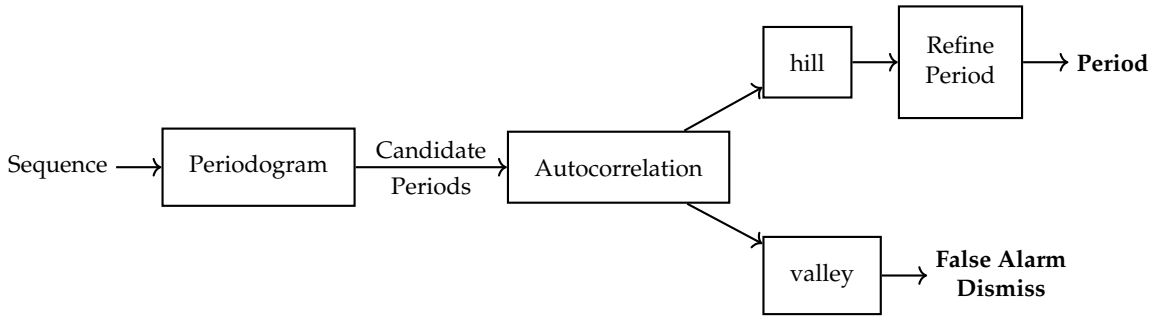


FIGURE 4.1: Diagram of the AUTOPERIOD method, redrawn from Vlachos et al. [15].

and return this as a discovered period of the pattern. If a candidate period lies in a valley, we discard it as a false alarm.

4.1.2 Mathematical Process

We present the mathematical process of AUTOPERIOD following the path illustrated in Figure 4.1. For a non-complex input sequence to AUTOPERIOD, the following steps are made:

1. Transform the sequence into a sequence of complex numbers, $S = s_1 s_2 ... s_n \longrightarrow X_1, X_2, ..., X_n = X$ using Discrete Fourier Transform (DFT):

$$X(f_{k/N}) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} b_{n+1} e^{-\frac{j2\pi kn}{N}}, \quad k = 0, 1, ..., N-1$$

   where k/N denotes the frequency of $X(f_{k/N})$. By first converting the signal to its Fourier transform, we represent the original signal as a "linear combination of the complex sinusoids" [15].

2. Calculate the periodogram of the signal. The periodogram is defined as the squared length of each Fourier coefficient, i.e.,

$$\mathcal{P}(f_{k/N}) = \|X(f_{k/N})\|^2, \quad k = 0, 1, ..., \lceil \frac{N-1}{2} \rceil$$

3. We set a threshold to extract the important frequencies of the periodogram. The threshold is set by permuting S randomly so the permuted sequence S' does not exhibit periodic behavior. The maximum power of the permuted sequence will consequently not indicate a period in the sequence, and is set as the threshold. For a better confidence level, we perform this step 100 times and set the power threshold to the 99th largest value.

4. The frequencies with a high enough power spectrum according to the threshold correspond to a range of periods, i.e., $X(f_{k/N})$ corresponds to periods $\langle \frac{N}{k} ... \frac{N}{k-1} \rangle$. We use the autocorrelation function to extract the exact period. For different delays of the signal, τ, the autocorrelation function is defined as

$$R(\tau) = \sum_{i=1}^{n} s_\tau s_{i+\tau}$$

This function will exhibit peaks for a periodic delay. Therefore, for each frequency $X(f_{k/N})$ with possible periods in the range $[l, r\rangle = [\frac{N}{k}, \frac{N}{k-1}\rangle$, we calculate the autocorrelation series $\{R(l), R(l+1), \ldots, R(r-1)\}$.

5. We fit each calculated autocorrelation series with a quadratic function. If the function is convex, we discard the frequency. If the function is concave, there will be a peak at $t^* = \arg\max_{l \leq t < r} R(t)$ which is returned as a detected period.

The method returns a set of periods discovered in the signal but does not provide information on where the pattern occurs in the time series (its initial offset). Still, it successfully detects the correct periods of a signal and has, in consequence, been used by other algorithms for period detection, like the algorithm discussed in the following section.

4.2 Spatiotemporal Periodic Pattern Mining

We briefly summarize spatiotemporal periodic pattern mining. As discussed in Chapter 3, the difference between spatiotemporal and categorical data is that spatiotemporal data is collected using instruments with error rates in the real world. Hence, different values in the real world can stem from the equivalent location and time, but minor deviations in the measuring devices result in slightly different values. For this reason, we cannot simply convert spatiotemporal data to categorical or time-series data and then find patterns using, e.g., max-subpattern trees [8] or the `AUTOPERIOD` algorithm [16]. A spatiotemporal periodic pattern mining algorithm must include a method to handle these fuzzy datapoints.

Many papers discussing these types of patterns [16, 17] focus on periodic patterns of a single location and not periodic trajectory movements. However, `Periodica` by Li et al. [18] includes an additional step to combine the discovered patterns into a periodic behavior. `Periodica` is a two-stage algorithm to mine *periodic behaviors* in spatiotemporal trajectory data. Stage one detects *reference spots* from which periods are extracted if there are any. Stage two subsequently mines the periodic behaviors using the periods from stage one. For the rest of this section, we will present the `Periodica` algorithm in detail.

4.2.1 Problem Definition

We have an initial dataset as the one defined by Definition 2.1.2, but without the textual element due to the nature of the algorithm. Additionally, the data has to be evenly sampled. If it is not, interpolation is used to obtain a constant time gap between each data point. The new interpolated sequence is denoted $LOC = \{loc_1 loc_2 ... loc_n\}$, where $loc_i = (loc_i.x, loc_i.y)$ is the interpolated location at timestamp $t_i \in \mathcal{T}$. A period T is defined as a regular time interval in the movement. The algorithm clusters the coordinates into reference spots, each defined as a dense area frequently visited. We define the set of reference spots as $O = \{o_1, o_2, ..., o_d\}$ for d reference spots. The reference spot set is the same as the region set R of Definition 2.1.4, clustered on density areas. We let o_0 denote all areas outside the reference spots.

With these definitions, we can define a periodic behavior $\langle T, \mathbf{P}\rangle$. In such a behavior, T is the period and $\mathbf{P}$ is a categorical distribution matrix, where $\mathbf{P}_{ij} (1 \leq i \leq d, 1 \leq j \leq T)$ denotes the probability that the object is at reference spot o_i at relative timestamp T_j. The problem is defined as:

Definition 4.2.1 (Periodic Behavior Mining)**.** Given a length-n movement sequence *LOC*, mine all periodic behaviors $\{\langle T, \mathbf{P}\rangle\}$ [18].

The pseudo-code for `Periodica`, originally presented by Li et al. [18], is displayed in Algorithm 1. Lines 1-4 handle stage one in detecting periods and lines 5-8 mine patterns per period in the second stage. In the following, each stage is further explained.

Algorithm 1: `Periodica` (originally published by Li et al. [18]).

Input: *Sequence* $LOC = \{loc_1, loc_2, ..., loc_n\}$.
Result: A set of periodic behaviors.
`/* Stage 1: Detect periods. */`
1 Find reference spots $O = \{o_1, o_2, ..., o_d\}$.
2 **for** *each* $o_i \in O$ **do**
3 Detect periods in o_i and store the periods in P_i.
4 $P_{set} \leftarrow P_{set} \cup P_i$.
`/* Stage 2: Mine periodic behaviors. */`
5 **for** *each* $T \in P_{set}$ **do**
6 $O_T = \{o_i | T \in P_i\}$.
7 Construct the symbolized sequence S using O_T.
8 Mine periodic behaviors in S.
9 **return**

4.2.2 Stage 1: Period Detection

The first stage of the algorithm is period detection. This step includes two substeps: (1) Finding the reference spots and (2) detecting periods per reference spot. The general logic behind this is that by viewing the data from each reference spot, it is easier to extract the periods as we simplify the data to single locations instead of looking at all of them simultaneously.

Finding Reference Spots

Before we can extract periods we need to identify the reference spots. A reference spot is defined as a dense area, and so they are detected by calculating the density across the map. `Periodica` divides the map into a regular $w \times h$ grid of a desired resolution, and use a known kernel method, a bivariate normal density kernel [26], to estimate the density of each cell. For each grid cell c of a map of C points, the density is estimated as

$$f(c) = \frac{1}{C\gamma^2} \sum_{i=1}^{C} \frac{1}{2\pi} e^{-\frac{|c-loc_i|^2}{2\gamma^2}}$$

where $|c - loc_i|$ is the distance between cell c and the location loc_i and γ is a smoothing parameter determined by σ_x and σ_y, the standard deviations of the whole sequence in its x and y-coordinates, respectively. We get

$$\gamma = \frac{1}{2}(\sigma_x^2 + \sigma_y^2)^{\frac{1}{2}} C^{-\frac{1}{6}}. \tag{4.1}$$

The densities of each cell define contour lines by joining subsequent cells of the same density so that any point within the reference spot has a higher or equal density than

the boundary. To extract high densities, we need to define what constitutes a high density. A natural choice is to choose the top-p% density values. The authors of `Periodica` set p to be 15% [18].

Detect Periods in Each Reference Spot

For each reference spot, a binary sequence B is defined so that $B = b_1 b_2, ...b_n$ where $b_i = 1$ if the object is within the given reference spot at time $t_i \in \mathcal{T}$ and 0 otherwise. We then apply the `AUTOPERIOD` algorithm on the binary sequences to detect periods and store these per reference spot. We have now identified the periodic locations with their respective periods.

4.2.3 Stage 2: Mining Periodic Behaviors

Next, we are interested in movements *between* reference spots. Accordingly, the second step combines reference spots with the same periods and separates the timeline into behaviors containing these reference spots.

Combining the Reference Spots

We combine all reference spots with the same period T so that $O_T = \{o_a, o_b, \dots\}$ are all the reference spots in O with period T. We have this information as each period is detected from each reference spot. The trajectory of locations, $LOC = loc_1 loc_2 ... loc_n$ is transformed to a symbolized trajectory of regions $S = s_1 s_2 ... s_n$, so $o_i \longrightarrow s_i = j$ if loc_i is within region o_j. We let o_0 signify all regions not in O_T.

Separating the Behaviors

We divide the input sequence movement into segments, as there might be multiple movement patterns for the same period. An example is a part-time worker who periodically moves from his/her home to the workplace every Monday at 9:00 AM for nine months. However, the person changes its movement to periodically move to the beach every Wednesday at 1:00 PM during the summer months. The period is the same (7 days), but there are two different patterns associated with this period.

The algorithm for categorizing the segments is presented in Algorithm 2. The trajectory of regions is divided into segments of size T, so we get $\lfloor \frac{n}{T} \rfloor$ segments. I^j denotes the j-th segment and T_k, $(1 \leq k \leq T)$, is the relative timestamp within a segment. Lastly, $I^j_k = i$ means the object is within region o_i in the j-th segment at relative timestamp T_k.

The categorical distribution matrix, $\mathbf{P}$, defines the periodic movement of a behavior. Mathematically, $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_T]$ where $\mathbf{p}_k = [p(x_k = 0), ..., p(x_k = d)]^T$ and x_k is a categorical random variable indicating a selection of a reference spot at relative timestamp T_k. From this, we get that $p(x_k = i)$ signifies the probability that the object is in reference spot o_i at relative timestamp T_k, and $\sum_{i=1}^{d} p(x_k = i) = 1$. In summary, $\mathbf{P}$ is a $d \times T$ matrix where p_{ik} indicates the probability of the object being in reference spot o_i in relative timestamp T_k for $i = 1, 2, ..., d, k = 1, 2, ..., T$. This matrix can represent a periodic movement with some period T.

Algorithm 2: Mining Periodic Behaviors (originally published by Li et al. [18]).

Input: Sequence S, period T, number of clusters X.
Result: K periodic behaviors.

```
Segment S into m = S.length/T segments.
Initialize x = m clusters C_1, ..., C_k, each consisting of one segment.
Compute the pairwise distances among the clusters, d_ij = dist(H(C_i), H(C_j)).
while x > X do
    Select d_st such that s, t = arg min_{i,j} d_ij.
    Merge clusters C_s and C_t to a new cluster C_new.
    Calculate the distances between C_new and the remaining clusters.
    x = x − 1.
return {H(C_i), 1 ≤ i ≤ X}.
```

To find $p(x_k = i)$, suppose the segments $\mathcal{I} = I^a, I^b, \ldots$ all follow the same periodic behavior. We model the probability that $\mathcal{I}$ is generated by $\mathbf{P}$ as

$$P(\mathcal{I}|\mathbf{P}) = \sum_{I^j \in \mathcal{I}} \sum_{k=1}^{T} p(x_k = I_k^j).$$

The best generative model can be found through Maximum Likelihood Estimation (MLE) by finding the optimal solution to the log-likelihood. According to Li et al. [18], the well known solution to this problem gives

$$p(x_k = i) = \frac{\sum_{I^j \in \mathcal{I}} \mathbf{1}_{I_k^j = 1}}{|\mathcal{I}|}, \tag{4.2}$$

where $\mathbf{1}_A$ is 1 if A is true and 0 otherwise. In simple terms, $p(x_k = 1)$ is defined as the relative frequency of reference spot o_i in timestamp t_k over all the segments in $\mathcal{I}$.

The obvious problem here is to find $\mathcal{I}$ for each behavior, i.e., divide all the segments into distinct clusters where each cluster represents one periodic behavior. Identifying these behaviors is the main purpose of Algorithm 2. The notation $\mathbf{H}(C_i)$ in the algorithm represents the periodic pattern of the segments $\mathcal{I}_i$ in cluster C_i, i.e. $\mathbf{H}(C_i) = \langle T, \mathbf{P} \rangle$.

Additionally, the function *dist* returns the distance between two patterns. Li et al. [18] use the Kullback-Leibler divergence between the two clusters' distribution matrices, $\mathbf{P}$ and $\mathbf{Q}$, as a distance measure. The Kullback-Leibler divergence is defined as

$$KL(\mathbf{P}\|\mathbf{Q}) = \sum_{k=1}^{T} \sum_{i=0}^{d} p(x_k = i) \log \frac{p(x_k = i)}{q(x_k = i)}$$

where $p(x_k = i)$ and $q(x_k = i)$ are smoothed with some background variable u and smoothing parameter λ to avoid infinite values, i.e.

$$p(x_k = i) = (1 - \lambda)p(x_k = i) + \lambda u \tag{4.3}$$

and likewise for $q(x_k = i)$. The reader is referred to Li et al. [18] for why this distance metric is chosen. For two periodic behaviors, $\mathbf{H}_1 = \langle T, \mathbf{P} \rangle$ and $\mathbf{H}_2 = \langle T, \mathbf{Q} \rangle$, the

distance between them is defined as

$$dist(\mathbf{H}_1, \mathbf{H}_2) = KL(\mathbf{P} \| \mathbf{Q}).$$

The general idea of Algorithm 2 is to create one cluster per segment, and then combine interactively the two clusters that are most similar until we have our desired number of clusters, X. When two clusters are merged, the new cluster accumulates all the segments in the original clusters, $C_{new} = C_s \cup C_t$, the period is identical for all three clusters and the distribution matrix of the new behavior $\mathbf{H}(C_{new}) = \langle T, \mathbf{P_{new}} \rangle$ is calculated as

$$\mathbf{P_{new}} = \frac{|C_s|}{|C_s| + |C_t|}\mathbf{P_s} + \frac{|C_t|}{|C_s| + |C_t|}\mathbf{P_t}, \tag{4.4}$$

where $|C|$ is the number of segments in cluster C.

Finally, to avoid having to guess the number of clusters, and thereby the number of periodic patterns, Li et al. [18] define a *representation error* to detect when to stop combining clusters. The representation error for behavior $\mathbf{H}(C) = \langle T, \mathbf{P} \rangle$ is defined as

$$E(C) = \frac{\sum_{I^j \in C} \sum_{i=1}^{T} \mathbf{1}_{I_i^j \neq 0}(1 - p(x_i = I_i^j))}{\sum_{I^j \in C} \sum_{i=1}^{T} \mathbf{1}_{I_i^j \neq 0}} \tag{4.5}$$

and the overall representation error $\mathcal{E}_x$ for x clusters is

$$\mathcal{E}_x = \frac{1}{x} \sum_{i=1}^{x} E(C_i). \tag{4.6}$$

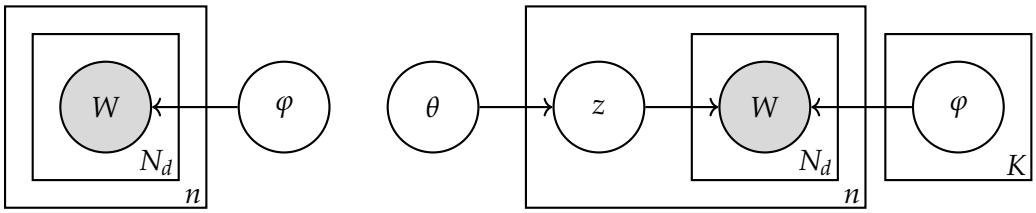
If the overall representation error exhibits a dramatic increase, the newly merged cluster contains two different behaviors. Therefore, by monitoring the representation error, we can stop merging clusters at the appropriate time.

The algorithm returns a set of periodic behaviors $\mathbf{H}_1, \mathbf{H}_2, \ldots$ along with the segments belonging to each behavior. Experiments conducted by the authors of `Periodica` present successful experiments with detailed patterns in noisy and complicated data [18].

4.3 Topic Modeling

We now turn our attention to text categorization. It is not straightforward to extract meaningful topics from a text. A human can easily categorize most terms into overlapping categories because we know what each word means and how each word relates to other words. This is, however, not easy for a computer. A computer sees terms in their position and frequency but does not know their meanings. Different methods have been proposed to extract topics from text, most notably latent topic models.

These models have in common that they assume each document is probabilistically generated by some topic distribution over documents and, for each topic, a word distribution. The method assumes each word in a document is generated by selecting a topic based on the document's topic distribution and then selecting a word based on the chosen topic's word distribution. The goal of latent topic modeling is to find the topic distribution and the word distributions of each topic. This section



(A) Unigram model. (B) Mixture of unigrams model.

FIGURE 4.2: Graphical representations of the simplest forms of topic modeling.

presents the unigram model and Probabilistic Latent Semantic Analysis (pLSA). Further, we briefly touch upon Latent Dirichlet Allocation (LDA), an extension of pLSA and one of the most used latent topic models today.

4.3.1 Unigram Models

In probabilistic topic modeling, each document is viewed as an unordered bag of words (BoW). The unigram model assumes some overall distribution of these words and generates a document by selecting words based on this distribution. Each document probability is thereby a combined probability of its words. The probability of each document d consisting of N_d words is calculated as

$$p(W_d) = \prod_{i=1}^{N_d} p(w_i),$$

where W_d represents the document as a BoW, and each word $w_i \in W_d$ has a probability $p(w_i)$ of occurring in the document collection.

The unigram model does not consider topics in the documents. A simple expansion of the unigram model is the mixture of unigram models. In this case, each topic has a word distribution instead of there being one global word distribution. Therefore, to generate a document, a topic is first selected for the document, and then each word in the document is generated from this topic's word distribution. The probability of each document is now modeled as

$$p(W_d) = \sum_z p(z) \prod_{i=1}^{N_d} p(w_i|z)$$

where z represents a topic and $p(z)$ the probability of this topic.

The graphical models of the unigram model and the mixture of unigram models can be found in Figure 4.2. The unigram model draws N_d words from a global word distribution φ for each of the n documents. The variable z is introduced for the mixture model, representing a topic selected per document in D. This topic decides which word distribution φ to use to generate the N_d words per document d. For K topics, we have K word distributions. The global topic distribution is defined by θ.

4.3.2 Probabilistic Latent Semantic Analysis

By selecting a topic per word generated instead of a per-document, Probabilistic Latent Semantic Analysis (pLSA) [27] extends the mixture of unigrams model. The process results in each document having a unique topic distribution. This is more

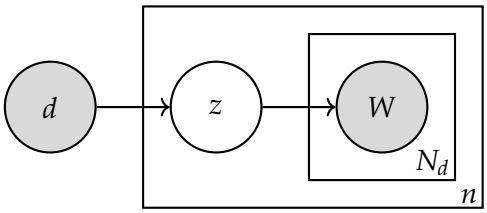


FIGURE 4.3: Graphical representation of the pLSA model.

consistent with real-world data, where a document can mostly discuss, e.g., soccer but also politics, economy, and discrimination. pLSA estimates the probability of each co-occurrence of a document and a word as

$$p(w,d) = p(d)\sum_z p(z|d)p(w|z)$$

where $p(d)$ is the probability of the document, $p(z|d)$ the probability of topic z in document d and $p(w|z)$ is the probability of the word given the selected topic.

Figure 4.3 illustrates the generative process of pLSA, where the topic is drawn given the document, and the word is drawn from the topic distribution like before. pLSA is a probabilistic version of Latent Semantic Analysis (LSA) [28]. LSA can be illustrated by constructing a document-term matrix, and the goal is to decompose it into a document-topic matrix and a topic-term matrix. LSA uses singular vector decomposition (SVD) to achieve this, while pLSA uses a probabilistic approach. Even though the two approaches use different methods, the goal is ultimately the same: decomposing the matrix and extracting the topics.

Inferring the Unknown Variables

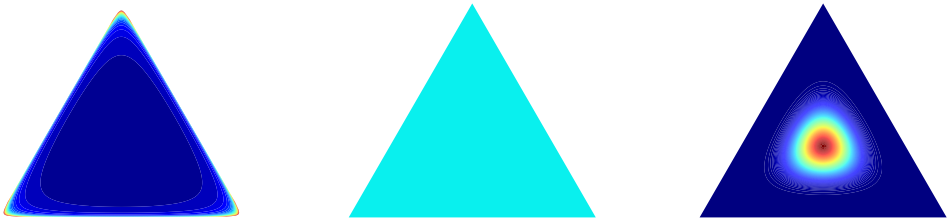
To obtain any information from the pLSA model, we need to solve it, i.e., infer the unknown variables $p(w|z)$ and $p(z|d)$. To do this, we use the Expectation Maximisation (EM) algorithm on the likelihood function. The likelihood function measures how well the model fits the data for different values of the unknown parameters, and the EM algorithm is designed to maximize this likelihood. It contains two main steps: (1) initialize the hidden variables randomly or according to some prior knowledge about the variable, and (2) iterate the E(xpectation)-step and M(aximization)-step until the likelihood function converges. The E-step calculates the expected likelihood of the complete data (the Q-function) for the current parameters. The M-step re-estimates the parameters by maximizing the Q-function.

In pLSA, the E-step is defined as computing the posterior probabilities for the latent, or hidden, variable, $p(z|d,w)$, an introduced variable indicating that word w in document d is generated from topic z. It is calculated as

$$p(z|d,w) = \frac{p(z)p(d|z)p(w|z)}{\sum_{z' \in Z} p(z')p(d|z')p(q|z')}.$$

The three values used in this calculation are the three unknown variables that the algorithm is trying to find. These are updated in the M-step as

$$p(w|z) = \sum_{d \in D} n(w,d)p(z|d,w),$$



(A) $\alpha = 0.1$. (B) $\alpha = 1.0$. (C) $\alpha = 10.0$.

FIGURE 4.4: Dirichlet distributions contour plots for different values of α.

$$p(d|z) = \sum_{w \in W} n(w,d)p(z|d,w),$$

and

$$p(z) = \sum_{d \in D} \sum_{w \in W} n(w,d)p(z|d,w),$$

where $n(w,d)$ is the word count of w in document d. The variables are updated in the E-step and M-step until convergence. The final inferred word distributions per topic is $p(w|z)$ and the final topic distributions per document is extracted using Bayes' rule so that $p(z|d) = (d|z)p(z)/p(d)$.

There are a few issues with pLSA. Firstly, there are no parameters to model $p(d)$, so new documents are hard to analyze. Secondly, pLSA is prone to overfitting as the number of parameters grows linearly with the number of documents. Additionally, using the EM algorithm, we find the *local* maximum, i.e., there might be a higher peak. A straightforward method in solving this problem is to apply the algorithm multiple times with different initialization values and then choosing the values that give the highest maximum value in all the runs. In any case, pLSA is not regularly used in topic modeling, in part because of the aforementioned limitations.

4.3.3 Latent Dirichlet Allocation

The shortcomings of pLSA inspired Blei et al. to develop Latent Dirichlet Allocation (LDA) [19]. They noted that the pLSA model is dependent on the BoW model. Thus the order of the documents is negligible. The authors refer to a theorem of *exchangeability* by de Finetti [29] that "any collection of exchangeable random variables has a representation as a mixture distribution — in general an infinite mixture" [19]. This results in the possibility of capturing the documents' intra-document statistical structure by using a mixture *distribution* and not just a mixture model.

LDA is similar to pLSA initially in that it assumes each document is generated by some topic and word distributions. However, LDA introduces Dirichlet priors on the topic distributions. A Dirichlet distribution is a distribution of distributions. Thus, drawing from the distribution results in a distribution. LDA uses such a distribution to generate the word and topic distributions.

A Dirichlet distribution is parameterized by a vector $\boldsymbol{\alpha}$ of positive real numbers. It provides distributions of some K number of options, where the value of $\boldsymbol{\alpha}$ decides the concentration of the distributions. If all the values of $\boldsymbol{\alpha}$ are the same, we have a symmetric Dirichlet distribution which is parametrized by a single scalar α, which we assume in the rest of this chapter. Figure 4.4 shows the distribution for different values of α with $K = 3$. We see that for $\alpha = 1$, the distribution is uniform. For $\alpha > 1$,

the distribution drawn will tend to cluster in the middle, while $\alpha < 1$ provides sparse distributions that cluster in the corners. LDA uses two distributions parameterized by α and β, respectively, to generate topic distributions per document with high probabilities of a few topics and to generate word distributions per topic with high probabilities of some words. Consequently, we set $\alpha < 1$ and $\beta < 1$ to get sparse distributions that cluster in the corners, like in Figure 4.4a.

Generative Process

The generative process of LDA for a document collection D consisting of n documents of length N_d consists of the following steps.

1. Choose $\theta_i \sim Dir(\alpha)$ for $i \in 1, 2, ..., n$ as topic distributions per document.
2. Choose $\varphi_k \sim Dir(\beta)$ for $k \in 1, 2, ..., K$ as word distributions per topic.
3. For each document d, and for each word position $\hat{n} = 1, 2, \ldots, N_d$:
   (a) Choose a topic $z_{\hat{n},d} \sim \text{Multinomial}(\theta_d)$.
   (b) Choose a word $w_{\hat{n},d} \sim \text{Multinomial}(\varphi_{z_{\hat{n},d}})$.

The geographical representation of the generative process is illustrated in Figure 4.5. The goal of LDA is the same as for pLSA, i.e., to find the word distributions per topic and topic distributions per word. Typical solutions are an approximation of the posterior distribution by Monte Carlo simulation [19], and Gibbs Sampling [30]. We leave out the mathematical details as they are out of scope for this thesis. Due to LDA's ability to classify new documents and avoid overfitting, the model is the preferred topic model as of today.

In recent years, LDA has been further improved by combining it with word embeddings [31], which replaces the BoW representation of text. Word embeddings reduce the dimensionality of the texts and allow topic modeling across languages. We do not explore word embeddings in combination with LDA in this thesis, but we revisit the concept in our discussion of future works in Chapter 8.

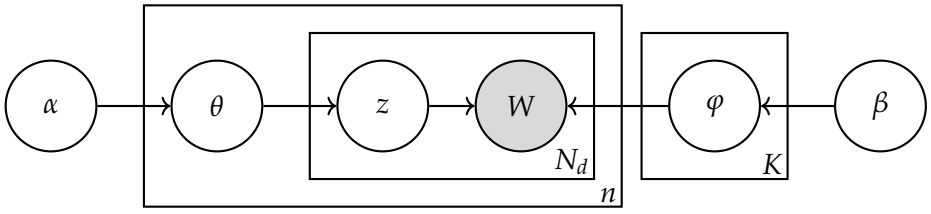


FIGURE 4.5: Graphical representation of the LDA model.

4.4 Latent Periodic Topic Analysis

In topic modeling, the mixture model is defined to model some specific dimensions of the input. This does not only need to be topics like in pLSA and LDA. For example, Yin et al. [3] developed Latent Periodic Topic Analysis (LPTA) to find thematic periodic and bursty patterns in a document collection of timestamped text documents. The model incorporates periodic and bursty time elements into a probabilistic topic modeling scheme by appending a mean timestamp and standard deviation to each topic. In this section, we present this algorithm as it models periodic topics in a unique approach.

Similar to topic modeling, LPTA assumes the document collection input to the algorithm is generated by a latent topic model that needs to be discovered. However, each word in the document is additionally assigned a timestamp, drawn from a distribution dependent on the topic type. The type of each topic is predefined, meaning the user initializes the algorithm with the number of periodic and bursty topics and the period of the periodic topics. Thereafter, LPTA tries to fit a model with the defined number of periodic and bursty topics to the dataset.

4.4.1 Generative Process

Using the same notation as in previous sections, where φ_z is the word distribution for topic z, and θ_d is the topic distribution for document d, we detail the steps taken to generate a document collection in the LPTA algorithm.

For each document d, and for each word w to generate in document d:

1. Choose a topic $z \sim \text{Multinomial}(\theta_d)$.
2. Sample a timestamp depending on the type of z:
   (a) Background topic: $t \sim U(t_{start}, t_{end})$, where t_{start} and t_{end} are the start time and end time respectively of the document collection.
   (b) Bursty topic: $t \sim N(\mu_z, \sigma_z^2)$.
   (c) Periodic topic: $t \sim N(\mu_z + kT, \sigma_z^2)$, where k is sampled from a uniform distribution and T is the periodic interval.
3. Choose a word $w \sim \text{Multinomial}(\varphi_z)$.

Note that steps 1 and 3 are the same as steps 3a and 3b in the generative process of LDA in the previous section. Additionally, LPTA includes a timestamp sampling step where the timestamp is sampled according to the type of the selected topic. If the topic is a background topic, the timestamp is modeled as a uniform distribution, i.e.,

$$p(t|z) = \frac{1}{t_{end} - t_{start}}. \tag{4.7}$$

For a bursty topic, LPTA models the timestamp as a Gaussian distribution as the distribution should contain a single peak, i.e.,

$$p(t|z) = \frac{1}{\sqrt{2\pi}\sigma_z} \exp\left(-\frac{(t-\mu_z)^2}{\sigma_z^2}\right). \tag{4.8}$$

Lastly, for a periodic topic, the topic's time distribution should have multiple peaks. We therefore model a periodic topic as a mixture of Gaussian distributions, i.e.,

$$p(t|z) = \sum_k p(t|z,k)p(k), \tag{4.9}$$

where $p(k)$ is uniform in terms of k and

$$p(t|z,k) = \frac{1}{\sqrt{2\pi}\sigma_z} \exp\left(-\frac{(t-\mu_z - kT)^2}{\sigma_z^2}\right). \tag{4.10}$$

The value of k indicates the periodic interval in which t belongs. Say we divide the timeline into intervals of T. For interval $k = 1, 2, ..., \lceil n/T \rceil$, the value of Equation 4.10 is set to 0 if t does not belong to interval k. For a document d with time t_d that belong to interval I_d, Equation 4.9 becomes

$$p(t_d|z) = p(t_d|z, I_d)p(I_d). \tag{4.11}$$

The generative process defines how the model would have generated the document collection. However, given the document collection, we wish to find the model.

4.4.2 Inferring the Unknown Variables

As described for pLSA, a latent topic model can be solved by maximizing the likelihood function, alternatively the log-likelihood, using the EM algorithm. For the unknown parameters $\Psi = \{\varphi, \theta, \mu, \sigma\}$, we have the log-likelihood function

$$L(\Psi; D) = \log p(D|\Psi) = \log \prod_{d \in D} p(W_d, t_d|\Psi), \tag{4.12}$$

where

$$\log p(W_d, t_d|\Psi) = \sum_d \sum_w n(w, d) \log \sum_z p(t_d|z)p(w|z)p(z|d), \tag{4.13}$$

and $n(w, d)$ is the word count of w in document d.

The Q-function is calculated in the E-step, which calculates the expected log-likelihood of $\Psi^{(t)}$ being the true Ψ, i.e., a measure of how well the model with parameters Ψ fits the data D. We get

$$Q(\Psi|\Psi^{(t)}) = E_{D|\Psi^{(t)}} \log L(\Psi; D) \tag{4.14}$$

in iteration t of the EM algorithm. Next, the M-step maximizes the expected value so that

$$\Psi^{(t+1)} = \arg\max_{\Psi} Q(\Psi|\Psi^{(t)}). \tag{4.15}$$

Just as for pLSA, we approximate the Q-function by defining a latent variable $p(z|d, w)$, defined as the probability that word w in document d was generated by topic z. The latent variable is used to approximate the unknown variables of Equation 4.13. Finally, we iteratively update the latent and unknown variables in the E-step and M-step, respectively, to find the local maximum. Note that we do not need to explicitly calculate the values of Equations 4.14 and 4.15, but simply the variables necessary to calculate them.

E-step

In the E-step of the EM algorithm, we update the latent variable according the time distribution of the topic, the word distribution of the topic and the topic distribution of the document. We get

$$p(z|d, w) = \frac{p(t_d|z)p(w|z)p(z|d)}{\sum_{z'} p(t_d|z')p(w|z')p(z'|d)}.$$

The latent variable is then used in the M-step to update the unknown variables.

M-step

The variables to update in the M-step are the word distributions $p(w|z)$, the topic distributions $p(z|d)$ and the time distributions per topic $p(t_d|z)$. We update them using the following equations:

$$p(w|z) = \frac{\sum_d n(w,d)p(z|d,w)}{\sum_d \sum'_w n(d,w')p(z|d,w')},$$

$$p(z|d) = \frac{\sum_w n(w,d)p(z|d,w)}{\sum_w \sum'_z n(w,d)p(z'|d,w)}.$$

$p(t_d|z)$ is defined by Equations 4.7, 4.8 and 4.11 depending on the topic type. These are again dependent on μ and σ, which are also updated at this point. If z is a background topic, no changes are made as the timestamps for background topics are not dependent on μ and σ. For bursty topics, μ_z and σ_z are updated as

$$\mu_z = \frac{\sum_d \sum_w n(w,d)p(z|d,w)t_d}{\sum_d \sum_w n(w,d)p(z|d,w)},$$

$$\sigma_z = \sqrt{\frac{\sum_d \sum_w n(w,d)p(z|d,w)(t_d - \mu_z)^2}{\sum_d \sum_w n(w,d)p(z|d,w)}}.$$

If the topic is periodic, recall the timeline is partitioned into intervals of length T, and I_d is the corresponding interval of document d to be used in Equation 4.11. μ_z and σ_z are updated as

$$\mu_z = \frac{\sum_d \sum_w n(w,d)p(z|d,w)(t_d - I_d T)}{\sum_d \sum_w n(w,d)p(z|d,w)}, \tag{4.16}$$

$$\sigma_z = \sqrt{\frac{\sum_d \sum_w n(w,d)p(z|d,w)(t_d - \mu_z - I_d T)^2}{\sum_d \sum_w n(w,d)p(z|d,w)}}, \tag{4.17}$$

where subtracting the interval number T times from the timestamp results in the relative timestamp T_i within the periodic movement of the document. The EM steps are repeated until convergence, after which we can interpret the results.

Inferred Results

After convergence, we have inferred word distributions per topic $\varphi_z = \{p(w|z)\}_{w \in V}$ and topic distributions per document $\theta_z = \{p(z|d)\}_{z \in Z}$. Additionally, LPTA provides a mean timestamp μ_z and standard deviation σ_z per topic. Note that each topic is linked to only one pattern. If a topic occurs in multiple patterns, it needs to be modeled multiple times with different periods.

One of the main drawbacks of LPTA is that the user must specify the number of bursty topics and the number of periodic topics as parameters to the algorithm and the period of each periodic topic. These parameters are in most cases not known, and the wrong number of topics or the wrong period might lead to poor results. The authors of LPTA mention that it is possible to utilize Schwarz's Bayesian information criterion (BIC) to set the periods (and the number of topics) [3]. A BIC value

measures the log-likelihood of the model and the model complexity. We use the criterion by training the model with different parameters and selecting the model with the lowest BIC value.

4.5 Spatiotemporal Theme Pattern Mining

A different variant of the latent topic model considers the spatiotemporal aspect. As far as we know, there is no published research on *periodic* spatiotemporal topic discovery. However, multiple research papers consider the spatial [21–23], periodic temporal [3, 4] or the spatiotemporal [20, 32] aspect of a topic model.

This section describes the Probabilistic Spatiotemporal Theme Analysis (PSTA) method proposed by Mei et al. [20], which incorporates the spatiotemporal dimension into a topic model. We focus on this method despite its relatively old age (2006) as it includes both the spatial and temporal dimensions, and it does not contain any additional dimensions out of scope for this thesis. In a pLSA inspired algorithm, Mei et al. mine themes from geotagged and timestamped documents and describe these themes' temporal and spatial patterns.

4.5.1 Definitions

The document collection D for PSTA is that of Definition 2.1.2. Next, each location and timestamp value per document is categorized into discrete values, so we get $\tilde{D} = \{(W_1, \tilde{t}_1, \tilde{l}_1), ..., (W_n, \tilde{t}_n, \tilde{l}_n)\}$ where $\tilde{t}_i \in \mathcal{T}$ and $\tilde{l}_i \in L$. The location set is defined as a region set R of Definition 2.1.4 clustered on names. Like previously, φ_z is the word distribution for topic z, and θ_d is the topic distribution for document d.

Moreover, we define the theme life cycle of a topic in a location as a conditional probability distribution of a timestamp given a topic and location, i.e., $\{p(t|z,l)\}_{t\in T}$. The overall life cycle of a topic is defined as $\{p(t|z)\}_{t\in T}$ if no location is specified. To model spatial patterns, the theme snapshot is defined as a conditional probability distribution of a theme and location for a given timestamp, i.e., $\{p(z,l|t)\}_{z\in Z, l\in L}$.

To be able to perform spatiotemporal theme analysis, the authors define three goals of PSTA:

- Discover the global themes of the documents with their word distributions, $\Phi = \{\varphi_1, ..., \varphi_k\}$.
- For each theme and location pair, compute the life cycle $p(t|z,l)$ for all $t \in T$. The life cycle will provide information on how relevant a theme is in its location over time.
- For each time period, compute the theme snapshot $p(z,l|t)$ for all $z \in Z$ and $l \in L$. The theme snapshot will provide information on what topics are most relevant per location in a given timestamp.

A standard latent topic model accomplishes the first goal. By adding spatiotemporal traits to this model, the authors also achieve the second and third goals. The PSTA model is detailed next.

4.5.2 Generative Process

The general idea of PSTA is similar to pLSA and differs only in the additional spatiotemporal aspect. Further, a standard step in pLSA not previously mentioned is to define a background topic with word distribution φ_B. This topic is separate from the "thematical" topics and is designed to attract common terms with low informational value. PSTA similarly defines a background topic with an occurrence rate of λ_B. Consequently, each word in a document is generated either from the background topic or a thematical topic. Moreover, the thematical topics are generated either from a topic distribution per document, $\theta_d = p(z|d)$, or from a topic distribution per timestamp and location pair, $\theta_{l,t} = p(z|l,t)$. The proportion of each element is defined by $(1 - \lambda_{TL})$ and λ_{TL}, respectively. We can now define the generative process for a document collection D.

For each document d with a location l and timestamp t, and for each word w to generate in document d:

1. Choose the background model with probability λ_B or the theme model with probability $1 - \lambda_B$.
2. Choose a topic z based on the model.
   - If the background model is chosen, choose the background topic $z = B$.
   - If the theme model is chosen, select a topic distribution θ as the topic distribution of the location and timestamp, $\theta_{t,l}$, with probability λ_{TL}, or the topic distribution of the document, θ_d, with probability $1 - \lambda_{TL}$.

     Then sample a topic $z \sim \theta$.
3. Sample a word $w \sim \varphi_z$.

This is the assumed generative process, and the task for PSTA is to fit this model optimally with the input dataset. Similar to the previous topic models discussed, this is done by defining a mixture model and using the EM algorithm to estimate the unknown variables.

4.5.3 Inferring the Unknown Variables

PSTA is solved in the same manner as the previously mentioned topic models. We define a mixture model based on the generative process, and infer the unknown variables using the EM algorithm. The mixture model is defined as

$$p(w : d, t, l) = \lambda_B p(w|z_B) + (1 - \lambda_B) \sum_{z \in Z} p(w, z|d, t, l),$$

where $p(w, z|d, t, l)$ is the probability of the word and topic given the theme model. We decompose it using Bayes' rule to

$$\begin{aligned} p(w, z|d, t, l) =& p(w|z)p(z|d, t, l) \\ =& p(w|z)((1 - \lambda_{TL})p(z|d) + \lambda_{TL}p(z|t, l)). \end{aligned}$$

The mixture model is applied to a log-likelihood function, so that

$$\begin{aligned}\log p(D) &= \sum_{d\in D}\sum_{w\in V} n(w,d)\log p(w:d,t,l)\\ &= \sum_{d\in D}\sum_{w\in V} n(w,d)\log[\lambda_B p(w|z_B)\\ &\quad + (1-\lambda_B)\sum_{z\in Z}(p(w|z)((1-\lambda_{TL})p(z|d)+\lambda_{TL}p(z|t,l)))].\end{aligned}$$

Recall that $n(w,d)$ is the word count of w in document d. The λ values can be fixed empirically, and the background topic is set to the relative frequency of each word for simplicity. We get

$$\varphi_B = p(w|z_B) = \frac{\sum_{d\in D} n(w,d)}{\sum_{w\in V}\sum_{d\in D} n(w,d)}.$$

The ultimate goal is to find the unknown parameters of the mixture model, $p(w|z)$, $p(z|d)$ and $p(z|t,l)$, using the EM algorithm. To do so, PSTA utilizes two hidden variables. First, $p(z|d,w)$ is the probability that word w in document d was generated from topic z. Second, $p(z,t,l|d,w)$ is the probability that word w in document d was generated from topic z *and* that topic z was chosen according to the topic distribution for the location and time of the document, and not the topic distribution of the document. These are calculated in the E-step and used subsequently in the M-step to iteratively estimate the unknown values until convergence. Due to the complexity of the equations, we first comment on the meaning of each equation before displaying it.

E-step

The probability that a word w in a document d was generated by a given topic z is defined by the probability of selecting the thematic model $(1-\lambda_B)$ and the word probability $(p(w|z))$, as well as the probabilities of the topic being selected either by the document's topic distribution $((1-\lambda_{TL})p(z|d))$ or by the timestamp and location's topic distributions $(\lambda_{TL}p(z|t_d,l_d))$. Combined, we get

$$p(z|d,w) = \frac{(1-\lambda_B)p(w|z)[(1-\lambda_{TL})p(z|d)+\lambda_{TL}p(z|t_d,l_d)]}{\lambda_B p(w|z_B)+(1-\lambda_B)\sum_{z'\in Z} p(w|z')[(1-\lambda_{TL})p(z'|d)+\lambda_{TL}p(z'|t_d,l_d)]}.$$

The probability that a word w in a document d was generated by a given topic z that was generated from the timestamp t_d and location's l_d topic distribution is simply the probability of the topic being selected given the timestamp and location of the document (λ_{TL}) multiplied with the probability of selecting that topic $(p(z|t_d,l_d))$. We get

$$p(z,t,l|d,w) = \frac{\lambda_{TL}p(z|t_d,l_d)}{(1-\lambda_{TL})p(z|d)+\lambda_{TL}p(z|t_d,l_d))}.$$

It is initially surprising that the word is not present in the equation, resulting in an equal probability for all the words in the document. This remark is not mentioned in the original paper on PSTA. Most likely, it arises from the fact that if we were to include the probability of the word, we would also have to normalize with this probability to make the distribution sum to one. Inevitably, the word probabilities cancel out, and we are left with the equation as displayed above.

M-step

The unknown topic distribution per document d is calculated as the probability of the topic generating the words of the document ($n(w,d)p(z|d,w)$) and the probability that the topic is not selected based on the timestamp and location ($1 - p(z, t_d, l_d|d,w)$). The equation is

$$p(z|d) = \frac{\sum_{w \in V} n(w,d)p(z|d,w)(1 - p(z,t_d,l_d|d,w))}{\sum_{z' \in Z}\sum_{w \in V} n(w,d)p(z'|d,w)(1 - p(z',t_d,l_d|d,w))},$$

where the word count factor reduces the sum from all the words in the vocabulary to all the words in the document.

Next, the topic distribution per timestamp t_d and location l_d is calculated as a normalized sum over all the documents with the relevant timestamp and location, the probability that the topic generated the words of each document ($n(w,d)p(z|d,w)$) and that the topic was generated based on the location and timestamp ($p(z,t_d,l_d|d,w)$). This is calculated as

$$p(z|t,l) = \frac{\sum_{d \in D: t_d = t, l_d = l}\sum_{w \in V} n(w,d)p(z|d,w)p(z,t,l|d,w)}{\sum_{z' \in Z}\sum_{d \in D: t_d = t, l_d = l}\sum_{w \in V} n(w,d)p(z'|d,w)p(z',t,l|d,w)}.$$

Lastly, the word distribution is simply the probability that the words of a document is generated by the given topic, i.e.,

$$p(w|z) = \frac{\sum_{d \in D} n(w,d)p(z|d,w)}{\sum_{w' \in V}\sum_{d \in D} c(w',d)p(z|d,w')}.$$

The latent and unknown variables are updated accordingly until convergence, resulting in a successful inference of the unknown variables of the mixture model.

4.5.4 Spatiotemporal Analysis

The estimated parameters can be used to find the theme life cycles and theme snapshots of the discovered topics. For a theme and location, the theme life cycle is given by

$$p(t|z,l) = \frac{p(z|t,l)p(t,l)}{\sum_{t' \in T} p(z|t',l)p(t',l)}, \tag{4.18}$$

where $p(t,l) = \frac{n(t,l)}{n()}$ is the word count of all documents within location l and timestamp t divided by the word count for all documents. The theme snapshot is given by

$$p(z,l|t) = \frac{p(z|t,l)p(t,l)}{\sum_{l' \in L}\sum_{z' \in Z} p(z'|t,l')p(t,l')}. \tag{4.19}$$

We can use this information in multiple applications of analysis. Mei et al. provide some examples of use cases [20]. For the PTTP Problem of Definition 2.3.2, we can use the spatiotemporal information in further analysis. The theme life cycles can be used in a periodic pattern algorithm for time series, resulting in periodic patterns per location and theme. Looking at the theme snapshots for consecutive timestamps around the period discovered it should reveal if the themes "move" geographically. Chapter 5 explores the possibilities in PSTA and other previously mentioned algorithms.

4.6 Summary

In this chapter, we presented algorithms for periodic pattern mining, spatiotemporal periodic analysis, periodic topic mining, and spatiotemporal topic mining. The described algorithms were `AUTOPERIOD`, `Periodica`, LPTA, and PSTA, respectively. Next, these algorithms will serve as a basis for expanded algorithms that solve the defined PTTP Problem.

Chapter 5

Mining Periodic Topic Trajectory Patterns

This chapter presents possible methods to solve the Periodic Topic Trajectory Pattern (PTTP) Problem of Definition 2.3.2. The methods presented in this chapter are extensions of the algorithms described in detail in Chapter 4. With the proposed algorithms below, we answer RQ2. The chapter starts with a PSTA-based algorithm in Section 5.1. Next, an LPTA-based algorithm is presented in Section 5.2 and a `Periodica`-based algorithm in Section 5.3. We finalize the chapter with a summary.

5.1 PSTA with Additional Analysis

As mentioned in Section 4.5, Probabilistic Spatiotemporal Topic Analysis (PSTA) already provides the tools to perform periodic topic trajectory analysis by periodically analyzing the theme life cycles defined by Equation 4.18. The setup is similar to that of `Torpedo` by Wang et al. [4], in that we first model the data and then periodically analyze the model using `AUTOPERIOD`. We call PSTA with additional trajectory analysis for `PSTA+`.

5.1.1 Model

The algorithm is executed by first running the PSTA algorithm in its entirety, resulting in theme life cycles for all locations and themes. The theme life cycle is a time series, representing the probability that the given location and theme co-occur for each timestamp. By utilizing a periodic pattern algorithm, e.g., the one described in Section 4.1, we can extract periodicities per location and theme. We combine the discovered patterns with the same period for different locations and sort them by their relative offsets. As a result, we get a trajectory of periodic theme patterns. The pseudocode for the method is presented in Algorithm 3.

We briefly comment on line 6 in Algorithm 3. As `AUTOPERIOD` does not return the relative offset of the periodic patterns discovered, we find these manually. The first peak of the theme life cycle for a given topic and location that fits the discovered period T is set as the offset $\tau_{l,z}$ of the specific location and topic pair. Further, the relative timestamp offset is set as $\tau_{l,z} \mod T$. We discard any pattern with no identified offset and those patterns where the first periodic peak occurs after $1/3$ of the timeline. Although, if we are interested in subpatterns of the timeline, we keep all patterns regardless of offset. The lowest offset value of the locations within the pattern is set as the initial offset, τ, when we store the pattern in line 11.

Algorithm 3: PSTA+.

Input: Dataset $D = \{d_1, d_2, ..., d_n\}$.
Result: A set of PTTPs.

```
Mine theme life cycles using the PSTA algorithm, see Algorithm 3.
for each z ∈ Z do
    for each l ∈ L do
        cycle = {p(t|z,l)}_{t∈T}.
        Mine periodic patterns in cycle using AUTOPERIOD.
        Find offsets from the peaks of the theme life cycles.
        Store pattern ⟨T, τ, φ_z, l⟩ in P_T.
    for each discovered period T do
        Sort the patterns in P_T by τ.
        Define the trajectory δ = {(P.l, P.τ)}_{P∈P_T}.
        Store the periodic pattern ⟨T, τ_0, φ_z, δ⟩ in P_PTTP.
return P_PTTP.
```

5.1.2 Complexity Analysis

The time complexity of PSTA is $O(Knu + K|L||\mathcal{T}||V|)$ per iteration of the EM algorithm [20]. Recall that K is the number of topics and n the number of documents. Further, u represents the average number of unique words in a document, capped at 70 words for Twitter data consisting of 140 characters. Lastly, $|\mathcal{T}|$, $|L|$, and $|V|$ are the number of timestamps, locations, and unique terms in the dataset, respectively. Moreover, the time complexities of the additional analysis steps are

- $O(K|L||\mathcal{T}|)$ for defining the theme life cycle.
- $O(K|L||\mathcal{T}| \log |\mathcal{T}|)$ for the AUTOPERIOD implementation using DFT.
- $O(\tilde{p}K|L||\mathcal{T}|)$ for finding the offsets of each location and topic combination, where $\tilde{p}$ is the average number of periods discovered per location and topic.
- $O(\tilde{T}K|L|^2)$ for sorting the location trajectory, where $\tilde{T}$ is the average number of discovered periods. In most cases, the number of locations in the pattern trajectory should be significantly less than $|L|$, reducing the quadratic factor to linear.

Combined, the analysis phase has a time complexity of $O(K|L|(|\mathcal{T}| + |\mathcal{T}| \log |\mathcal{T}| + \tilde{p}|\mathcal{T}| + \tilde{T}|L|)) = O(K|L||\mathcal{T}| \log |\mathcal{T}| + K\tilde{T}|L|^2)$. Because of the number of iterations of the EM algorithm is unknown, we define the variable *iter* as the number of iterations, and conclude that the time complexity of PSTA+ is $O(iter * Knu + iter * K|L||\mathcal{T}||V| + K|L||\mathcal{T}| \log |\mathcal{T}| + K\tilde{T}|L|^2) = O(iter * Knu + iter * K|L||\mathcal{T}||V|)$.

The space complexity of the algorithm is $O(Knu + Kn + K|V| + Kn + K|\mathcal{T}||L| + |\mathcal{T}|) = O(Knu + K|V| + K|\mathcal{T}||L|)$. The six initial addends of the expression are the space requirements for respectively the two latent variables, $p(z|d, w)$ and $p(z, t, l|d, w)$, the three unknown variables, $p(w|z)$, $p(z|d)$ and $p(z|t, l)$ and lastly the theme life cycle, $p(t|z, l)$.

To speed up the algorithm, it is also possible to store statistics about the dataset to be used in the EM algorithm. This will increase the space requirements and decrease the

runtime. Consequently, both the system specifications and user preferences should be considered when deciding this.

5.2 LPTA with Geographical Information

Section 4.4 describes the Latent Periodic Topic Analysis (LPTA) algorithm. It incorporates periodic time into a topic model to find recurring topics in a set of timestamped documents. Given this model, it should be possible to define a mixture model that extends LPTA with a geographical dimension so that it can find periodic geographical topics. The time component needs to be connected to the location, so the time, location, and topic are joint periodic. We call such a model `GeoLPTA`.

5.2.1 Model

Given the LPTA model, we introduce a model that infers the time given the latent topic *and* the known location of the document. The mixture model becomes

$$p(w,t|\Psi) = \sum_{z \in Z} p(z|d)p(w|z)p(t|z,l_d)$$

for the same unknown variables $\Psi = \{\varphi, \theta, \mu, \sigma\}$ representing word distributions per topic, topic distributions per document and mean and standard deviations for bursty and periodic topics, respectively. As bursty topics are out of scope for this thesis, we remove the ability to discover bursty topics in `GeoLPTA` so the runtime will compare better to the other proposed models.

The timestamp is extended from LPTA to be dependent on location in addition to the topic, propagating to the μ and σ values, so they represent the collections of the means and standard deviations of timestamps for periodic topics *per location*. More specifically, μ_{z,l_d} and σ_{x,l_d} are respectively the mean and standard deviation of timestamps for topic z at location l_d.

The mixture model can be solved identically as the original LPTA algorithm, detailed in Section 4.4, with the extension that the means and standard deviations are defined per location. The change is implemented by iterating over all documents within the current location instead of all documents when calculating the means and standard deviations in Equations 4.16 and 4.17, i.e. $\sum_{d \in D}$ is replaced with $\sum_{d \in D_l}$ where D_l is the collection of all $d_i \in D$ where $loc_i \in l$. This propagates to the time distributions of Equations 4.7 and 4.9. We present the resulting equations for the EM algorithm in `GeoLPTA` and bold the changes from LPTA.

E-step

$$p(z|d,w) = \frac{p(t_d|z,\mathbf{l_d})p(w|z)p(z|d)}{\sum_{z'} p(t_d|z',\mathbf{l_d})p(w|z')p(z'|d)}.$$

M-step

$$p(w|z) = \frac{\sum_d n(d,w)p(z|d,w)}{\sum_d \sum'_w n(d,w')p(z|d,w')},$$

$$p(z|d) = \frac{\sum_w n(d,w)p(z|d,w)}{\sum_w \sum'_z n(d,w)p(z'|d,w)}.$$

For the time distributions, we first update μ and σ for the periodic topics:

$$\boldsymbol{\mu}_{z,l} = \frac{\sum_{d \in D_l} \sum_w n(d,w) p(z|d,w)(t_d - I_d T)}{\sum_{d \in D_l} \sum_w n(d,w) p(z|d,w)},$$

$$\boldsymbol{\sigma}_{z,l} = \sqrt{\frac{\sum_{d \in D_l} \sum_w n(d,w) p(z|d,w)(t_d - \boldsymbol{\mu}_{z,l} - I_d T)^2}{\sum_{d \in D_l} \sum_w n(d,w) p(z|d,w)}}.$$

Then we update $p(t|z,l)$ with the new means and standard deviations:

Periodic:

$$p(t|z,\mathbf{1}) = p(t|z,\mathbf{1},k)p(k) = \frac{1}{\sqrt{2\pi}\boldsymbol{\sigma}_{z,l}} \exp\left(-\frac{(t - \boldsymbol{\mu}_{z,l} - kT)^2}{\boldsymbol{\sigma}_{z,l}^2}\right) * p(k).$$

Background (fixed):

$$p(t|z,\mathbf{1}) = \frac{1}{t_{end} - t_{start}}.$$

At convergence, we have an inferred time distribution for periodic topics per location. However, we are interested in how each periodic topic *moves* geographically. A simple approach would be to sort the locations by their mean value for each topic and create a trajectory of the sorted locations. However, this assumes the pattern is apparent in all locations, which is unlikely.

A filtering step is necessary to filter out those locations that show no signs of periodicity. If the standard deviation converges to 0, we get an undefined distribution, i.e. $\lim_{\sigma_{z,l} \longrightarrow 0} p(t|z,l) = \infty$. When this happens, we can flag the location as non-periodic. Further, the means and standard deviations will, in some cases, divide by 0 if the latent variable is 0 for all the words in the document. If this happens, we also flag the location as non-periodic.

The full algorithm for `GeoLPTA` is presented in Algorithm 4.

Algorithm 4: `GeoLPTA`.

Input: Periods $\{T1, T2...\}$, Dataset $D = \{d_1, d_2, ..., d_n\}$.
Result: A set of PTTPs.

```
Perform EM algorithm on latent and unknown variables until convergence.
for each z in Z do
    Define movement δ_z.
    for each l in L do
        if not Flagged(z,l) then
            Add l to δ_z.
    Sort δ_z by μ_{z,l}.
    Store the periodic pattern ⟨T, τ_0, φ_z, δ_z⟩ in P_PTTP.
return P_PTTP.
```

5.2.2 Complexity Analysis

The time complexity of LPTA is $O(iter * K|W|)$, where *iter* is like in PSTA+ the number of iterations of the EM algorithm, and $|W|$ is defined as the count of all the words in all the documents [3]. The equation can be rewritten as $O(iter * Knu)$ for comparability, where K is the number of topics, n is the number of documents, and u is the average number of words per document. The additional analysis step in GeoLPTA has a time complexity of $O(K|L|)$. As $n \geq |L|$ and $u > 1$, the overall time complexity becomes $O(iter * Knu)$.

The space complexity is dominated by the storage of the distributions in the EM algorithm. For the latent variable, the space complexity is $O(Knu)$. The unknown variables have space complexities $O(K|V|)$ for the topics, $O(Kn)$ for the topic distribution per document, and $O(K|L||\mathcal{T}| + 2(K-1)|L|)$ for the topic and location time distributions. Recall that the topic and location time distribution also includes mean and standard deviation values for all periodic topic and location pairs. The overall space complexity for GeoLPTA is $O(Knu + K|V| + K|L||\mathcal{T}|)$ for $|\mathcal{T}| > 2$.

Lastly, just like PSTA+, it is possible to store additional statistics in exchange for a decreased runtime.

5.3 Periodica with Topics

The spatiotemporal periodic pattern mining algorithm Periodica, presented in Section 4.2, mines periodic behaviors in geographical trajectory data. We extend this model by adding a topic analysis step and subsequently perform each step of Periodica per discovered topic. We call the extended model TopicPeriodica.

5.3.1 Model

By expanding the Periodica algorithm with a topic model, each reference spot is analyzed per topic. In this manner, we can mine recurring geographic trajectory topics. The pseudocode for the algorithm is presented in Algorithm 5. In the following paragraphs, we discuss how the algorithm differs from Periodica.

The expanded model follows the general logic of Periodica, with some minor additions or alterations. After finding the reference spots in the first stage of the algorithm, the documents are segmented into regional clustered datasets (see Definition 2.1.5) according to their timestamp and reference spot. They are subsequently analyzed using a topic model, e.g., LDA. In this topic model, each $D_{j,k}$ is treated as a single document. Consequently, we get a topic distribution per reference spot and timestamp combination. Next, for each reference spot discovered from the geographical data and for each topic discovered in the topic model, we replace the binarization method with a method that handles the topic distribution values instead of just the presence or non-presence of a moving object. The value in position i in the new vector indicates the probability of the current topic in the given reference spot for timestamps $t_i \in \mathcal{T}$. The presence of a topic is set to 0 if the probability is less than some threshold value ϵ. The vectors are input to the AUTOPERIOD algorithm described in Section 4.1 to detect cycles.

Given the information about periods for each reference spot and topic combination, we wish to create a movement trajectory. Again, we introduce some slight changes to the original algorithm. Firstly, we need to iterate for each period detected *and* each

Algorithm 5: `TopicPeriodica`.

Input: Dataset $D = \{d_1, d_2, ..., d_n\}$.
Result: A set of PTTPs.

```
/* Stage 1: Detect periods.                                    */
```
1 Find reference spots $O = \{o_1, o_2, ..., o_d\}$.
2 Find topics Z and topic distributions $\theta_{D_{k,j}}$ for all $t_k \in \mathcal{T}$ and $o_j \in O$.
3 **for** *each* $o_i \in O$ **do**
4 **for** *each* $z_j \in Z$ **do**
5 Detect periods in o_i for topic z_j and store the periods in $P_{i,j}$.
6 $P_{set} \leftarrow P_{set} \cup P_{i,j}$.

```
/* Stage 2: Mine periodic behaviors.                           */
```
7 **for** *each* $z_j \in Z$ **do**
8 **for** *each* $T \in P_{set}$ **do**
9 $O_T = \{o_i | T \in P_{i,j}\}$.
10 Construct the symbolized sequence $S = \{s_1, ..., s_n\}$ using O_T where s_k is a list of all $o_i \in O_T$ where $\theta_{D_{k,i}}(z_j) > \epsilon$.
11 Mine periodic behaviors in S, see Algorithm 2. Store them in $\mathbf{P}_{\text{PTTP}}$.
12 **return** $\mathbf{P}_{\text{PTTP}}$.

topic from the topic model. Further, we map our data per period and subject to store *all* the reference spots where the topic probability is above some threshold for every timestamp.

This change breaks the notion that the sum over all topics per timestamp in the distribution matrix should be equal to one, as the initial value is the relative frequency as defined by Equation 4.2. Logically, it does not sum to one as we no longer have a physical object that can only be at one place at a time. However, for each merge of clusters, we update the distribution matrix according to Equation 4.4, which normalizes each value according to the number of segments in each cluster. The normalization results in the distribution summing to one even though the probability might be higher as a topic can be present in multiple locations simultaneously. We keep this in mind as we read the results.

Next, the reference spots per topic are used to mine periodic behaviors. We mine behaviors per topic per period across reference spots using the same logic as `Periodica`. However, a small change is implemented in the clustering of the segments to reduce the number of false patterns. While `Periodica` clusters any segment together, we require that the clustered segments must be adjacent. Logically, if a pattern with period T only reoccurs every second segment, the real period is, in fact, $2T$. If it occurs randomly in the segments, it is not periodic at all.

5.3.2 Complexity Analysis

The time complexity of `TopicPeriodica` is the sum of the time complexities of the distinct steps of the algorithm. The time complexity of the detection of reference spots is $O(whn)$, where w and h are respectively the width and height of the space grid, and n is the number of documents. The time complexity of LDA depends on the implementation and hyperparameters. For a small number of topics, as we have, it is shown to be in polynomial time [33]. We define $O(LDA_t)$ as the time complexity of

LDA. Further, the AUTOPERIOD algorithm has a time complexity of $O(Kd|\mathcal{T}|\log|\mathcal{T}|)$. Recall that K is the number of topics, d is the number of reference spots, and $|\mathcal{T}|$ is the number of discrete timestamps. Further, we calculate the symbolized sequence for each topic and period in that topic and then mine periodic behaviors. This step gives a time complexity of $O(Kp\bar{d}|\mathcal{T}|)$, where p is the average number of periods discovered per location and topic combination and $\bar{d}$ is the average number of detected reference spots per period.

The algorithm for mining periodic behaviors from the symbolized sequence is $O(mdT + m(m\log m + 2dT)) = O(m^2\log m + 2mdT)$, where m is the number of iterations which in the worst-case scenario it is the number of initial segments and T is the period. The first addend of the sum is the time complexity of the initial calculation of the distance between the different behaviors. The second is the calculations per iteration, which includes the fetching of the minimum pair to merge and the computation of the newly merged clusters. This calculation differs slightly from the original clustering algorithm's time complexity, as we have included a restriction that the merging clusters need to be adjacent. Overall, the time complexity of TopicPeriodica is $O(whn + LDA_t + Kd|\mathcal{T}|\log|\mathcal{T}| + Kp\bar{d}|\mathcal{T}| + m^2\log m + 2mdT)$. Note that whn and possibly LDA_t will dominate the runtimes, unless for sparse datasets where $n \simeq |\mathcal{T}|$.

The space complexity when calculating the reference spots is $O(wh)$. LDA is again dependent on the implementation, and we define the space complexity as $O(LDA_s)$. AUTOPERIOD has a space complexity of $O(|\mathcal{T}|)$, and the symbolized sequence of $O(p|\mathcal{T}|)$. Lastly, the space complexity for mining periodic behaviors is $O(mpdT|\mathcal{T}|)$. The total space complexity is $O(wh + LDA_s + |\mathcal{T}| + p|\mathcal{T}| + mpdT|\mathcal{T}|) = O(wh + LDA_s + mpdT|\mathcal{T}|)$.

5.4 Summary

In this chapter, three models were presented to solve the defined PTTP Problem. We consider RQ2 consequently answered. The models introduced are PSTA+, GeoLPTA and TopicPeriodica. To evaluate each algorithm, they were implemented and executed with different datasets. The methodology of these experiments is presented next.

Chapter 6

Evaluation Methodology

The proposed models of Chapter 5 are evaluated and analyzed by reviewing the returned patterns and by comparing their runtimes. This chapter provides details on how we conduct the thesis experiments by implementing and executing each algorithm. We start by presenting the experiment's datasets in Section 6.1, before moving on to how we preprocess these datasets in Section 6.2. Further, we present the performance evaluation strategy in Section 6.3. The actual implementation details are presented in Section 6.4 before we provide details on the environment in which the algorithms are executed and timed in Section 6.5. Lastly, we discuss how we estimate and select the variable parameters of each algorithm in Section 6.6. We briefly summarize the chapter in Section 6.7.

6.1 Datasets

We utilize three datasets consisting of geotagged tweets. The first dataset is synthetically generated to test the algorithms' ability to mine the correct periodic patterns. The second and third datasets are larger real datasets selected to test the scalability of each algorithm. We next present the three datasets.

6.1.1 Synthetic Dataset

The synthetic dataset is designed to contain two specified patterns that the algorithms should be able to find. From a preprocessed Twitter dataset of 4000 tweets from 247 different locations, we insert two periodic patterns, summarized in Table 6.1. An illustration of the same patterns is presented in Figure 6.1. The same tweet defines each topic and is inserted in periodic positions in the dataset. Topic 1 is a topic on Italian food, and Topic 2 is a topic about football. Respectively, the two tweets that define each topic are

Italian food: "Parmesan is the best Italian food ingredient. My food life consists of Italian parmesan, tomatoes, beef, and parmesan. Lets eat all the food!"

Football: "Football is my life. Kick a ball in the goal; the goal is to win, win, win!! Sports! Football is a sports :) Lets kick"

Additionally, we set new timestamps for all the tweets to create an evenly sampled dataset, with eight tweets per day for 500 days. The result is 4000 tweets from January 1, 2018, 12:00 AM till May 15, 2019, 9:00 PM. Every seven days from day 0, two of the eight tweets of that day is the Italian food-tweet, and every 15 days starting at day 4, two of the eight tweets of that day is the football-tweet. The rest of the tweets are random tweets from the original dataset.

ID	Topic	Period (days)	Offset (days)	Location Trajectory (offset: location)
1	Italian food	7	0	**0:** Puerto Carreno, Vichada, CO **1:** Neustadt, Hamburg, DE
2	Football	15	4	**4:** Cabanbanan, Calabarzon, PH **6:** Olavarria, Buenos Aires, AR **7:** Las Pinas, Calabarzon, PH **11:** Scarborough, Ontario, CA

TABLE 6.1: Summary of inserted patterns in the synthetic dataset.

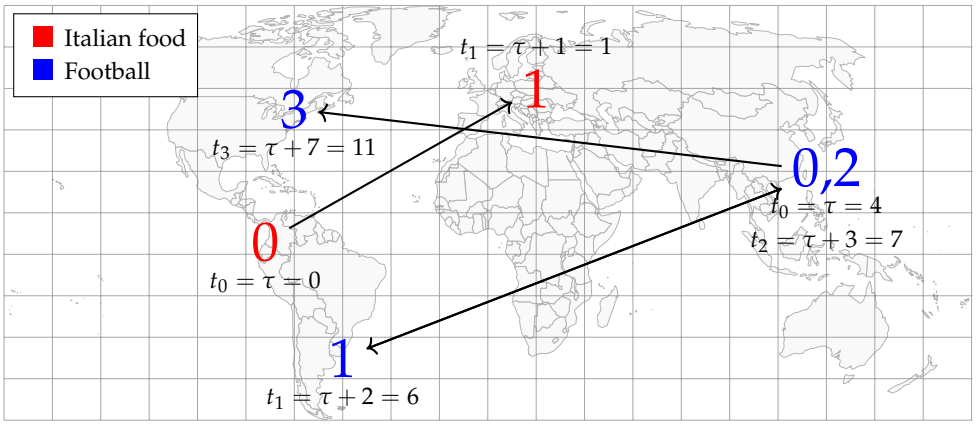


FIGURE 6.1: Illustration of the trajectory of the inserted periodic patterns in the synthetic dataset.

We check for inherent patterns in the dataset by applying each algorithm to the dataset without the inserted patterns and checking the results. If patterns are output, we check if they are consistently output, indicating that the pattern is actual. We find one such pattern when applying the `TopicPeriodica` algorithm to the pre-inserted dataset. The pattern is about employment and job careers, has a period of 19 days, and moves around in Europe. Likewise, the same pattern occurs in `GeoLPTA` for numerous different periods. This pattern will likely reoccur when we execute `TopicPeriodica` on the post-inserted dataset, and possibly the other algorithms as well.

6.1.2 Real Datasets

The second and third datasets consist of significantly more tweets than the synthetic dataset. They are both original datasets from the Twitter API, and we make sure they contain at least one tweet per day and are limited to the US. We set the location granularity at the state level, resulting in 50 different locations (one for each state in the US). The second dataset contains tweets from October 3, 2015, until April 11, 2016, and the third from May 11, 2016, until June 1, 2016. While the second dataset contains about 85 000 tweets over 161 days, the third dataset consists of 22 million tweets over just 22 days.

We use the second dataset to test the initial scalability of the algorithms and the third for further research on the initially scalable algorithms. We extract subsets to

Subset proportion	Size (MiB)	#Tweets	#Locations	#Timestamps (days)	#Words
0.01	0.082	852	38	2	2672
0.1	0.82	8525	49	12	12 389
0.2	1.64	17 051	50	20	18 875
0.5	4.1	42 628	50	46	31 290
0.7	5.74	59 679	50	61	37 465
1.0	8.2	85 257	50	192	45 448

TABLE 6.2: Subsets of the second dataset.

Subset proportion	Size (GiB)	#Tweets	#Locations	#Timestamps (hours)	#Words
0.01	0.022	220 881	50	6	76 475
0.1	0.22	2 208 813	50	49	344 781
0.2	0.44	4 417 627	50	100	558 807
0.5	1.1	11 044 069	51	246	1 020 043
0.7	1.54	15 461 696	51	344	1 259 534
1.0	2.2	22 088 138	51	498	1 559 893

TABLE 6.3: Subsets of the third dataset.

measure how the algorithms scale with the input size. The characteristics of the second dataset and its subsets are summarized in Table 6.2, and the third dataset is summarized in Table 6.3. In all the experiments using subsets of the second and third datasets, we set the number of topics to two ($K = 2$).

6.2 Preprocessing

We preprocess each dataset in a Python program utilizing the Pandas[1] library for fast data processing. As Twitter data contains few terms in a colloquial language, often with abbreviations and spelling errors, we must take numerous steps to improve the text quality of the tweets [34]. The topic models utilized in our algorithms are language-dependent. As Twitter data includes an inferred language component, it is easy to extract all the inferred English tweets. Further, we utilize the *tweet-preprocessor*[2] library to clean the text of URLs, mentions, emojis, smileys and numbers. We keep the hashtags as they often emphasize the theme of the tweet. Further, the *gensim*[3] library is utilized to remove stopwords.

Next, we lowercase the text and clear excessive white space. The colloquial language of Twitter data results in multiple tweets containing words with the same letter repeated numerous times, e.g., spelling "good" as "gooooooood". This example results in two separate words in the vocabulary. We implement a simple find-and-replace approach of removing excessive letters, which overall results in a highly improved and reduced vocabulary. Further, a reliable statistical categorization relies on a certain amount of data for each term. Therefore, terms only mentioned once combined

[1] https://pandas.pydata.org/
[2] https://github.com/s/preprocessor
[3] https://radimrehurek.com/gensim/

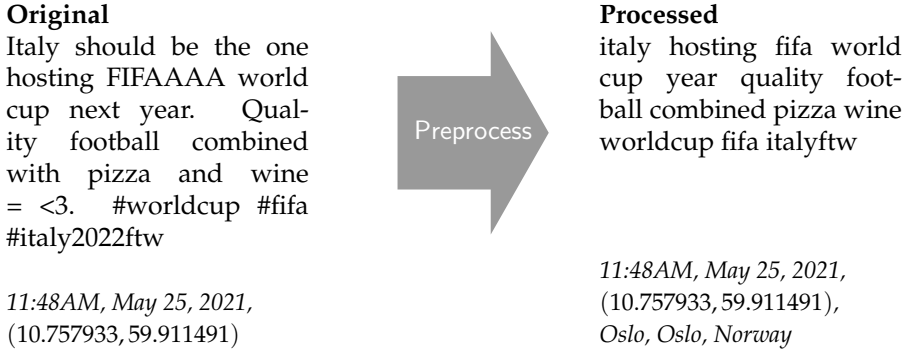


FIGURE 6.2: Transformation of a tweet before and after preprocessing.

in all the documents are removed, which more than halves the vocabulary. Lastly, we eliminate all words with less than three characters and all tweets with less than three words. Other preprocessing techniques like stemming and lemmatization are not applied as this actually decrease sentiment classification accuracy for Twitter data [34].

In addition to text preprocessing, supplementary geographical information is valuable for the topic modeling-based methods that rely on categorizing the locations. Therefore, we employ the *reverse-geocoder*[4] library to extract name, administrative region, and country-code for each longitude and latitude value in the dataset. An example illustrating the preprocessing is presented in Figure 6.2.

6.3 Performance Measures

The qualitative performance is measured by analyzing the returned patterns of each algorithm. As initial values affect the results, the algorithm is executed ten times, and a representative result is selected to be presented. If there are great differences between the executions, this will be commented on.

The quantitative performance is measured by the runtime for different dataset sizes, defined by Tables 6.2 and 6.3, and for $2 - 10$ topics. When we vary the number of topics, we use the full second dataset. Additionally, we note the maximum space usage per algorithm for the largest dataset proportions.

Each algorithm's runtime is measured as the wall clock time from the algorithm's initialization to its completion. It excludes the reading of the input data and the writing of the results to file. We divide the runtime into three distinct phases: the initialization, the execution, and the additional analysis. The initialization phase calculates possible statistics and metadata to speed up the execution phase. The principal part of the algorithm is the execution phase, and the analysis phase contains additional analysis after EM convergence for the topic modeling-based algorithms. Note that `TopicPeriodica` does not have an initialization nor analysis phase, which will result in a negligible runtime for these phases.

We execute each algorithm 11 times per unique input. We note that the first run always has a higher runtime than the subsequent ten iterations. This is likely due

[4] https://github.com/thampiman/reverse-geocoder

to cache operations and Just-In-Time (JIT)[5] compilation in Java. Consequently, we exclude the runtime of the first execution to mitigate any external factors. Moreover, we reset the algorithm after each iteration. We measure the runtime in nanoseconds from the algorithm's initialization to the analysis function returns. After 11 iterations, we average the last ten iterations and return an average runtime. We present an overview of the evaluation process in Algorithm 6.

Algorithm 6: Benchmarking.

Input: Input pathname p, number of topics K.
Result: The average runtime of the algorithm.

```
for i ∈ {1, ..., 11} do
    Load dataset.
    t1 ← wall clock time.
    Initialize(p).
    t2 ← wall clock time.
    Execute(K).
    t3 ← wall clock time.
    Analyze().
    t4 ← wall clock time.
    if i ≠ 1 then
        I ← I ∪ {t2 − t1}.
        E ← E ∪ {t3 − t2}.
        A ← A ∪ {t4 − t3}.
        Write patterns to file.
        Log peak memory usage.
    Reset().
return Avg(I), Avg(E), Avg(A).
```

6.4 Implementation

The implementation of the preprocessing and the algorithms are publicly available[6] as of the date of publication of this thesis. All algorithms and helper functions are serially implemented in Java[7], except for the text preprocessing (see Section 6.2). We chose Java as the main programming language as it makes the algorithms' time consumption more transparent. To model topics in `TopicPeriodica`, we used an LDA implementation by MALLET (MAchine Learning for LanguagE Toolkit)[8]. The `AUTOPERIOD` algorithm by Vlachos et al. [1] used in `TopicPeriodica` and `PSTA+` was implemented as well, utilizing the ezFFTW[9] library for the DFT calculations. The library is a Java wrapper for the C-implemented FFTW (Fastest Fourier Transform in the West)[10]. Lastly, to decrease the runtimes of the algorithms, we stored additional information about the dataset in the initialization phase of the topic modeling-based algorithms.

[5] https://www.ibm.com/docs/en/sdk-java-technology/8?topic=reference-jit-compiler
[6] https://github.com/liseph/masterproject
[7] https://www.java.com/en/
[8] http://mallet.cs.umass.edu/
[9] https://github.com/hageldave/ezFFTW
[10] http://fftw.org/

6.5 Environment

Runtime measurements were done on a P2P-based database system for data-intensive applications called DASCOSA [35][11] . It is a system with 40 Intel Xeon Silver 3210 processors and four memory disks of 3.7TB. Each processor has ten cores with 20 threads each and a total cache size of 13.75 MB. The operating system running beneath is Ubuntu 18.04.5 LTS.

6.6 Parameter Estimation

For the three proposed models, several user-input parameters need to be set by the user or program. This section discusses how we set these parameters in our experiments and how each parameter influences the results. Finally, we discuss the time and location granularity, which is common for all three algorithms.

6.6.1 PSTA+

All the algorithms need to set the number of topics. Additionally, PSTA+ uses discrimination values to extract the background topic and decide the importance of the time and location when selecting a topic. Further, as the algorithm applies the EM algorithm, both the initial values and the limit as to what constitutes convergence will significantly affect the runtime and the result of the algorithm. We discuss how to set these values.

Number of topics: The number of topics impacts both the runtime and the memory usage. Empirically, the higher the number of topics, the probability of finding false topics and patterns increases. For this reason, we recommend setting a small number of topics initially and increasing it if the resulting topics discovered seem to be merged topics. If nothing else is specified, we use $K = 2$.

λ_B and λ_{TL}: λ_B decides how discriminative the extracted themes will be, and λ_{TL} decides how vital the temporal and geographical information is when selecting a topic for each word in the generative process. They are set by Mei et al. empirically to respectively between 0.9 and 0.95, and between 0.5 and 0.7 [20]. We find that the results do not vary significantly for different values within these intervals, and so we set λ_B to 0.9 and λ_{TL} to 0.5.

Initial values of distributions: We find that the initial values of the distributions contribute significantly to whether the algorithm finds the correct topics. We set the initial values of the topics to a uniform distribution, removing any initial bias of the resulting topics. The rest of the latent and unknown variables are set to random distributions. We recommend running the algorithm multiple times with a different seed to alleviate the skewness of the initial values for the most accurate results.

Convergence limit: In the EM algorithm, we calculate the difference between the old value and the new value for each iteration and check whether this difference is less than the convergence limit. We find empirically that a value lower than 0.01 does not change the outcome, while a higher value stops the algorithm too soon in some cases. For this reason, we set the convergence limit to 0.01.

[11] https://research.idi.ntnu.no/dascosa/

6.6.2 GeoLPTA

In addition to the number of topics, we also set the periods to discover. Further, similarly to PSTA+, we need to set initial values of the distribution and a convergence limit. We also include a standard deviation minimum value, which is necessary to avoid illegal divisions on zero. These values are discussed and set in the following.

Number of topics: The number of topics is also the number of patterns for GeoLPTA, affecting the results and runtime. We find that for a high number of topics, the algorithm performs poorly and is not able to detect correct topics. Therefore, we keep the number of topics low and increase it if necessary. If nothing else is specified, we use $K = 2$.

Periods: In addition to the number of topics, GeoLPTA requires the user to input the actual periods. As discussed in Section 4.4, the authors of LPTA suggest utilizing Schwarz's Bayesian information criterion (BIC) to set the periods (and the number of topics). We do not utilize this method, but it is possible for more accurate results. In our experiments, we set the periods based on domain knowledge of the dataset.

Initial values of distributions: Also for GeoLPTA, we find that the initial values of the distributions contribute significantly to whether the algorithm finds the correct patterns or not. Setting each variable to a random distribution as the initial value provides accurate results in about 50% of the runs and partly incorrect results in the other half. We discover that setting each variable to a uniform distribution provides the correct patterns every time, with slightly less accurate mean values. Therefore, if one wishes high accuracy, we recommend executing the algorithm multiple times with random initial distributions. However, for faster results, a uniform distribution suffices, which we use in our experiments.

Convergence limit: The convergence limit is used in the same way as for PSTA+. We set it to the same value as in PSTA+, i.e., 0.01.

Standard deviation minimum value: If the standard deviation goes to 0, the time distribution goes to infinity due to zero divisions. For this reason, we define a threshold so that if the standard deviation is lower than this threshold, we manually set the time distribution to a uniform distribution. Empirically we find that a threshold of 0.01 is sufficient.

6.6.3 TopicPeriodica

Again we first set the number of topics. Additionally, we need to set the LDA hyperparameters. Further, to reduce the number of false patterns, the algorithm includes a topic presence threshold, so all values below this threshold are set to 0. We also set the smoothing parameter λ. Lastly, the algorithm needs to know when to stop merging clusters, decided by a representation error threshold set manually. The following list discusses and sets the aforementioned variables.

Number of topics: TopicPeriodica uses LDA to extract topics from the document collection, and there is no obvious answer to how many topics we should set. Empirically, the runtime is not greatly affected by the number of topics, and we find that a value of around ten topics provides accurate results. For the most accurate results, there are multiple research papers on determining the

best number of topics for an LDA topic model [36–38], but this is outside the scope of this thesis.

α and β: LDA needs α and β values for the Dirichlet distributions. The values should be between zero and one to ensure sparse distributions. We set them to 0.1.

Number of LDA iterations: The number of LDA iterations affects the accuracy and runtime of the algorithm. From the documentation of MALLET, the value should be between 1000 and 2000 iterations[12]. We set the value to 1500 as we find that a higher value does not change the outcome, while a lower value gives less accurate topics.

Topic presence: We need to decide how large a topic probability needs to be to count as present. The threshold should depend on domain knowledge about the dataset or the user's preferences to how sensitive the algorithm should be. We set the threshold empirically for our datasets to 0.4.

Smoothing parameter λ: This variable is used when we smooth the categorical random variable probability in the Kullback-Leibler divergence of Equation 4.3. The value should be small, and we find that it is not critical what the exact value is. We set it to 0.1.

Representation error limit: If the representation error makes a sudden jump, we stop merging clusters in the final phase of the algorithm. To find an appropriate limit, we track how the representation error changes and find that a value of 0.2 is representative of a significant "jump".

6.6.4 Deciding Time and Location Granularity

All algorithms require a discrete timeline, a location set, or both. We accordingly need to set the granularity for these two sets. For locations, different options are by city, country, or coordinate grids. The choice depends on what kind of patterns we are looking for and how the dataset is sampled. We divide the locations into administrative regions for both the synthetic and the real datasets. `TopicPeriodica` includes a clustering of the data points in the algorithm and subsequently does not use the predefined location set.

To model periodic behaviors, it is useful to have evenly sampled data. Tweets are not evenly sampled and can vary in their frequency. To fit Twitter data with periodic pattern algorithms, we define a set of equally spaced timestamps $\mathcal{T}$ of Definition 2.1.3. The time granularity decides the distance, A, between each timestamp in $\mathcal{T}$. Obvious choices for A are one hour, 24 hours, one week, and one month. A fine granularity will result in analyzing the data in full detail and possibly discovering more patterns. In contrast, a coarse granularity will result in fewer operations on the datasets, but some patterns might be missed.

The time granularity can be set empirically or based on domain knowledge. Moreover, the algorithms expect at least one tweet per timestamp, so we need to set a low granularity for sparse datasets for the algorithms to function. For a considerable data collection, space and runtime restrictions might also impact the choice of time granularity. A possible solution is to choose a low granularity, e.g., one week, and then rerun the algorithm with a higher granularity with a reduced dataset based on

[12] `http://mallet.cs.umass.edu/topics-devel.php`

the discovered patterns. In general, we use $A = 24h$, i.e., we cluster the documents by date if nothing else is specified. Due to the short time span of the third dataset, we set the time granularity for this dataset to 1 hour, i.e., $A = 1h$. We reserve automatic timeline and location clustering based on the properties of each dataset for future work.

6.7 Summary

This chapter presented the methodology used in the experiments of the thesis. We summarized the three datasets used in the experiments and the preprocessing applied to these datasets. Further, we explained how we measured the performance of each algorithm. We summarized language implementation details and the environment in which the algorithms were executed. Lastly, we informed on what we set as hyperparameters and how they possibly affect each algorithm.

Chapter 7

Experimental Results and Evaluations

This chapter presents and evaluates the experimental results. We divide the evaluation into two sections. First, Section 7.1 presents the qualitative results, consisting of an analysis of how accurate each algorithm is in finding Periodic Topic Trajectory Patterns (PTTP). Second, we present runtime measurements of the algorithms and evaluate their quantitative performance in Section 7.2. Finally, an overall evaluation of the experimental results is conducted in Section 7.3. Combined, these results provide the performance of each algorithm, answering RQ3. We conclude the chapter with a summary.

7.1 Qualitative Results and Evaluations

We first qualitatively evaluate how each of the algorithms achieves the goal of mining PTTPs in a Twitter dataset. Moreover, the algorithms should not discover any false patterns. We do this by presenting the returned PTTPs for the three algorithms `PSTA+`, `GeoLPTA` and `TopicPeriodica` using the synthetic dataset, and a summary of the patterns discovered using the second dataset. Additionally, each result includes a discussion on what it says about the performance of the relevant algorithm. Lastly, we qualitatively compare the three algorithms.

7.1.1 Patterns Discovered by PSTA+

We execute two instances of `PSTA+`. The first instance has the number of topics set to 2 ($K = 2$), and the second has the number of topics set to 5 ($K = 5$). In this way, we test how the algorithm performs for the correct number of topics and too many topics. Additionally, we analyze whether the algorithm returns non-existent PTTPs.

For relevance, we only display the correctly identified patterns in Table 7.1 for $K = 2$. We omit the results of the second execution ($K = 5$) as the relevant patterns are identical to those of Table 7.1, with slightly higher probability values in the relevant terms. In addition to the patterns presented, the algorithm returns 15 incorrect patterns for $K = 2$ and 18 incorrect patterns for $K = 5$. These PTTPs are categorized as false patterns, as they are not consistently output when inputting the synthetic dataset to the three algorithms without the inserted patterns. The complete results of the algorithm are displayed in Appendix A.1. In general, the large number of returned patterns indicates that `PSTA+` returns too many patterns in most runs.

Topic (term: probability)	**Period (days)**	**Initial offset (days)**	**Movement (offset: location)**
food: 0.1108, parmesan: 0.1104, italian: 0.0738, eat: 0.0370, ingredient: 0.0368, beef: 0.0368, tomatoes: 0.0365, best: 0.0330, orange: 0.0010, accounting: 0.0010	7.0	0.0	0.0: Vichada, CO, 1.0: Hamburg, DE
win: 0.1045, football: 0.0699, sports: 0.0699, goal: 0.0691, kick: 0.0691, ball: 0.0349, driver: 0.0030, transportation: 0.0025, truck: 0.0022, place: 0.0019	3.0	4.0	1.0: Calabarzon, PH
	15.0	4.0	4.0: Calabarzon, PH, 6.0: Buenos Aires, AR, 11.0: Ontario, CA

TABLE 7.1: A subset of the returned PTTPs of PSTA+ for $K = 2$.

From the results of Table 7.2, we conclude that the algorithm successfully finds the two inserted topics. The topics themselves are displayed as their top ten terms, and we see that relevant terms dominate these. Moreover, each pattern has the correct period and almost complete movement. We say *almost* complete because of the two patterns identified for the football topic in Table 7.1. Recall that the inserted football pattern occurs in the Philippines at two different offsets. With the offset for the first occurrence being four days and the second being seven days, we get a gap of three days. As three divides 15, this three-day gap will periodically occur, resulting in the algorithm identifying it as a separate pattern. Thus, the pattern is not wrong in itself but rather a result of an erroneous division of the movement trajectory. The identical offset and topic should indicate a correlation between the two patterns, but an inattentive user might interpret the results as unrelated.

The non-displayed patterns are mostly PTTPs with large initial offsets. Keep in mind that we can extract the relevant results as we know what the correct patterns are. In most cases, this is unknown, and all the returned PTTPs seem equally correct. We note that the large initial offsets seem to be an indication of incorrect patterns. However, a correct pattern can also have a large initial offset, so this is not a reliable confidence measure. Another possible verification method is to run the algorithm multiple times with a different number of topics to see which patterns reoccur. Recurring patterns indicate a higher relevance and presence in the dataset.

Note: As a consequence of the initial values of each distribution being set to a random distribution, the results are slightly different for each run. In about half of the runs, the algorithm is not able to distinguish the two inserted topics. Consequently, we get one topic with a mixture of food- and football-related terms and the other topic(s) as non-related to the inserted topics. In these cases, `PSTA+` is still able to find the same trajectories as those displayed in Table 7.1 for the one relevant topic. Again we stress the necessity of running the algorithm multiple times to verify the results.

7.1.2 Patterns Discovered by GeoLPTA

We next analyze two cases for `GeoLPTA` where the period input parameters are set to: (1) $K = 2$ with periods seven and 15 days and (2) $K = 3$ with periods four, seven, and 25 days. Ergo, we test whether the algorithm can find the patterns for correct input and whether it is possible to separate between correct and incorrect PTTPs in the output. The results of the two different input parameters are summarized in Tables 7.2 and 7.3. We first comment on the topics. Later, we discuss the movements per topic, which are not shown fully due to their non-relevance. The complete results are presented in Appendix A.2.

From the first execution, the results indicate that the algorithm successfully can identify the correct topics for a precise input and a partly correct input. Moreover, the topic terms are more accurate than those identified for `PSTA+`, even with slightly lower probabilities. Next, we analyze the second execution and the algorithm's ability to differentiate between correctly and incorrectly returned patterns. Of the three periods, the seven-day topic is correct, and the four-day topic is probably incorrect. The 25-day topic resembles the employment topic with a period of 19 days returned by `TopicPeriodica` before we inserted the patterns. In conclusion, we have one correct, one partly correct, and one incorrect topic in the output. Regrettably, as the algorithm assumes the input parameters are correct, it provides no information to indicate these conclusions.

Topic 1 $T = 7.0$ $\boldsymbol{\delta} = \ldots$	**Topic 2** $T = 15.0$ $\boldsymbol{\delta} = \ldots$ (2.47, 1.50): Calabarzon, PH ...
food: 0.0880, parmesan: 0.0877, italian: 0.0584, best: 0.0307, life: 0.0297, eat: 0.0295, tomatoes: 0.0293, lets: 0.0293, ingredient: 0.0292, consists: 0.0292	win: 0.0908, football: 0.0611, goal: 0.0607, sports: 0.0604, kick: 0.0604, life: 0.0324, lets: 0.0306, ball: 0.0303, job: 0.0088, hiring: 0.0075

TABLE 7.2: Returned PTTPs of `GeoLPTA` for periods seven and 15.0 days.

Topic 1 $T = 4.0$ $\boldsymbol{\delta} = \ldots$	**Topic 2** $T = 7.0$ $\boldsymbol{\delta} = \ldots$	**Topic 3** $T = 25.0$ $\boldsymbol{\delta} = \ldots$
win: 0.0291, football: 0.0195, sports: 0.0191, kick: 0.0189, goal: 0.0185, job: 0.0142, life: 0.0118, hiring: 0.0113, lets: 0.0095, ball: 0.0094	food: 0.1016, parmesan: 0.1012, italian: 0.0675, life: 0.0375, lets: 0.0369, best: 0.0353, eat: 0.0340, tomatoes: 0.0337, consists: 0.0337, ingredient: 0.0337	job: 0.0176, hiring: 0.0160, careerarc: 0.0085, win: 0.0072, like: 0.0067, day: 0.0060, latest: 0.0060, great: 0.0056, amp: 0.0055, football: 0.0053

TABLE 7.3: Returned PTTPs of `GeoLPTA` for periods four, seven, and 25.0 days.

Next, we discuss the pattern movements. A significant problem with the GeoLPTA patterns looks to be the location trajectory. Since the algorithm assumes that the topic is periodic in all locations when it calculates means and standard deviation values, all the time distributions are output as periodic. For this reason, an added filtering step is implemented, which reduces the trajectory from 247 locations to 17 locations for the seven-day pattern and 42 locations for the 15-day pattern. These numbers indicate that the filtering logic was not as effective as intended. Worse yet, the resulting trajectories lack almost all the correct locations, and in the case where a correct location is returned, the offset is wrong. This wrong offset is connected to yet another limitation of the algorithm, namely that each location only has one mean value. Inevitably, the two occurrences of the football topic in the Philippines are merged into one, resulting in a reduced movement and skewed offset. In summary, the offset calculations are limiting, and the filtering step removes too few and the wrong locations.

The failure of the filtering step is connected to a discovered side effect of the non-periodicity detection when the standard deviation is 0. If the periodic pattern repeats itself every period at the exact same offset, we have a perfect pattern, and the standard deviation will statistically be 0. Consequently, it is impossible to tell whether the standard deviation is 0 due to non-periodicity or perfect periodicity. All the patterns are perfectly periodic in the synthetic dataset, resulting in all the correct locations being filtered out. The one exception is the location that occurs multiple times in the movement and is thereby not perfectly periodic when treated as only having one occurrence. Although such patterns are unlikely to take place in real data, it is a fundamental problem that if a pattern is *too* periodic, the algorithm discards it.

To conclude, the algorithm lacks a confidence measure, or an information gain metric [10] to better separate the periodic and non-periodic locations in the analysis phase and to provide some indication of how well the patterns fit the data. A natural starting point is to utilize the Q-function of the EM algorithm better. We leave this for future work. All in all, the patterns discovered using GeoLPTA are not successful PTTPs as the algorithm is not able to identify the cyclic movements of the patterns.

7.1.3 Patterns Discovered by TopicPeriodica

We next present and discuss the PTTPs returned by TopicPeriodica. We first present the reference spots and the topics discovered by the LDA model. Finally, the patterns are presented and analyzed.

Reference Spots and Topics

The reference spots are extracted from a calculated density map illustrated in Figure 7.1. Reference spot IDs mark each detected contour line, and there is an approximate overlay of a world map for reference. The map shows that the algorithm identifies five main areas: North America, South America, Europe, South Asia, and Southeast Asia. Table 7.4 lists the specific countries assigned to each reference spot. We note that the location granularity of TopicPeriodica is coarse compared to the administrative regions utilized in PSTA+ and GeoLPTA, which will result in less detailed movements.

Next, the identified periodic topics are presented in Table 7.5, displayed with their top ten words and relevance score. We define the headers based on the terms of each topic. As the topic modeling utilized is a pure LDA model, it is not surprising

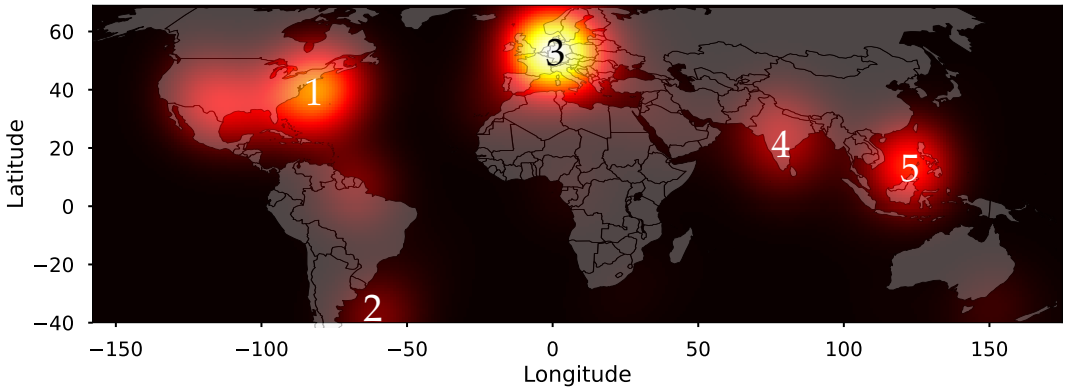


FIGURE 7.1: Density map identifying five reference spots.

Ref. spot 1 (N. America)	Ref. spot 2 (S. America)	Ref. spot 3 (Europe)	Ref. spot 4 (S. Asia)	Ref. spot 5 (SE. Asia)
CA, CO, DO, US	AR	AT, BE, CH, CZ, DE, DK, ES, FR, GB, GR, IE, IT, MA, NL, NO, PL, PT, RO, SE, SI, SK, TN	BD, IN, PK	BN, HK, PH, TW

TABLE 7.4: Reference spots and their corresponding identified country codes.

that the topics from this model are the most accurate of the three models. The topics "Italian food" and "football" are the two periodic topics we are looking for.

Additionally, two more topics are returned as periodic. The periodic employment topic is inherent in the synthetic dataset, while the happy topic is most likely incorrectly returned. We note that the terms of this topic are relatively generic (e.g., love, happy, good, like) and have a low relevance score, so there is a probability that the user would discard the pattern based on its lack of content.

Complete Patterns

We are now ready to present the returned PTTPs. Table 7.6 summarizes the different periods and movements found by the algorithm per topic. The reference spots are represented by their given main area in Table 7.4 for readability. We exclude the movement trajectory of the two additional patterns discussed above as they are not relevant for the discussion.

First, we note that the Italian food topic is returned with the correct seven-day period and the correct movement and offsets. Additionally, `TopicPeriodica` identifies two patterns with periods of two days and three days for the same topic. However, their movement trajectory probabilities tell us that the topic is actually not moving in periods but has a constant presence of $13 - 15\%$ in reference spot 1 in all timestamps. Therefore, we can discard these patterns and consider if such patterns should be discarded as part of the algorithm.

Italian food	Football	Employment	Happy
food: 877.0	win: 810	job: 371.0	like: 141.0
parmesan: 864.0	sports: 539.0	hiring: 332.0	good: 99.0
italian: 576.0	goal: 537	careerarc: 180.0	love: 93.0
lets: 297.0	kick: 537.0	latest: 125.0	people: 82.0
eat: 294.0	football: 536.0	work: 116.0	today: 81.0
best: 294.0	life: 273.0	click: 115.0	know: 73.0
tomatoes: 289.0	ball: 271.0	opening: 81.0	happy: 73.0
life: 289.0	lets: 265.0	apply: 79.0	think: 72.0
beef: 289.0	pts: 3.0	jobs: 74.0	time: 72.0
consists: 288.0	gettin: 2.0	want:72.0	amp: 67.0

TABLE 7.5: Periodic topics discovered by `TopicPeriodica`.

Further, the football pattern returned has the correct period of 15 days and the correct location trajectory and offsets. Again, the algorithm returns two additional patterns with the same topic, this time with a period of three days and seven days. The PTTP with a seven-day period is, similar to the incorrect periods for Italian food, constant for all timestamps in the same reference spot, and is hence discarded. The period of three days, however, does not display such constant presence. Still, we discard it for the following reason: The correct period of 15 days has two occurrences in Southeast Asia (reference spot 5) with offsets 4 and 7. We illustrate the pattern as a sequence of 15 values,

$$****X**X*******,$$

where X in position i means an occurrence of the topic in the fifth reference spot at relative timestamp T_i and $*$ means no such occurrence. If we divide this into five segments of three days, i.e.,

$$*** \quad *X* \quad *X* \quad *** \quad ***,$$

two of the segments (the second and third) has an X at offset 1. Two out of five segments are 40%, and, thus, we get a periodic pattern with a period of three days and an initial offset of one day with a 40% frequency. This is the same logic that made `PSTA+` return a similar three-day pattern. In this case, however, the same movement is also included in the 15-day PTTP and can be discarded without losing any information.

To summarize, `TopicPeriodica` perfectly identifies the correct period and complete location trajectory for each topic. Hence, the algorithm is the only one of the three proposed models that fully capture the inserted PTTPs. Although, the coarse location granularity limits the level of detail of the returned patterns. Like the two other algorithms, it returns some incorrect patterns as well. However, these patterns include information that makes it possible for either the recipient or the program to discard them.

7.1.4 Patterns in the Second Dataset

We lastly present a summary of how each algorithm performed on the second dataset. As we have no prior knowledge of the PTTPs in this dataset, we can only compare the algorithms in how they differ in their results. Ideally, they would all return the

Topic	**Period (days)**	**Movement (offset: ref. spot (frequency))**
Italian food	2.0	0: Europe (0.14), 1: Europe (0.14)
	3.0	0: North America (0.15), 1: North America (0.15), 2: North America (0.13)
	7.0	0: North America (0.99), 1: Europe (1.00)
Football	3.0	1: Southeast Asia (0.40)
	7.0	0: South America (0.07), 1: South America (0.07), 2: South America (0.07), 3: South America (0.07), 4: South America (0.06), 5: South America (0.06), 6: South America (0.07)
	15.0	4: Southeast Asia (1.00), 6: South America (1.00), 7: Southeast Asia (1.00), 11: North America (0.94)
Employment	17.0	-
Happy	102.0	-

TABLE 7.6: Returned PTTPs from `TopicPeriodica`.

#Tweets	Algorithm	Non-empty results (%)	Avg. #PTTPs when results
852	PSTA+	0.0	-
	GeoLPTA	100.0	2
	TopicPeriodica	0.0	-
8525	PSTA+	60.0	1.7
	GeoLPTA	100.0	2
	TopicPeriodica	0.0	-
17 051	PSTA+	90.0	1.8
	GeoLPTA	100.0	2
	TopicPeriodica	0.0	-
42 628	PSTA+	90.0	1.8
	GeoLPTA	100.0	2
	TopicPeriodica	0.0	-
59 679	PSTA+	100.0	2.3
	GeoLPTA	100.0	2
	TopicPeriodica	0.0	-
85 257	PSTA+	100.0	8.5
	GeoLPTA	100.0	2
	TopicPeriodica	100.0	2.3

TABLE 7.7: Results for different dataset sizes. Each algorithm is executed ten times.

same results, but they do not. Table 7.7 summarizes the percentage of times the algorithms returned non-empty results out of the ten iterations per subset. Additionally, when they did return non-empty results, we present how many patterns they returned on average.

The first thing to note is that TopicPeriodica does not return any results unless for the whole dataset. There are two possible reasons for this. The first is that the algorithm cannot find PTTPs as successfully as the two other algorithms when the dataset is small or has shorter timelines. One probable explanation for this is the coarse location granularity of the algorithm, which results in large aggregated datasets that hide detailed information. The second explanation is that there are no reliable patterns to discover due to the short time span. If this is true, the returned PTTPs of the two other algorithms are wrong.

This leads to a second observation: For the smallest datasets, both PSTA+ and GeoLPTA return periodic patterns in at least some of the iterations. Recall in Table 6.2 that the number of timestamps for the two smallest datasets are two and 12 days, i.e., not much data to determine periodic patterns reliably. We also note that the patterns returned are not the same for the different dataset sizes. Of course, GeoLPTA models the data on the assumption that the input parameters are correct. Consequently, the algorithm always returns a fixed number of patterns. This further emphasizes the necessity of a confidence measure in GeoLPTA to evaluate the patterns. In contrast, PSTA+ does not have this "excuse". However, we find that the returned patterns of PSTA+ for the three smallest datasets have periods of two and three days, which is

	PSTA+	GeoLPTA	TopicPeriodica
Correct topics	Yes	Yes	Yes
Correct periods	Yes	Yes	Yes
Correct trajectories	Mostly	No	Yes
Needs multiple runs	Yes	Yes	No
Includes non-existent PTTPs	Yes	Yes	Yes
Possible to discard incorrect PTTPs	No	No	Yes

TABLE 7.8: A summary of the qualitative evaluation of the three proposed models.

logical given the short time span of the datasets. On the other hand, the topics are not particularly distinct.

The results in Table 7.7 indicate that `TopicPeriodica` probably misses some of the smaller patterns, while `PSTA+` includes too many patterns. This is further verified by the average number of patterns returned for the largest subset, where `PSTA+` returns on average 8.5 patterns while `TopicPeriodica` returns 2.3.

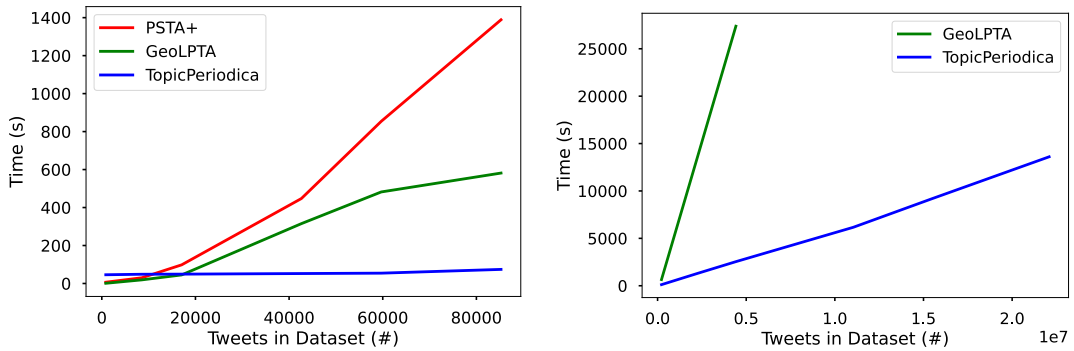
We omit the statistical summary of the results when varying the number of topics due to scalability issues for `PSTA+` which is discussed in Section 7.2. Further, `GeoLPTA` will always return the number of topics (and hence patterns) it is input. We are left with `TopicPeriodica`, which we briefly summarize. The algorithm returns somewhat the same patterns regardless of the number of topics. In all cases, a maximum of four patterns is returned. This indicates that the algorithm is resilient to noise and identifies the correct periodic topics even among numerous non-periodic topics.

7.1.5 Qualitative Comparison

The three algorithms all have different approaches in finding the same result, and so naturally, there are some differences in how well they perform. Firstly, all three algorithms correctly identify the periodic topics and periods. `TopicPeriodica` returns the most accurate topic terms, followed by `GeoLPTA` and then `PSTA+`.

Further, while `GeoLPTA` returns a fixed number of patterns, `PSTA+` returns the most patterns and consequently the most wrong patterns. Moreover, `PSTA+` is greatly affected by the initial values of the distributions and needs multiple runs to extract the correct patterns and filter out incorrect ones. In contrast, `TopicPeriodica` returns a small number of patterns per topic, with only a few wrong patterns, and only needs one execution. We lastly note that from both `PSTA+` and `TopicPeriodica`, we wrongly get a three-day periodic pattern for the football topic that was inherent in the 15-day periodic pattern. As a consequence, we learned to be aware of periodic patterns within the same topic, as they are probably correlated.

Finally, both `PSTA+` and `TopicPeriodica` correctly identifies the location trajectories per pattern. However, while the movements reported by `TopicPeriodica` include probability values per trajectory point, `PSTA+` has no such information. It is thus not able to distinguish between probable true and false patterns. Furthermore, `TopicPeriodica` is more restrictive in how it identifies PTTPs than `PSTA+`, resulting in an overall low probability of false patterns. Lastly, `GeoLPTA` fails in identifying correct pattern movements, resulting in the algorithm *not* being able to solve the PTTP Problem of Definition 2.3.2.



(A) Using the second dataset.

(B) Using the third dataset.

FIGURE 7.2: Runtimes for different dataset sizes.

A color-coded summary of the three algorithms' qualitative evaluation is presented in Table 7.8. The summary illustrates that `TopicPeriodica` is the most accurate of the three algorithms, and `GeoLPTA` the least.

7.2 Quantitative Results and Evaluations

In the quantitative evaluation, we use the second and third datasets to time each algorithm for different values of dataset sizes and numbers of topics. Each result is presented with a short discussion, which is utilized in a revised time complexity analysis. Lastly, we compare the algorithms regarding their calculated and actual space usage.

7.2.1 Effects of Varying the Dataset Size

The average runtime per algorithm when varying the dataset size of the input is presented in Figures 7.2a and 7.2b. They show runtimes for the second and third dataset, respectively.

The initial scalability is tested using the second dataset. Overall, there are great differences in runtimes in Figure 7.2a. While `TopicPeriodica` uses at most 70s to complete, `PSTA+` runs for 1400s before completion for the largest portion of the dataset. `GeoLPTA` is somewhere in between the two with a maximum runtime of about 600s. These vast differences are due to a close to quadratic increase in runtime for `PSTA+` and a linear increase for `GeoLPTA`. `TopicPeriodica` has a relatively constant runtime for dataset sizes up until 60k tweets, where we experience a slight increase.

Next, we analyze the runtimes displayed in Figure 7.2b, recorded using the larger third dataset. Due to its high runtimes, we excluded `PSTA+` from these experiments. While `TopicPeriodica` was close to unaffected by the dataset size for the second dataset, Figure 7.2b shows a clear linear correlation between the dataset size and the runtime of the algorithm. This indicates that the algorithm has a dominating step with a close to constant runtime, but for large enough datasets, the dataset-dependent operations become more time-consuming than this constant runtime. We discuss what this constant runtime is in the revised time complexity analysis in Section 7.2.3. On the other hand, `GeoLPTA` performs similarly on the third dataset as it did on the second, showing a linear correlation. However, this is steeper than the one for `TopicPeriodica`, resulting in such high runtimes the experiment had to be

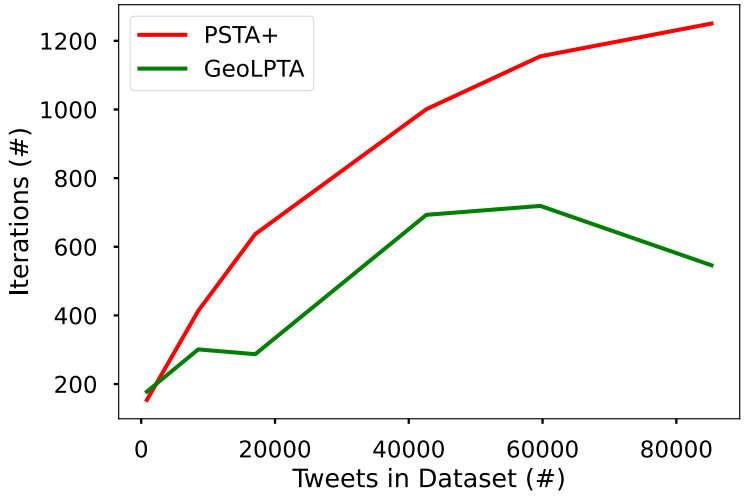


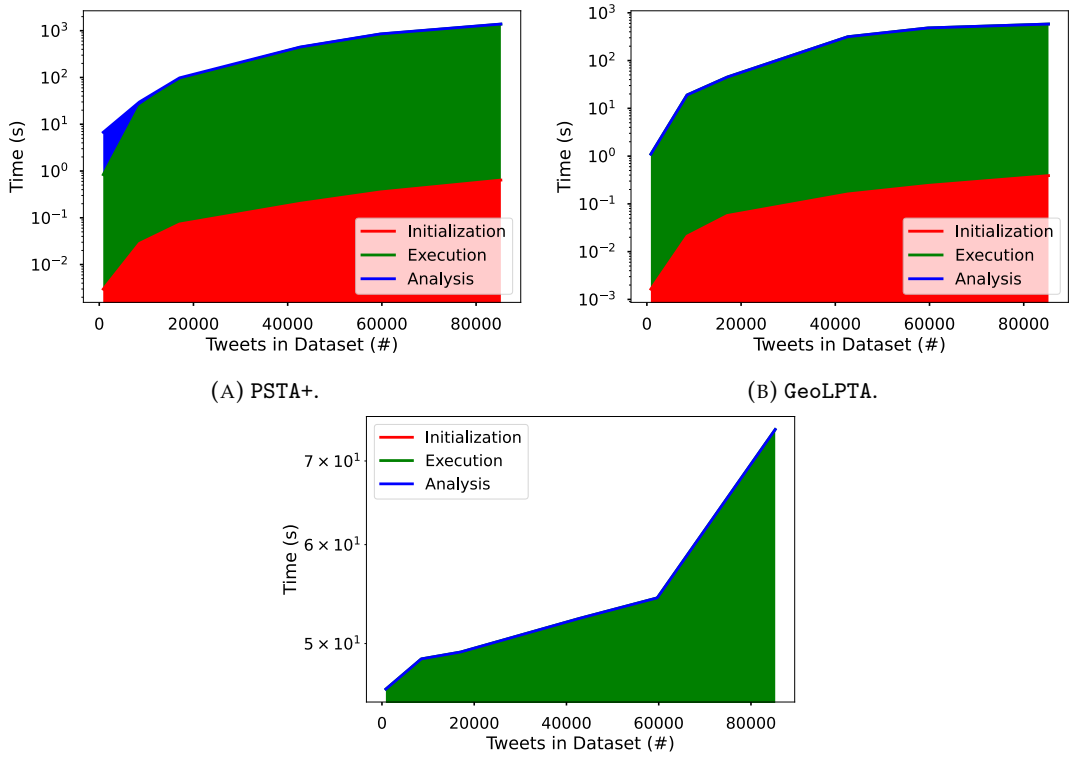
FIGURE 7.3: Number of iterations before convergence when varying the dataset size.

cut after 4 million tweets. The first iteration using 11 million tweets ran for over 72*h* before it was terminated.

Further, we present the average number of iterations of the EM algorithm on the second dataset for `PSTA+` and `GeoLPTA` in Figure 7.3. For `PSTA+`, there is an almost linear correlation between the number of documents and the number of iterations. This explains the quadratic increase in runtime in Figure 7.2a. We mathematically explain this in the final revised time complexity analysis. Consequently, the algorithm is not scalable and, in that sense, useless for larger datasets. On the other hand, there seems to be no such correlation for `GeoLPTA`, and the runtimes correlate linearly with the increase in dataset size and number of iterations. We see that a dip in the number of iterations results in a more gradual, yet still increasing, slope in the runtime.

Lastly, we look at the logarithmic runtimes of the experiments using the second dataset to see what parts consume the most time. Similar views for the third dataset are omitted as they bring no new information. Figure 7.4 illustrate the same runtimes as in Figure 7.2a in logarithmic scale. For `TopicPeriodica`, we see more clearly the slight increase in runtime previously mentioned for the full dataset. The increase is likely connected to the jump in the number of timestamps, from 61 to 191 as listed in Table 6.2. We discuss this further in Section 7.2.3. For the other two algorithms, the graphs show that the initialization runtime grows with the dataset size. Yet, the main part of the algorithms is not surprisingly the execution phase, while the analysis phase is close to trivial. Although, we see that for smaller datasets, the analysis phase is non-negligible for `PSTA+`. As the dataset size grows, however, the execution phase dominates the runtime.

To conclude, when increasing the number of documents in the input, `TopicPeriodica` is negligibly affected for medium-sized datasets and linearly affected for large datasets. Regardless, it is quantitatively superior. `GeoLPTA` displays a steep linear growth in runtime, while `PSTA+` displays a quadratic increase in runtime with the data input size.



(A) PSTA+.

(B) GeoLPTA.

(C) TopicPeriodica.

FIGURE 7.4: Logarithmic view of the runtimes for different dataset sizes.

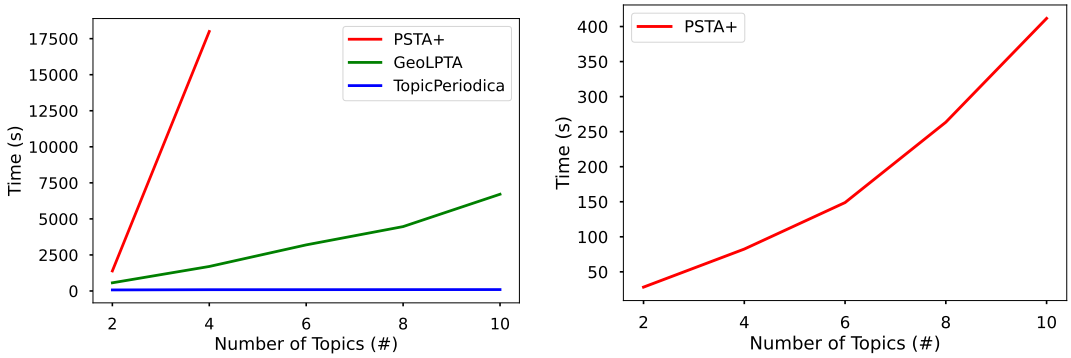
7.2.2 Effects of Varying the Number of Topics

The average runtimes per algorithm when varying the number of topics, K, using the second dataset are presented in Figure 7.5. Further, the number of iterations until convergence for PSTA+ and GeoLPTA are illustrated in Figure 7.6.

First, we analyze the runtimes of GeoLPTA and TopicPeriodica displayed in Figure 7.5a. We see that TopicPeriodica has a low runtime and is negligibly affected by the number of topics. On the other hand, the graph shows a close to quadratic rise in the runtime of GeoLPTA. This is due to a linear correlation between the number of iterations and the number of topics, as displayed in Figure 7.6a.

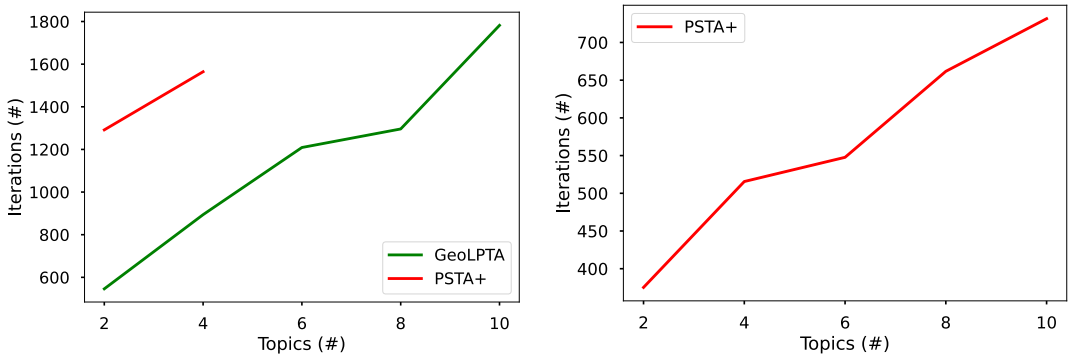
The experiments with PSTA+ were cut at $K = 4$ due to excessive runtimes. To perform a meaningful analysis of PSTA+ when it comes to the number of topics, we executed the algorithm again with 10% the dataset. These are the runtimes presented in Figure 7.5b. Again we experience a quadratic correlation, this time between the number of topics and the runtime, due to the linear correlation evident in Figure 7.6b between the number of iterations and the number of topics. We conclude that PSTA+ is not scalable for an increasing number of topics.

Lastly, we display the runtimes of PSTA+ and GeoLPTA in logarithmic form in Figure 7.7 to get a view of how the different parts of the algorithms are affected by the number of topics. We omit such a view for TopicPeriodica because, as we know, the algorithm only consists of one phase. The runtimes for PSTA+ are from the small dataset execution of 8525 tweets, so the graphs are not directly comparable. PSTA+ has a stable initialization phase for all the variations of K, while the analysis phase



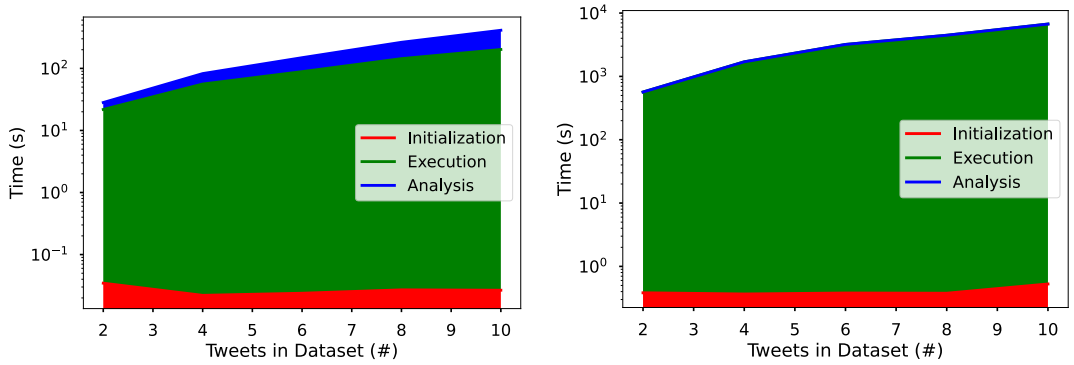
(A) Using the second dataset. (B) Using 10% of the second dataset.

FIGURE 7.5: Runtimes when varying the dataset size.



(A) Using the second dataset. (B) Using 10% of the second dataset.

FIGURE 7.6: Number of iterations before convergence when varying the number of topics.



(A) PSTA+ using a dataset of 8525 tweets.

(B) GeoLPTA.

FIGURE 7.7: Logarithmic view of the runtimes for different values of K.

consumes a more considerable part of the total runtime for higher values of K. This is expected as the analysis phase loops through each topic to find patterns. GeoLPTA has a relatively stable initialization phase regardless of the number of topics and a negligible analysis phase. The negligible analysis phase is due to the low complexity of the loop of topics in GeoLPTA as opposed to in PSTA+. In general, just as for the experiments with different data sizes, the dominating phase of both algorithms is the execution phase.

Overall, when increasing the number of assumed topics in the dataset, TopicPeriodica is hardly affected. In contrast, both GeoLPTA and PSTA+ display poor scalabilities with close to quadratic correlations between runtime and the number of topics. Thus, we conclude that TopicPeriodica is the superior algorithm within the experiments discussed in this section.

7.2.3 Revised Time Complexity Analysis

With the experiments revealing qualities of the unknown variables of the time complexities, we revise these for each algorithm and update them if possible. Recall from Chapter 5, the time complexities of the algorithms are

PSTA+: $O(iter_1 * Knu + iter_1 * K|L||\mathcal{T}||V|)$

GeoLPTA: $O(iter_2 * Knu)$

TopicPeriodica: $O(whn + LDA_t + Kd|\mathcal{T}|\log|\mathcal{T}| + Kp\bar{d}|\mathcal{T}| + m^2 \log m + 2mdT)$

The linear correlation discovered between the number of documents and the number of iterations for PSTA+ indicates that $iter_1 = C_0 n$ for some constant $C_0 > 0$. Moreover, the linear correlation between the number of iterations and the number of topics enables a further decomposition so that $iter_1 = C_1 Kn$ for some constant $C_1 > 0$. This indicates the time complexity of PSTA+ is in fact

$$\begin{aligned} &O(iter_1 * Knu + iter_1 * K|\mathcal{T}||L||V|) \\ =&O(C_1 K^2 n^2 u + C_1 n K^2 |\mathcal{T}||L||V|), \end{aligned}$$

resulting in a quadratic runtime both when it comes to the dataset size *and* the number of topics. Given that $K \ll n$[1], K^2 is negligible in the expression. However, with

[1] K is a lot less than n.

more documents comes more possible topics, and so a larger dataset will lead to a greater than quadratic rise in runtime. We are left with a poorly scalable algorithm that is only applicable for small datasets with few patterns.

`GeoLPTA` was affirmed to be linearly correlated with the number of documents and not quadratically like `PSTA+`. However, the linear correlation between the number of iterations and the number of topics means we rewrite the number of iterations to $iter_2 = C_2K$ for a constant $C_2 > 0$. This propagates to the time complexity, and we get $O(iter * Knu) = O(C_2K^2nu)$. As we can see, this is quadratic regarding the number of topics, but again, as $K \ll n$, this is not a big concern.

The runtimes of `TopicPeriodica` are not noticeably affected by the dataset size initially but rather by the number of timestamps. However, for larger datasets, they linearly follow the dataset size. The most likely explanation of this is the unknown impact of the LDA algorithm from the time complexity listed above. Recall that we pass $d * |\mathcal{T}|$ "documents"[2] to the LDA algorithm, which is somewhat scaled with the number of documents and the number of iterations [33]. Consequently, we get $O(LDA_t) = O(iter_3 d|\mathcal{T}| * LDA_{t1})$, where $iter_3$ is the number of iterations (set to 1500 in our experiments), and LDA_{t1} is an unknown value. Consequently, while n is small, the runtime of `TopicPeriodica` is close to constant with a slight dependence on $|\mathcal{T}|$ and d. When $O(LDA_t) < O(whn)$, the algorithm instead scales linearly with the number of documents. Conclusively, the revised time complexity of `TopicPeriodica` is $O(whn + iter_3 d|\mathcal{T}|LDA_{t1})$.

7.2.4 Memory Usage

Lastly, we analyze the space usage of the algorithms, as this can be equally important when we are dealing with large datasets. First, we look at the space complexities before shortly commenting on the space used during the experiments.

Recall from Chapter 5 that the space requirements for the three algorithms are

PSTA+: $O(Knu + K|V| + K|\mathcal{T}||L|)$

GeoLPTA: $O(Knu + K|V| + K|\mathcal{T}||L|)$

TopicPeriodica: $O(wh + LDA_s + mpdT|\mathcal{T}|)$

First note that the space complexities for `PSTA+` and `GeoLPTA` are the same. Next, the space complexity of `TopicPeriodica` contains the unknown value LDA_s. As LDA is a latent topic model, we can assume that the memory usage does not greatly differ from the memory usage of the latent topic model-based algorithms `PSTA+` and `GeoLPTA` without the spatiotemporal factors, i.e. $O(LDA_s) \leq O(Knu + K|V|)$. Further, there is no reason to divide the geographic space into very small grid cells, so $wh \leq n$. Lastly, often in the experiments, the number of iterations was the initial number of segments, so we get $m = \lceil |\mathcal{T}|/T \rceil$, and both p (the average number of periods per location and topic) and d (the number of reference spots) are insignificant compared to $|\mathcal{T}|^2$. We get

$$\begin{aligned} & O(|\mathcal{T}|/T * pdT|\mathcal{T}|) \\ =& O(|\mathcal{T}|^2 pd) \\ =& O(|\mathcal{T}|^2). \end{aligned}$$

[2] One combined document per reference spot and timestamp combination.

	Space usage (MiB)	Usage of reserved memory (%)
PSTA+	2678	46.34
GeoLPTA	2627	45.79
TopicPeriodica	407	16.60

TABLE 7.9: Space usage of the three algorithms on the full second dataset.

We see that if $K|L| < |\mathcal{T}|$ and $O(LDA_s) = O(Knu + K|V|)$, `TopicPeriodica` requires slightly more memory than the two topic modeling-based algorithms. In any case, the space usage of the three algorithms will be relatively low compared to the CPU usage.

The memory usage observed during the experiments indicates that $O(LDA_s) \ll O(Knu + K|V|)$. For the full second dataset, the approximate maximum memory usage is presented in Table 7.9, with `TopicPeriodica` utilizing only a fraction of the memory utilized by `PSTA+` and `GeoLPTA`. We note that the memory usage of `PSTA+` and `GeoLPTA` is higher than the expected memory usage from the space complexities. The reason for this is that due to high runtimes, the algorithms were implemented with additional statistical analysis in the initialization step to avoid having to recalculate these at each iteration. We stored information per document, time and location, and word in the vocabulary. Consequently, the memory usage more than doubled.

At the same time, the memory usage of all the algorithms is low compared to the CPU usage and is consequently a non-issue. The only exception might be for `GeoLPTA` on the third dataset, where we had to terminate the algorithm at 11 million tweets due to unexpected high runtimes. This might be partly due to memory overflow, but the memory utilization was not monitored for this dataset, so further research is necessary to conclude this issue.

To conclude, `TopicPeriodica` utilizes significantly less memory than the other algorithms. However, none of the algorithms utilize unrealistically much space and are in that sense all passable when it comes to space usage.

7.3 Overall Performance

The comparisons within the three different metrics clearly show that `TopicPeriodica` is the superior algorithm. It returns the most accurate results, is substantially faster, and scales better for larger datasets and a varying number of topics. Compared to the other proposed models, a downside is that the discovered locations span large areas, resulting in less detailed patterns. However, it is worth noting that this is due to the algorithm being independent on a predefined set of locations, as it automatically defines critical regions based on the input data.

Due to `GeoLPTA`'s inability to identify location trajectories per topic, it is irrelevant that it executes noticeably faster and scales better than `PSTA+`. If it were possible to define a metric to measure the *actual* periodicity of the locations and not the modeled periodicity, the approach would be interesting for further research. In contrast,

the high runtimes of PSTA+ make the fact that the algorithm returns correct results irrelevant, as it does not scale and is not applicable on larger datasets.

Why TopicPeriodica is so much faster than the two other algorithms is probably partly because it uses a standard LDA implementation. While PSTA+ and GeoLPTA implements the topic modeling from scratch, TopicPeriodica uses an existing model which is standardized and implemented efficiently. The log view of the runtimes of PSTA+ and GeoLPTA showed that the EM algorithm (execution phase) dominated the total runtime. Optimizations applied to this phase would likely decrease the runtime gap between the algorithms.

Nonetheless, TopicPeriodica is superior in both the qualitative and quantitative evaluation and is the only one of the three proposed algorithms at this point that is applicable to real-world data. Moreover, the algorithm is a starting point for optimizations and other approaches that can further enhance performance. We discuss some of these in Section 8.2.

7.4 Summary

This chapter answered RQ3 as it presented the experimental results of executing the three proposed algorithms; PSTA+, GeoLPTA and TopicPeriodica. Each algorithm returned a qualitative result (PTTPs) and a quantitative result (runtimes). Combined, the results constitute the performance of each algorithm. We evaluated each algorithm in itself and compared it to the other algorithms. TopicPeriodica was concluded to have the best performance of the three algorithms.

Chapter 8

Conclusions and Future Work

As mobile phones are increasingly a part of our everyday attire, the amount of spatiotemporal textual data grows exponentially. Such rich data consists of timestamps, GPS coordinates, and texts, and it contains concealed information about the general public. In this thesis, we aimed to develop algorithms that can extract some of this information. More specifically, we wished to extract periodic information diffusion from Twitter data by mining Periodic Topic Trajectory Patterns (PTTP). Analyzing how information spreads geographically in periodic intervals can predict future patterns and detect anomalies as they happen. Twitter data is rich in content as it contains opinions, experiences, and historical events, making it a perfect input data source for a PTTP algorithm.

We began our work by defining the problem formally before performing a literature review on relevant research areas. We found three main approaches that seemed possible to use as a basis when developing more complex algorithms. The information gained in this step was used to define three algorithms that mine Periodic Topic Trajectory Patterns. These algorithms were implemented and tested using three different datasets. Finally, the experiments were used to evaluate each algorithm.

8.1 Conclusions

We introduced this thesis by presenting three research questions that would guide our approach throughout the thesis. In this section, we discuss and answer these questions based on our research and experiments.

RQ1 *How can a Periodic Topic Trajectory Pattern be formally defined?*

We are the first to our knowledge to define the Periodic Topic Trajectory Pattern. The pattern is concisely defined in Definition 2.3.1 and illustrated with an example in Figure 2.2. The definition relies on the numerous definitions of Chapter 2.

RQ2 *What algorithms exist that partly solve the problem? Can we expand them, so they fully solve the problem?*

Three algorithms were presented that represent three different approaches in solving the PTTP problem. The first approach adds a periodicity detection step to a geographic probabilistic topic modeling algorithm. We implement this by expanding the algorithm PSTA [20] with an additional periodic analysis step, resulting in the algorithm `PSTA+`. The second approach applies a geographical dimension to periodic probabilistic topic modeling. The resulting model is based on the algorithm

LPTA [3], which we called GeoLPTA. Lastly, the third approach adds a topic modeling step to a periodic trajectory pattern mining algorithm, in our case, the algorithm Periodica [18]. This resulted in a modified algorithm TopicPeriodica.

The difference between the two topic modeling-based algorithms PSTA+ and GeoLPTA is the order of modeling. PSTA+ models the data without any assumptions of periodicity and checks for periodic patterns later, while GeoLPTA directly models the patterns periodically. Both TopicPeriodica and PSTA+ fully solve the problem, while GeoLPTA is only able to solve the problem partially.

There are, of course, more approaches to developing an algorithm to solve the defined problem. We discuss some of these in Section 8.2.

RQ3 *If yes, what is the performance?*

GeoLPTA has a satisfying runtime and scales relatively well but is not able to fully identify PTTP patterns due to its inability to define the trajectory movements of the patterns. This is due to the assumption that all locations are periodic, and so they are modeled accordingly. Consequently, the performance of GeoLPTA is poor. On the other hand, PSTA+ correctly identifies PTTPs, but does not scale and is consequently useless in a data mining context. Further, the algorithm includes multiple PTTPs in its results that were incorrect, without any information to indicate the confidence of each pattern. Also this algorithm is concluded to have a poor performance. The remaining algorithm, TopicPeriodica, is the only one of the three approaches that display a satisfactory performance. We found this algorithm to give precise and correct patterns with an acceptable runtime that is not highly affected by neither the dataset size nor the number of topics for medium-sized datasets and that scales linearly with the dataset size for large datasets. In those cases where the algorithm returned incorrect patterns, these patterns also included data that made it possible to discard them.

Overall, the topic modeling approaches displayed poor performance. The periodic geographical approach was best in terms of the pattern quality and the efficiency and scalability of the algorithm.

8.2 Future Work

This thesis initially defines and explores the PTTP problem, leaving multiple optimizations and approaches unexplored. Firstly, Torpedo by Wang et al. [4] is a close relative to LPTA, PSTA+ and LDA. A natural development of the algorithms presented in this chapter is, therefore, to extend Torpedo with a geographical dimension and compare it to PSTA+ regarding qualitative results and runtime. Other promising topic modeling algorithms include LATM [21], and LGTA [23], which only lack the temporal component.

Another area of interest is how the text is interpreted. In this thesis, we have focused on probabilistic topic modeling. However, for short documents like Twitter data, most tweets only contain one topic. Consequently, the algorithms spend unnecessary resources looking for multiple topics in each tweet. We could instead use word embeddings (e.g. *word2vec* [31], BERT [39]) to cluster documents based on their similarity, or combine LDA and word embeddings [40].

Next, we recommend exploring alternative approaches to model a PTTP algorithm different from the three presented in this thesis. While TopicPeriodica is based on

periodic spatiotemporal pattern mining, we could go further back and look at trajectory pattern mining [16, 41] or even sequential pattern mining [42–44]. As this algorithm was the most successful of the three approaches of this thesis, further research on this area is a logical next step. We also encourage alternative implementations of `TopicPeriodica`, like detecting reference spots from time-dependent densities or other measures that also capture infrequent periodical areas [18].

Lastly, automating the algorithms to find the number of topics, location and time granularities, and other hyperparameters based on the input data would increase the algorithms' accuracy and possibly eliminate the need to run them multiple times. We could, e.g., use the Chinese Restaurant Franchise scenario like Guo and Gong [24] or Schwarz's Bayesian information criterion as previously discussed. We would also consider utilizing Gaussian Process models for more accurate periodicity detection [2]. Such changes should be considered incorporated into the existing algorithms or any new PTTP algorithms.

Appendix A

Complete Outputs

This appendix presents the complete outputs of the PSTA+ and GeoLPTA executions. We omit the details from the main paper as they are not relevant to the thesis research questions.

A.1 PSTA+

We present the returned patterns of PSTA+ for two and five patterns in Tables A.1 and A.3. The topics of Table A.3 are presented in Table A.2 due to space restrictions. These results include patterns that are incorrect and in some cases they can be discarded.

Several patterns have large initial offsets as the tables display, which indicate the patterns are not present throughout the timeline. If we wanted to mine *full* periodic patterns, we could include a filtering step that removed all patterns with an initial offset less than the period. However, in many cases, the user is interested in the sub-patterns. Consequently, we leave the algorithm as it is and give the user the decision to filter out patterns based on domain knowledge or user criteria after output.

A.2 GeoLPTA

We present the full output when executing GeoLPTA with two periodic patterns of seven and 15 days in Table A.4, and four, seven and 25 days in Table A.5. These tables include the full location trajectories returned per pattern, which were incorrect compared to the inserted patterns the algorithm was supposed to find. Parts of the movement for the 25-day pattern in Table A.5 are hidden due to space limitations.

Topic (term: probability)	**Period (days)**	**Initial Offset (days)**	**Movement (offset: location)**
food: 0.1108, parmesan: 0.1104, italian: 0.0738, eat: 0.0370, ingredient: 0.0368, beef: 0.0368, tomatoes: 0.0365, best: 0.0330, orange: 0.0010, accounting: 0.0010	3	12	0: Hamburg, DE
	7	0	0: Vichada, CO 1: Hamburg, DE
	79	120	41: Metro Manila, PH
	85	114	29: Metro Manila, PH
	91	138	47: New Jersey, US
	93	101	8: Metro Manila, PH
	102	80	80: Arizona, US
win: 0.1045, football: 0.0699, sports: 0.0699, goal: 0.0691, kick: 0.0691, ball: 0.0349, driver: 0.0030, transportation: 0.0025, truck: 0.0022, place: 0.0019	3	4	1: Calabarzon, PH
	4	56	0: Ontario, CA
	5	46	1: Buenos Aires, AR
	7	36	1: Buenos Aires, AR 4: Ontario, CA
	8	127	7: Central Visayas, PH
	12	63	3.0: Madeira, PT
	15	4	4: Calabarzon, PH 6: Buenos Aires, AR 11: Ontario, CA
	27	15	15: England, GB
	79	119	40: Metro Manila, PH
	128	4	4: Central Visayas, PH
	204	76	76: Georgia, US

TABLE A.1: The complete results of the PSTA+ algorithm for $K = 2$.

Topic 1	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**
parmesan: 0.1830, food: 0.1790, italian: 0.1217, beef: 0.0612, ingredient: 0.0610, tomatoes: 0.0607, eat: 0.0589, best: 0.0527, amu: 0.0012, tall: 0.0008	win: 0.1929, kick: 0.1303, goal: 0.1298, sports: 0.1280, football: 0.1266, ball: 0.0640, thoughts: 0.0011, coach: 0.0010, lil: 0.0010, saint: 0.0009	wanna: 0.0127, driver: 0.0126, class: 0.0096, transportation: 0.0086, truck: 0.0080, taking: 0.0078, owner: 0.0062, operator: 0.0056, racist: 0.0039, weiner: 0.0038	disrespect: 0.0060, chicago: 0.0050, manufacturing: 0.0049, sister: 0.0046, senior: 0.0044, feel: 0.0043, point: 0.0039, old: 0.0039, miss: 0.0037, sept: 0.0034	korea: 0.0066, god: 0.0058, years: 0.0054, wow: 0.0049, north: 0.0045, photo: 0.0045, game: 0.0044, told: 0.0044, healthcare: 0.0042, trump: 0.0040

TABLE A.2: Topics identified in PSTA+ for $K = 5$ represented by their top ten terms with corresponding probabilities.

Topic ID	Period (days)	Initial Offset (days)	Movement (offset: location)
1	7	0	0: Vichada, CO 1: Hamburg, DE
	147	120	120: NCT, IN
2	2	51	0: Ontario, CA, 1: Buenos Aires, AR
	3	4	1: Calabarzon, PH
	5	61	1: Buenos Aires, AR
	8	53	5: Buenos Aires, AR
	10	91	1: Ontario, CA
	15	4	4: Calabarzon, PH 6: Buenos Aires, AR 11: Ontario, CA
	129	7	4: Central Visayas, PH
3	80	52	52: Pennsylvania, US
	129	6	6: Central Visayas, PH
	207	149	149: California, US
4	60	23	23: Ontario, CA
	122	108	108: NCT, IN
	145	123	123: NCT, IN
	199	151	151: Dhaka, BD
	204	134	134: Punjab, PK
5	7	128	2: Central Visayas, PH

TABLE A.3: The complete results of the PSTA+ algorithm for five topics.

Topic (term: prob)	**Period (days)**	**Movement ((mean, std. deviation): location)**
food: 0.0880, parmesan: 0.0877, italian: 0.0584, best: 0.0307, life: 0.0297, eat: 0.0295, tomatoes: 0.0293, lets: 0.0293, ingredient: 0.0292, consists: 0.0292	7.0	(1.27, 0.44): Western Cape, ZA, (1.35, 0.48): Kentucky, US, (1.46, 0.50): Iowa, US, (1.47, 0.50): South Australia, AU, (1.55, 0.50): Qabis, TN, (1.91, 0.29): Maryland, US, (2.22, 0.41): Ohio, US, (2.37, 0.48): Rio de Janeiro, BR, (2.44, 0.50): Illinois, US, (2.46, 0.50): Karnataka, IN, (2.46, 0.50): Ontario, CA, (2.49, 0.50): Madeira, PT, (2.51, 0.50): England, GB, (2.56, 0.50): Pennsylvania, US, (3.51, 0.50): Oklahoma, US, (3.90, 0.29): Arizona, US, (3.92, 0.27): California, US
win: 0.0908, football: 0.0611, goal: 0.0607, sports: 0.0604, kick: 0.0604, life: 0.0324, lets: 0.0306, ball: 0.0303, job: 0.0088, hiring: 0.0075	15.0	(0.28, 0.45): Alexandria, EG, (1.52, 0.50): Skane, SE, (1.67, 0.47): Berlin, DE, (2.47, 1.50): Calabarzon, PH, (2.53, 0.50): Newfoundland and Labrador, CA, (3.33, 0.94): Colorado, US, (3.41, 0.49): Lombardy, IT, (3.50, 0.50): Western Cape, ZA, (3.64, 0.48): Ash Shariqah, AE, (3.65, 0.48): New Hampshire, US, (4.31, 0.68): Pennsylvania, US, (4.37, 0.48): Tamil Nadu, IN, (4.69, 0.46): Arizona, US, (5.43, 0.70): New York, US, (5.44, 0.50): Victoria, AU, (5.48, 0.50): Virginia, US, (5.65, 0.48): Munster, IE, (6.29, 0.45): Arkansas, US, (6.52, 0.50): Karnataka, IN, (6.54, 0.50): Central Visayas, PH, (6.59, 0.49): Wales, GB, (6.67, 0.47): Texas, US, (7.00, 0.87): Oregon, US, (7.17, 0.75): England, GB, (7.29, 0.79): Metro Manila, PH, (7.37, 0.48): Massachusetts, US, (7.43, 0.54): Picardie, FR, (7.47, 0.50): Madeira, PT, (7.47, 0.50): Nassarawa, NG, (7.97, 0.74): New South Wales, AU, (8.44, 0.50): NCT, IN, (8.49, 0.50): Scotland, GB, (8.50, 0.50): Bangkok, TH, (8.65, 0.48): California, US, (8.68, 0.47): Georgia, US, (9.36, 0.48): Queensland, AU, (9.76, 0.43): Nevada, US, (10.00, 1.00): Punjab, PK, (10.43, 0.50): Northern Ireland, GB, (10.55, 0.50): Florida, US, (11.58, 0.49): Central Greece, GR, (12.57, 0.49): Nordland, NO, (13.64, 0.48): Punjab, IN

TABLE A.4: Returned PTTPs of `GeoLPTA` for periods seven and 15.0 days.

Topic (term: prob)	**Period (days)**	**Movement ((mean, std. deviation): location)**
win: 0.0291, football: 0.0195, sports: 0.0191, kick: 0.0189, goal: 0.0185, job: 0.0142, life: 0.0118, hiring: 0.0113, lets: 0.0095, ball: 0.0094	4.0	(0.34, 0.48): Gauteng, ZA, (0.46, 0.50): Maine, US, (1.15, 0.36): Dhaka, BD, (1.30, 0.48): Scotland, GB, (1.48, 0.50): Georgia, US, (1.55, 0.50): Karnataka, IN, (1.55, 0.50): Metro Manila, PH, (1.62, 0.49): Newfoundland and Labrador, CA, (2.96, 0.20): Lagos, NG
food: 0.1016, parmesan: 0.1012, italian: 0.0675, life: 0.0375, lets: 0.0369, best: 0.0353, eat: 0.0340, tomatoes: 0.0337, consists: 0.0337, ingredient: 0.0337	7.0	(1.41, 0.49): Kentucky, US, (1.48, 0.76): Minnesota, US, (1.62, 0.49): Western Cape, ZA, (2.05, 0.21): Scotland, GB, (2.20, 0.40): Ohio, US, (2.36, 0.48): Telangana, IN, (2.48, 0.50): England, GB, (2.56, 0.50): Maharashtra, IN, (2.95, 0.21): New York, US, (3.06, 0.23): California, US, (4.50, 0.50): Oregon, US
job: 0.0176, hiring: 0.0160, careerarc: 0.0085, win: 0.0072, like: 0.0067, day: 0.0060, latest: 0.0060, great: 0.0056, amp: 0.0055, football: 0.0053	25.0	(0.92, 1.00): Lower Saxony, DE, (1.45, 0.50): Vysocina, CZ, (1.67, 0.47): Berlin, DE, (2.12, 1.47): Western Australia, AU, (2.40, 0.49): Bangkok, TH, (3.56, 0.50): Central Luzon, PH, (3.71, 0.45): Alsace, FR, (4.09, 1.06): Illinois, US, (4.41, 0.90): Telangana, IN, (4.44, 0.50): Coimbra, PT, (5.37, 1.23): Madeira, PT, (5.86, 0.99): Colorado, US, (6.01, 0.85): Tennessee, US, (6.33, 0.47): Kuala Lumpur, MY, (6.52, 1.30): Massachusetts, US, (7.26, 1.11): Virginia, US, (7.32, 0.70): Metro Manila, PH, ... (15.66, 0.48): Nevada, US, (16.00, 0.86): Newfoundland and Labrador, CA, (16.08, 0.93): Dhaka, BD, (16.21, 0.98): Qabis, TN, (16.48, 0.50): Texas, US, (16.93, 0.69): Florida, US, (17.11, 0.99): Rio de Janeiro, BR, (17.32, 0.46): Davao, PH, (17.62, 0.49): Odisha, IN, (18.65, 0.48): North Carolina, US, (19.86, 1.66): Maryland, US, (20.05, 0.74): Stara Zagora, BG, (20.74, 0.94): Nassarawa, NG, (20.95, 0.76): Oregon, US, (21.52, 0.50): Skane, SE, (22.57, 0.49): Nordland, NO, (23.65, 0.48): New Hampshire, US

TABLE A.5: Returned PTTPs of `GeoLPTA` for periods four, seven, and 25.0 days.

Bibliography

[1] Michail Vlachos et al. "Identifying similarities, periodicities and bursts for online search queries". In: *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. 2004, pp. 131–142.

[2] Daniel Preoţiuc-Pietro and Trevor Cohn. "A temporal model of text periodicities using Gaussian Processes". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. 2013, pp. 977–988.

[3] Zhijun Yin et al. "LPTA: A probabilistic model for latent periodic topic analysis". In: *2011 IEEE 11th International Conference on Data Mining*. IEEE. 2011, pp. 904–913.

[4] Jingjing Wang, Hongbo Deng, and Jiawei Han. "Torpedo: topic periodicity discovery from text data". In: *Next-Generation Analyst III*. Vol. 9499. International Society for Optics and Photonics. 2015, 94990A.

[5] Herman J Loether and G.D. McTavish. *Descriptive and inferential statistics; an introduction*. Tech. rep. 1993.

[6] Jiawei Han, Wan Gong, and Yiwen Yin. "Mining segment-wise periodic patterns in time-related databases." In: *KDD*. Vol. 98. 1998, pp. 214–218.

[7] Rakesh Agrawal, Ramakrishnan Srikant, et al. "Fast algorithms for mining association rules". In: *Proc. 20th int. conf. very large data bases, VLDB*. Vol. 1215. Citeseer. 1994, pp. 487–499.

[8] Jiawei Han, Guozhu Dong, and Yiwen Yin. "Efficient mining of partial periodic patterns in time series database". In: *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*. IEEE. 1999, pp. 106–115.

[9] Jiong Yang, Wei Wang, and Philip S. Yu. "Mining asynchronous periodic patterns in time series data". In: *IEEE Transactions on Knowledge and Data Engineering* 15.3 (2000), pp. 613–628.

[10] Jiong Yang, Wei Wang, and Philip S Yu. "InfoMiner: mining surprising periodic patterns". In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. 2001, pp. 395–400.

[11] R Yang, Wei Wang, and Philip S Yu. "InfoMiner+: mining partial periodic patterns with gap penalties". In: *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* IEEE. 2002, pp. 725–728.

[12] Sheng Ma and Joseph L Hellerstein. "Mining partially periodic event patterns with unknown periods". In: *Proceedings 17th International Conference on Data Engineering*. IEEE. 2001, pp. 205–214.

[13] Christos Berberidis et al. "Multiple and partial periodicity mining in time series databases". In: *ECAI*. Vol. 2. 2002, pp. 370–374.

[14] Christos Berberidis et al. "On the discovery of weak periodicities in large time series". In: *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer. 2002, pp. 51–61.

[15] Michail Vlachos, Philip Yu, and Vittorio Castelli. "On periodicity detection and structural periodic similarity". In: *Proceedings of the 2005 SIAM international conference on data mining*. SIAM. 2005, pp. 449–460.

[16] Huiping Cao, Nikos Mamoulis, and David W Cheung. "Discovery of periodic patterns in spatiotemporal sequences". In: *IEEE Transactions on Knowledge and Data Engineering* 19.4 (2007), pp. 453–467.

[17] Shirli Bar-David et al. "Methods for assessing movement path recursion with application to African buffalo in South Africa". In: *Ecology* 90.9 (2009), pp. 2467–2479.

[18] Zhenhui Li et al. "Mining periodic behaviors for moving objects". In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010, pp. 1099–1108.

[19] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *the Journal of machine Learning research* 3 (2003), pp. 993–1022.

[20] Qiaozhu Mei et al. "A probabilistic approach to spatiotemporal theme pattern mining on weblogs". In: *Proceedings of the 15th international conference on World Wide Web*. 2006, pp. 533–542.

[21] Chong Wang et al. "Mining geographic knowledge using location aware topic model". In: *Proceedings of the 4th ACM workshop on Geographical information retrieval*. 2007, pp. 65–70.

[22] Sergej Sizov. "GeoFolk: latent spatial semantics in web 2.0 social media". In: *Proceedings of the third ACM international conference on Web search and data mining*. 2010, pp. 281–290.

[23] Zhijun Yin et al. "Geographical topic discovery and comparison". In: *Proceedings of the 20th international conference on World wide web*. 2011, pp. 247–256.

[24] Jinjin Guo and Zhiguo Gong. "A nonparametric model for event discovery in the geospatial-temporal space". In: *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 2016, pp. 499–508.

[25] Yee Whye Teh et al. "Hierarchical dirichlet processes". In: *Journal of the american statistical association* 101.476 (2006), pp. 1566–1581.

[26] Brian J Worton. "Kernel methods for estimating the utilization distribution in home-range studies". In: *Ecology* 70.1 (1989), pp. 164–168.

[27] Thomas Hofmann. "Probabilistic latent semantic analysis". In: *arXiv preprint arXiv:1301.6705* (1999).

[28] Thomas K Landauer and Susan T Dumais. "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge." In: *Psychological review* 104.2 (1997), p. 211.

[29] Bruno De Finetti. *Theory of probability*. Vol. 1-2. Reprint of the 1975 translation. John Wiley & Sons Ltd., 1990.

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

[31] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[32] Quan Yuan et al. "Who, where, when and what: discover spatio-temporal topics for twitter users". In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2013, pp. 605–613.

[33] David A Sontag and Daniel M Roy. "Complexity of Inference in Latent Dirichlet Allocation." In: *NIPS*. Citeseer. 2011, pp. 1008–1016.

[34] Zhao Jianqiang and Gui Xiaolin. "Comparison research on text pre-processing methods on twitter sentiment analysis". In: *IEEE Access* 5 (2017), pp. 2870–2879.

[35] Jon Olav Hauglid, Norvald H Ryeng, and Kjetil Nørvåg. "The dascosa-db grid database system". In: *Grid and Cloud Database Management*. Springer, 2011, pp. 87–106.

[36] Rajkumar Arun et al. "On finding the natural number of topics with latent dirichlet allocation: Some observations". In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer. 2010, pp. 391–402.

[37] Weizhong Zhao et al. "A heuristic approach to determine an appropriate number of topics in topic modeling". In: *BMC bioinformatics*. Vol. 16. 13. Springer. 2015, pp. 1–10.

[38] Murzintcev Nikita. "Select number of topics for LDA model". In: *Available on: https://cran. rproject. org/web/packages/ldatuning/vignettes/topics. html.[March 3rd, 2019]* (2016).

[39] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

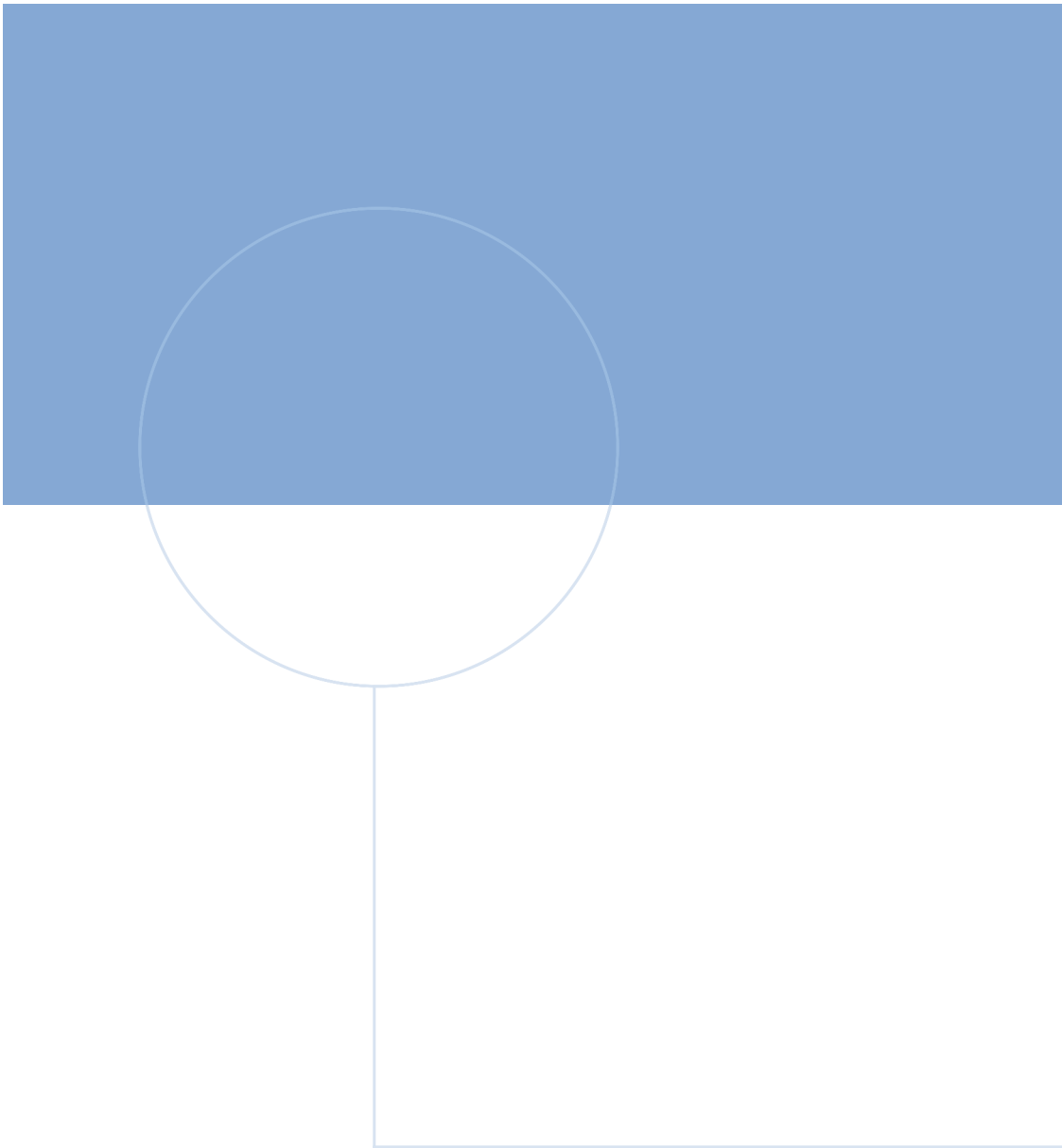
[40] Rajarshi Das, Manzil Zaheer, and Chris Dyer. "Gaussian lda for topic models with word embeddings". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 2015, pp. 795–804.

[41] Fosca Giannotti, Mirco Nanni, and Dino Pedreschi. "Efficient mining of temporally annotated sequences". In: *Proceedings of the 2006 SIAM international conference on data mining*. SIAM. 2006, pp. 348–359.

[42] Rakesh Agrawal and Ramakrishnan Srikant. "Mining sequential patterns". In: *Proceedings of the eleventh international conference on data engineering*. IEEE. 1995, pp. 3–14.

[43] Jiawei Han et al. "Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth". In: *proceedings of the 17th international conference on data engineering*. Citeseer. 2001, pp. 215–224.

[44] Philippe Fournier-Viger et al. "A survey of sequential pattern mining". In: *Data Science and Pattern Recognition* 1.1 (2017), pp. 54–77.



NTNU

Kunnskap for ei betre verd