Automatic Creation of a Large-Scale Tempo-Spatial and Semantic Medieval European Information System

Juri Opitz

Dept. of Computational Linguistics, Heidelberg University, 69120 Heidelberg

Abstract

In this paper, we automatically create a large **tempo-spatial and semantic medieval information system**, based on the Regesta Imperii corpus, which contains abstracts of charters issued by medieval European rulers. In order to build this system, we conduct the following two steps: **(i) place prediction**: we design a bootstrapping method that jointly resolves place names of the locations where a charter was created together with place names mentioned in the charter texts. **(ii) semantic linking**: we detect places, entities and their interactions with dependency parsing and named entity recognition and aggregate this information together with the place predictions in a single resource. The final resource spans over almost 1000 years of European medieval history. It contains approximately 180,000 place predictions for charter origin places and place predictions for more than one million medieval entities mentioned inside the charter texts, together with their relations. All code is available online under public license: https://github.com/flipz357/regesta-imperii-to-semgis.

Keywords

spatial linking of historic place names, historic GIS construction, European medieval entities

1. Introduction

Our aim is to facilitate the large-scale, spatially grounded and semantic investigation of European medieval entities and their relations. Working towards this aim, we leverage the Regesta Imperii (RI)[1], a large collection that contains more than 180,000 German abstracts (*regests*) of medieval charters and descriptions of events (such as battles or births) [5, 4]. The charters were issued by Holy Roman emperors, their wives, popes and imperial princes. In addition to a brief summary of the charter, each regest has been assigned the place name and the date of charter creation (if they are known).[2]

An example for such a regest is displayed in Figure 1. It summarizes a charter issued by Konrad I in April 912, while he resided in *Fulda*[3]. The charter records the bestowal of land property upon a monastery and proceeds by describing the locations of the transferred property more closely (the property lies near *Helmershausen, Grabfeld, Hengistdorf*, the place names are highlighted). Here, we would like that such place names are linked to coordinates, to spatially ground the charter, the occurring entities, and their relations. However, this constitutes a complex problem, for several reasons. For example, often a name refers to different

CHR 2020: Workshop on Computational Humanities Research, November 18–20, 2020, Amsterdam, The Netherlands

opitz@cl.uni-heidelberg.de (J. Opitz)
0000-0001-6892-4574 (J. Opitz)

[1] www.regesta-imperii.de

[2] To increase simplicity, from henceforth we allow ourselves to use the terms *regest*, *charter* and *document* interchangeably (even though regests as the summaries of charters only emerged later, from 1829 onwards).

[3] Fulda is a city in today's German federal state *Hesse*.

🇩🇪 *Konrad schenkt dem kloster Fulda unter abt Huoggi [...] 3 königshufen zu Helmershausen im gau Grabfeld und das ehemalige lehen seines vasallen Ramuolt in der mark Hengistdorf [...].*

🇬🇧 *Konrad bestows upon the monastery of Fulda under abbot Huoggi [...] 3 königshufen [ca. 360 acres] to Helmershausen in the district of Grabfeld and the former fiefdom of his vassal Ramuolt in the march Hengistdorf [...]*

Figure 1: Regest (excerpt) summarizing a charter issued by Konrad I (king of East Francia from 911 to 918) in April of year 912 in the location of Fulda and our English translation. blue: entities that are not (merely) places.

locations that are scattered across Europe, and we have to decide upon the correct one (e.g., there are locations in Thuringia and Bavaria carrying the name *Grabfeld*). Additionally, we observe a great spelling variation of place names and outdated historic place names (e.g., *Wirzburg/Wirzeburg/Herbepolis* → *Würzburg*), and some places are simply unidentifiable.

Finally, we may be interested in capturing further semantic content of this regest: e.g., we see that the monastery, at this time, was led by an abbot named *Huoggi* and Konrad I had a liege named *Ramuolt*, once the owner of a fiefdom in the place denoted by *Hengistdorf*.

We believe that aggregating such geo-spatial and semantic information on a large scale, for hundreds of thousands of these regests, would allow us to tap on new means for statistical investigation of the middle ages. Among other application scenarios, such a resource could prove valuable for *historic itinerary research*, a branch that determines the influence and reach of historic entities. For instance, let us assume that we were capable of accurately projecting the place names found in the RI onto maps. This would enable us to assess, e.g., times where an emperor traveled far and times where they preferred to stay in one place. Yet, we would not know, *why* they traveled to certain places or *what* they did there (at least, not without massive manual effort). If we knew these things all together, we could answer questions such as *did an emperor interact with important entities over longer distances?* Or *for which actions did the emperor chose to travel more far?* Finally, if we can assess such statistics for a particular emperor, we may also investigate patterns that generalize across multiple emperors. This can be viewed as *emperor profiling* [29] and investigates questions like, e.g., *which emperors are similar to each other (or differ from each other) with respect to their ruling habits?*

The remainder of this paper is structured as follows. First, in Section 2, we describe our task setup more precisely, and propose a method that targets the joint resolution of all place names detected in the RI. In Section 3, we evaluate the quality of the resolutions under different configurations of our method. We proceed by performing example data analyses and outlining a workflow that aggregates the method's result, together with semantic relations between entities, in a single knowledge graph structure (Section 4). Finally, in Section 5, we discuss some background, related work and specific limitations of our approach.

2. Task and method

2.1. Preliminaries

Notation Let $\mathcal{R} = \{1, ..., T\}$ denote a set of indices, or 'time stamps', where each index t identifies a unique regest. Let $\mathcal{V}$ be our place name vocabulary and $name_t \in \mathcal{V} \cup \{\varnothing\}$ the place name, where the charter t was issued ($\varnothing$ represents the unknown place name). Additionally, let $names_t \subseteq \mathcal{V}$ be the set of place names detected in regest t's textual content. And $C : \mathcal{V} \rightarrow 2^{\mathcal{P}}$ a function that returns a set of geo-spatial candidate points $\subseteq \mathcal{P}$ for a place name. Then, $cost : \mathcal{P} \times \mathcal{P} \times \Phi \rightarrow \mathbb{R}_+$ is a function which calculates the cost of traveling from one place to another, Φ denoting some arbitrary set of contexts that may or may not inform the calculation. Let us now consider the

Task. For every regest t

1. Predict the location of charter origin: we need to make a place prediction $p_{t,0} \in \mathcal{P}$ for $name_t$.
2. Find and resolve the place names mentioned in charter t's text: We need to make m_t place predictions $(p_{t,1}, ..., p_{t,m_t}) \in \mathcal{P}^{m_t}$, corresponding to the m_t place names detected in charter t's text.
3. Determine the semantic functions of the resolved places and entities in the context, e.g., *what* exactly did the emperor bestow onto *whom*?

With these three ingredients, we will be capable of constructing a first instance of our desired semantic tempo-spatial medieval information system.

2.2. Assumptions and setup

To make the predictions as informed as possible, we introduce two assumptions. Then we will describe the geo-data base that we use and the traveling cost function.

Assumption I, emperors chose routes that minimize traveling cost An emperor that issues k charters travels from the place at time-step t to the place at time step $t + k$. Now, for instance, consider that an emperor releases a charter in *Vienna* at t, then he releases one in the place denoted by the ambiguous name *Neustadt* at $t + 1$ and then again in *Vienna* at $t + 2$. Then we may confidently say that the *Neustadt* in question denotes the place also known as *Wiener Neustadt*, 45 km south of *Vienna* – as opposed to other places denoted by this name (e.g., *Neustadt in the Palatinate*, 700km west of *Vienna*). Generally, the traveling cost may not only incorporate the mere distance, but also other features that, e.g., reflect the significance of a city. For example, if the emperor traveled to *Prag*, then it is likely that he traveled to *Prague*, the capital of today's Czech Republic, a significant medieval place (instead of another location named *Prag* that just happens to lie more close (distance-wise) to his current stay.)

Assumption II, locations mentioned in a charter are clustered together One or several place names may be mentioned in the charter text (see Figure 1). Often, when an emperor traveled to a location, local entities interacted with the emperor, or his court. From this, we can conclude that locations that are referred to in the charter text can inform us to better resolve other place names in the text (and the name of the charter origin place), and vice versa.

Geo-data base We need a collection of places onto which we may project the place names found in the RI. For this, we use geonames[4] as our primary source, mainly for two reasons. First, it is freely accessible (CC BY 4.0). Second, it covers a great variety of places and contains

[4] https://www.geonames.org/

over eleven million place names. Among these names are also historic or antiquated ones, a property which makes it specifically suited for our task.

Cost function We have to define the cost from traveling from one place p to another p'. Here, we use a similar formula as prior work [23], but with an additional parameter $\mathcal{H} \subseteq \mathcal{P}$, which denotes a set with 'helper'-places as contexts (i.e., $\mathcal{H} \in \Phi$) that we can use to better inform the cost of traveling to p':

$$cost(p, p', \mathcal{H}) = \frac{d(p, p') + \lambda_1 |\mathcal{H}|^{-1} \sum_{p'' \in \mathcal{H}} d(p', p'')}{1 + \lambda_2 y^{name(p')} + \lambda_3 log_{1000}(pop(p'))}. \tag{1}$$

Here, $d(p, p')$ denotes the (vincenty) distance from p to p'. The greater the distance, the greater the cost. However, the cost also depends on other factors, that can be weighted with coefficients λ_2, λ_3. It decreases further when $y^{name(p')} \in [0, 1]$ increases, which is the string overlap of the place name mentioned in the regest with a place name of place p' in the geo data-base. Furthermore, we incorporate $pop(p')$, which is the population count of place p': it is less costly to visit densely populated places. We extract the population count from geonames, if given. On one hand, this seems problematic, since the true historic population counts are likely not well reflected by the counts stated in geonames. On the other hand, however, introducing such a bias is justified since many significant places of the middle ages are still today quite populated (e.g., Nuremberg, Rome, Prague, etc.), and there are no resources known to us that contain reliable historic population counts covering the temporal space of this work.

Finally, if p' lies close to the 'helper'-points in $\mathcal{H}$, the cost of traveling from p to p' will be reduced. By default, we set $\mathcal{H} = \{\varnothing\}$, but we will later use this parameter to interleave the resolution of place names detected in charter texts with the resolution of charter origin places.

2.3. Resolving places

Now, we will describe a method for predicting the places of charter origin and the places referred to in the charter's textual content.

Resolving emperor itineraries When determining the route of an emperor, we can project all possible routes, onto a (temporally ordered) directed acyclic graph (DAG). In this graph, $nodes(t) = C(name_t) = \{p \in C(name_t) \subseteq \mathcal{P}\}$ are candidate places for $name_t$ that are connected with weighted edges (travel cost) to all $nodes(t + 1) = C(name^{t+1}) = \{p \in C(name^{t+1}) \subseteq \mathcal{P}\}$, which are the candidate places for $name^{t+1}$. This enables us to determine the most suitable route in this type of DAG optimally with a simple search algorithm of linear complexity. Let this function be denoted by

$$resolveItinerary(X^{itinerary}, \mathbf{H}) = \{p_t^{\star} \in \mathcal{P}\}_{t=1}^{T} = Y^{itinerary}, \tag{2}$$

where $X^{itinerary} = \{name_1, ..., name_T\}$ are the place names to be resolved and $\mathbf{H} = \{\mathcal{H}_1, ..., \mathcal{H}_{\mathcal{T}}\}$ are sets with associated helper places (in the simplest case, $\mathbf{H}_t = \{\varnothing\}\ \forall t$).

Resolving places mentioned in the charter text We want to resolve all place names mentioned in the textual content of the regests. Consider any regest text that contains k place names $\{name_1, ..., name_k\}$, then we want to obtain a set $P^{\star} = \{p_i^{\star} \in \mathcal{P}\}_{i=1}^{k} \subseteq \mathcal{P}$ that minimizes the perimeter of all possible k-poligons that emerge when connecting the candidates of every place name to the candidates of every other place name. Let this function be denoted by

$$resolveTextPlaces(X^{text}, \mathbf{H}) = \{P_t^{\star} \subseteq \mathcal{P}\}_{t=1}^{T} = Y^{text}. \tag{3}$$

Here, $X^{text} = \{names_1, ..., names_T\}$ are the sets of place names to be resolved in every time step and $\mathbf{H} = \{\mathcal{H}_1, ..., \mathcal{H}_T\}$ are sets with associated helper places. By default, we do not assume any auxiliary context information available and set $\mathbf{H}_t = \{\varnothing\}\ \forall t)$.

Solving Eq. 3 is equivalent to solving the Steiner-tree problem when connecting every candidate from a specific place name to an extra terminal node *placename*. However, this problem is NP hard and becomes intractable with increasing k. We investigate two solution methods, (i) a Steiner tree approximation and (ii) a simple hill-climbing method. In an experiment (described more closely in Appendix A), we observe that both outperform the random base line by large margins and yield solutions that are similar. Furthermore, we find that the computational cost of the hill-climber is smaller, and thus we decide to use it as our main solver for Eq. 3.

2.4. Jointly resolving emperor itineraries and places mentioned in the charter texts via boot-strapping

At this point, we have described two methods for separate use: one method resolves the places mentioned as charter origin with a route search and the other method resolves the places mentioned in the text. However, the helper places **H** can be considered as a link that allows us to interleave Eq. 2 and Eq. 3, resulting in a method that jointly resolves places of charter origin and places mentioned in the charter text. This could prove valuable, for the following reason (Assumption II): if we knew what places are predicted in the charter text, we could better predict the origin of charter location. And, vice versa, if we knew the predicted origin of charter creation, we could better resolve the places mentioned in the charter text. More precisely, to incorporate this information, we design a boot-strapping algorithm.

Algorithm 1 Boot-strap for resolving place names in the RI

```
1: $X^{itinerary} \leftarrow$ list with place names of charter origin
2: $X^{text} \leftarrow$ list of lists with place names in charter texts
3: $Y^{itinerary} \leftarrow$ empty, to collect charter origin place predictions
4: $Y^{text} \leftarrow$ empty, to collect charter text place predictions
5: for iter in iterations do
6:     $Y^{itinerary} \leftarrow resolveItinerary(X^{itinerary}, Y^{text})$        ▷ resolve emperor itinerary, Eq. 2
7:     $Y^{text} \leftarrow resolveTextPlaces(X^{text}, Y^{itinerary})$        ▷ resolve charter text places, Eq. 3
8: return $Y^{itinerary}, Y^{text}$
```

It is described in Alg. 1 and outlined in Figure 2. In the first iteration (line 5, Alg 1), we resolve the places of charter origin with the shortest-path search $resolveItinerary(X^{itinerary}, \{\varnothing\})$ (Eq. 2), obtaining the result $Y^{itinerary}$, which contains the predicted coordinates for all traveling steps (line 6, Alg 1; right-bottom to top-right in Figure 2). In the next step (line 7, Alg 1), we predict the places for all the place names mentioned in the charter texts using $Y^{itinerary}$ as helper places: $resolveTextPlaces(X^{text}, Y^{itinerary})$ (Eq. 3). I.e., these predictions are in-

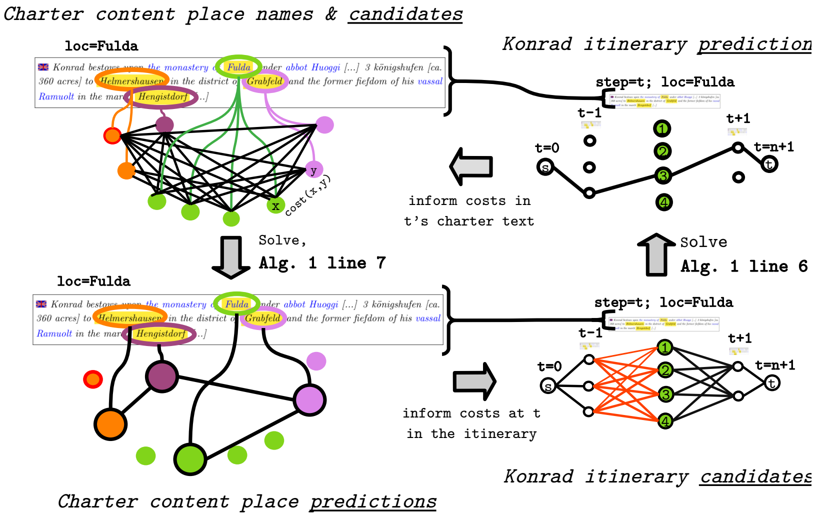

Charter content place names & candidates
loc=Fulda
Konrad bestows upon the monastery of Fulda under abbot Huoggi [...] 3 königshufen [ca. 360 acres] to Helmershausen in the district of Grabfeld and the former fiefdom of his vassal Ramuolt in the mark Hengistdorf [...]
y
x
cost(x,y)
Konrad itinerary prediction
step=t; loc=Fulda
t-1
t+1
t=0
t=n+1
s
t
1
2
3
4
inform costs in t's charter text
Solve, Alg. 1 line 7
Solve Alg. 1 line 6
loc=Fulda
step=t; loc=Fulda
inform costs at t in the itinerary
Konrad itinerary candidates
Charter content place predictions

Figure 2: Bootstrap outline.

Table 1
Two examples of significant medieval places and (some of) their various spelling variations in the RI.

name today	variations
Innsbruck	*Innsbruck A. Urk* | *Innsbrnck* | *Innsbr* | *Innsbrucker* | *Innspruck* | *Innsprugae* | *Innsprugg Innsprugk* | *Innsprugkh* | *Innspruk* | *Ynnsprugg Sambstag*| *Ynnsprugg Sonntag*| *Ynnsprugg Suntag* | *Ynnspruggk* | *Ynnsprugk*| *Ynnsprugkh*| *Ynnspruk*| *Ynnsprukh*| *Ynnspurgk* (...)
Frankfurt	*Franckford* | *Franckfort*| *Franckfurd*| *Franckfurt*| *Franckfurta*| *Franckfurter* *Franckfurth*| *Franckinford*| *Franckinfort*| *Franckinfurt* *Stadt Frankfurt Main*| *Stadt Frankfurt Mayn*| *Frankenvord* | *Frankenvorde* | *Franconofurd* (...)

formed with $Y^{itinerary}$ because we update the traveling costs using $Y^{itinerary}$ as helper places **H** (see also top-right to top-left to left-bottom in Figure 2). This concludes the first iteration.

2.5. Candidate extraction

In order to populate the solution space, we need to gather sets of candidates for place names. In our case, place names are either place names of charter origin or, in the charter text, place names that were labeled '*LOC*' by the named entity recognition (NER) program. Additionally, we consider parts of named entities (labeled '*PER*' by the NER program) that associate them with places. For instance, the phrase x *of* y (in German: x *von/v./de* y) is a common construction in the charter texts. In such cases, x is a person name and y is a place name.

Table 2
Candidate statistics.

	standard	+SCE
unique place names	135,671	135,671
avg. places per name	10.1	14.2
median (50th percentile)	3.0	5.0
75th percentile	9.0	13.0
95th percentile	41.0	58.0

Table 3
Parameterization summary.

parameter	description	parameterization
$\mathcal{P}$	geo data-base	geonames
$C(name)$	place candidates for a place name	$\{p \mid p \in \mathcal{P}, LevR(name, name(p)) = y^{name}\}$
y^{name}	max. string overlap to any name in $\mathcal{P}$	$max\{LevR(name, name(p)) \mid p \in \mathcal{P}\}$
$LevR$	string overlap measure	Levenshtein overlap ratio of two strings [36]
$cost(p, p', (\cdot))$	cost for traveling from p to p'	Eq. 1
$d(p, p')$	geo-spatial distance function	vincenty distance between two places [34]
$pop(p)$	population count of a place	$max(1, geonamesPopulationCount(p))$
λ_1	coef in Eq. 1 for helper place cost	0.25
λ_2	coef in Eq. 1 for string overlap	1
λ_3	coef in Eq. 1 for population	1

String search for candidate gathering If a place name *name* exactly matches a place name contained in our data base, we gather $n \geq 1$ candidate points. Here, n is determined by the amount of places that are associated with *name* (often, two or more distinct locations carry the same name, or are associated with the same name). However, many place names cannot be expected to match exactly with a place name in our geo-data-base, due to a great spelling variation of place names in the data (Table 1). To solve this problem, for any place name *name* which we do not find in geonames, we calculate the Levenshtein string overlap ratio $LevR$ of *name* with every place name $name'$ contained in geonames and denote the maximum overlap by $y^{name} \in [0, 1]$. Finally, we gather as candidates all points that are associated with all names $name'$ where $LevR(name, name') = y^{name}$. By this move, we are able to map spelling variations such as *Franckford* or *Innsbruk* to their today's names *Frankfurt (am Main)* or *Innsbruck*. Statistics of the candidates gathered are displayed in Table 2 (left column).

Simple candidate extension (SCE) We observe that some place names consist of longer descriptions, made up of several multi-word tokens. For example, "*im Fluss Saleph (heute: Gök-su) oberhalb von Seleucia (Silifke)*" (translation: in the river Saleph (today Gök-su), north of Seleucia (Silifke)). Therefore, in our primary method configuration, we use a simple candidate extension mechanism (short: SCE) to cover such place names. More precisely, we use each single token that starts with an upper-case character and look if there is a direct hit in the geonames data-base. If yes, we add the corresponding places as candidates. Statistics of the candidates gathered using SCE are displayed in Table 2 (right column).

2.6. Parameter summary

Our approach depends on several (hyper-) parameters, that were described in the previous parts of the paper. Here, we will state a summary, of the most important ones (Table 3).

We use geonames as our geo data-base $\mathcal{P}$. $C(name)$ returns candidates for a place name query by returning places that are assigned a name (in geonames) that has maximum (y^{name}) Levenshtein string overlap ratio *LevR* with the query. The context-dependent cost function (Eq. 1) has been described in detail in Section 2.2. As distance function we use the vincenty distance that calculates the distance between two points on the surface of a spheroid. Finally, *pop* returns the population count of a place, if it is stated in geonames, and 1 otherwise.

2.7. Post-processing

By now, we have predicted the places of charter origin and the places found in the charter texts, on the regest instance level. This has the advantage, that the same name can refer to different places at different times. E.g., there were multiple castles in Germany, Switzerland and Austria that go by the name *Ehrenfels*, and our resolutions may flexibly refer to one castle *Ehrenfels* at one specific mention and at a different *Ehrenfels* at another, depending on the context. However, we may also be interested to develop a mapping that maps every unique name to an 'unequivocal' point of reference. Thus, at the cost of flexibility, we may increase the average accuracy of our predictions, because a large amount of place names mentioned as charter origin places are indeed quite unequivocal (e.g., *Rome*, *Nuremberg*, *Cologne*, etc.). To achieve this mapping, we use two simple mechanisms.

Majority vote For every unique place name, we collect all predicted geoname-ids and assign to every mention the most-frequent id. We perform this step separately for the charter origins and the places referred to in the charter text, learning *two mappings* in total, one between place names mentioned as charter origin and coordinates and another one between place names mentioned in the text and coordinates.

Joint majority vote In this setup, we compute the most-frequent geoname ids not separately, but over our full predictions, to learn *one mapping* between place names and coordinates.

Unknown place interpolation When predicting the itineraries, we interpolate missing places of unknown place name by inserting the place at which the issuer resided lastly. When predicting places inside the charter texts, such a simple yet effective solution is not available. To interpolate these places, we use the mean coordinates of all other places in the charter text, that we were able to resolve.

3. Evaluation

By executing our method, we have obtained, for any regest t, place resolutions of the place names detected in the text by our NER system $(p_{t,1}, ..., p_{t,m_t}) \in \mathcal{P}^{m_t}$ and the place name from whence the corresponding charter was issued, denoted by $p_0^i \in \mathcal{P}$. Furthermore, by using the majority-vote, we have also obtained a mapping $majority : \mathcal{V} \cup \{unknown\} \rightarrow \mathcal{P} \cup \{\varnothing\}$, that assigns to every place name from our vocabulary an unequivocal point of reference.

Ground truth We evaluate the place predictions for the charter locations against the data set described in [23], which contains manual resolutions of most place names that are stated as charter origins. Note, that the manual resolution took place on a place name level and thus cannot account for instances where the same place name, under different circumstances, may refer to different places. Nevertheless, we use this data to create, for the majority of regests, a 'gold' resolution, by looking up the place that was assigned by the human to its place name. We denote this gold standard as a function $gold : \mathcal{V} \cup \{unknown\} \rightarrow \mathcal{P} \cup \{\varnothing\}$, which returns the assigned place if the place name is resolved in the human data or $\varnothing$ otherwise. Furthermore, we observe that the baseline [23] lacks prediction of some place names. Therefore, in our evaluation setup, we only consider instances where we have the gold prediction as well as the prediction of the baseline.

Micro and macro metrics We use two different ways of conducting the evaluation, either calculating micro- or macro errors. 'Micro' is calculated over all regest instances and thus will be biased towards resolutions of frequently mentioned place names.[5] Therefore, we also calculate a 'macro' error over the place name vocabulary. In the following, let $error(p, p') \equiv d(p, p')$ be the vincenty distance (km) and $G = \{t \mid t \in \mathcal{T},\ gold(name_t) \neq \varnothing\}$ denote pointers to charters where the charter origin name has a gold resolution (analogously, for any t, we can retrieve $G_t^{text} = \{j \mid j \in 1...m_t, gold(names_{t,j}) \neq \varnothing\}$, which identifies all places in the text of charter t for which we have a manual prediction accessible. Then, we can define the

$$Errors_{Micro}^{Itinerary} = \Big\{d(gold(name_t), p_{t,0}) \mid t \in G\Big\}. \tag{4}$$

To evaluate the text-place predictions, we use

$$Errors_{Micro}^{Text} = \Big\{d(gold(names_{t,j}), p_{t,j}) \mid t \in \mathcal{T}, j \in G_t^{text}\Big\}. \tag{5}$$

Similarly, we can define the macro errors as

$$Errors_{Macro}^{Itinerary} = \Big\{d(gold(name), majority(name)) \mid name \in names(G)\Big\}, \tag{6}$$

$$Errors_{Macro}^{Text} = \Big\{d(gold(name), majority(name)) \mid name \in \bigcup_{t=1}^{T} names(G_t^{text})\Big\}. \tag{7}$$

Results The main results are displayed in Table 4. We make several observations. Our combined system, either with or without SCE (simple candidate extension), performs considerably and consistently better in most scenarios than all baselines. Specifically, we find that the macro error is reduced. When predicting the itineraries, our best method outperforms the best baseline with Δ -47.7 km (265.8 vs. 219.1) and the random baseline by Δ -196.2 km (415.3 vs. 219.2). Notably, the median error is reduced to 12.2 km on the macro level (Opitz et al. [23]: 41 km, random: 176 km). This means that 50% of our macro level location predictions deviate less than 12.2 km from the gold coordinates.

A similar picture emerges when evaluating the predictions for the text error (here, we only have the random baseline since Opitz et al. [23] does not predict place names that occur in the texts). The random baseline is outperformed by our complete system by Δ -143.4 km (mean error), -12.5 km (median error) and -334.9 km (85th precentile).

[5] Because many frequently mentioned places tend to be of low ambiguity (Rome, Nuremberg, Frankfurt, etc.), we expect that they are resolved with higher accuracy

Table 4
Main results. Columns labeled with *50th/85th* display the *median/85th percentile* error. The error unit is always kilometers.

		Itinerary						Text					
		Micro errors			Macro errors			Micro errors			Macro errors		
	approach	mean	50th	85th	mean	50th	85th	mean	50th	85th	mean	50th	85th
baseline	random	241.8	2.8	632.4	-	-	-	202.85	2.7	478.9	-	-	-
	random+maj.	264.8	1.5	714.2	418.8	176.4	965.1	175.4	**1.1**	405.6	299.7	14.8	698.6
	random+joint maj.	233.8	1.5	714.2	415.3	166.7	967.7	172.7	1.1	405.6	296.4	14.8	679.2
	Opitz et al. [23]	112.9	**0.0**	238.4	-	-	-	-	-	-	-	-	-
	Opitz et al. [23]+center	81.7	**0.0**	82.8	265.8	41.0	606.1	-	-	-	-	-	-
this work	basic	141.1	2.7	360.4	-	-	-	107.66	2.6	194.8	-	-	-
	bootstrap	108.6	2.5	212.9	-	-	-	103.6	2.6	184.7	-	-	-
	basic+maj.	120.6	2.7	359.7	263.8	29.5	605.4	65.1	2.2	**43.7**	164.9	2.2	363.9
	bootstrap+maj.	82.7	2.3	64.3	252.8	22.9	585.7	64.3	2.2	**43.7**	164.4	2.1	372.2
	basic+joint maj.	85.0	2.0	62.3	257.9	23.6	599.0	65.1	2.1	48.4	160.8	**1.9**	362.7
	bootstrap+joint maj.	**80.5**	2.3	**47.1**	249.7	21.9	583.9	**63.4**	2.3	**43.7**	156.7	**1.9**	354.7
	bootstrap+joint maj. +SCE	90.9	2.2	143.2	**219.1**	**12.2**	**518.4**	65.4	2.2	44.9	**153.0**	2.3	**344.3**

Table 5
Detailed accuracy assessment.

	method	% < n km (higher is better)						% > n km (lower is better)			
		micro			macro			micro		macro	
		n=15	n=25	n=50	n=15	n=25	n=50	n=250	n=750	n=250	n=750
baseline	random	58.4	59.7	61.3	-	-	-	28.3	11.4	-	-
	random+maj.	63.2	63.7	64.8	37.9	39.4	42.0	26.1	13.7	44.6	20.3
	random+joint maj.	63.2	63.7	64.8	37.9	39.4	42.0	26.1	13.7	44.6	20.3
	Opitz et al. [23]	68.8	73.3	78.9	-	-	-	14.7	5.4	-	-
	Opitz et al. [23]+center	79.1	**80.2**	84.5	44.2	47.2	51.1	10.8	3.4	33.3	10.8
this work	basic	66.6	70.3	75.1	-	-	-	18.9	6.0	-	-
	bootstrapping	70.2	74.8	81.3	-	-	-	14.4	4.2	-	-
	basic+maj.	65.5	70.6	77.6	46.6	49.3	52.9	17.7	3.5	33.4	10.9
	bootstrapping+maj.	66.6	70.3	75.1	47.6	50.4	54.0	11.1	3.6	32.3	10.7
	basic+joint maj.	65.5	70.6	77.6	46.6	49.3	52.9	17.7	3.5	33.4	10.9
	bootstrap+joint maj.	72.6	78.0	**85.2**	48.3	51.2	54.9	**10.7**	2.6	31.2	10.3
	bootstrap+joint maj. +SCE	70.7	76.2	83.6	**51.5**	**55.7**	**60.1**	12.7	**2.4**	**28.0**	**8.8**

On the micro-level, the best baseline predicts the itineraries better than our simpler baselines. A possible explanation lies in the fact that our method allows larger candidate sets due to the inexact place name matching and thus takes a greater risk of a wrong guess. More specifically, in the case where there is no exact place name match, Opitz et al. [23] uses the place of the last prediction. This is a strong guess, since it happened quite frequently that charters were released repeatedly from the exact same place – on the macro-level this type of guess loses its predictiveness. This may also explain the worse performance of our full method (last row) compared with the same model without SCE in most cases of micro error assessment (e.g., penultimate row, left column of Table 4: mean error Δ +10.4 km mean error). In other words, SCE tends to reduces the macro error, increases the micro error.

Accuracies @kilometers Now, we want to assess the proportion of places that are resolved with certain accuracy levels: what percentage of places is resolved with a deviation of *less than 15km/25km/50km ('good predictions')* or with a deviation of *more than 250km/750km ('bad predictions').*

The results are shown in Table 5. Our method is outperformed by the baseline in micro error for places that lie closer than 15 and 25 km to the gold place. Here, the method of Opitz et al. [23] resolves 79.1% (respectively 80.2%) of the places with a deviation of less than 25 km, whereas our (best) setup achieves only 70.7% (respectively 76.2%). This is different on the macro-level, where the simple guess of inserting the last known place as the prediction loses its predictiveness. Here, our best setup consistently performs better than all baselines with an increase of 3.2, 8.5 and 9 percentage points over the respective best baseline result.

Our system also appears to make less 'bad predictions'. On the micro-level, 3.4% of predictions from the best baseline deviate more than 750 km from the gold coordinates, whereas ours achieves a lower ratio of only 2.4%. On the macro-level, 33.3% of the best baseline's predictions deviate more than 250km from the gold coordinates, which is reduced by our system by 5.3 percentage points (to 28% of predictions).

Accuracies @ages Now, we want to assess the error of our predictions with respect to different ages. To this aim, we create bins of resolutions that span 20 years each. For any such bin, we calculate the avg. Δ in kilometers versus the baseline system of Opitz et al. [23](+center). More precisely, this quantity is defined, for any *bin* as

$$mean(Errors_{(\cdot)}^{(\cdot),ours,bin}) - mean(Errors_{(\cdot)}^{(\cdot),baseline,bin}) \quad (8)$$

This means that in times where Eq. 8 yields a negative result, our method outperforms the baseline, whereas in times in which this quantity is positive, the baseline outperforms our system.

The results are shown in Figure 3. In this Figure, *Cumulative* indicates Δ sums that have accumulated up to a certain time (decrease: our method works better; increase: baseline method works better). It indicates that (i) our method works better than the baseline, both in terms of micro and macro error, over most of the considered time bins (blue dotted line and green star line), especially when considering the earlier middle ages (700-1000 CE) and the later middle ages (1200-1350). (ii), when considering the macro error, our method appears to overall outperform the baseline: the red diamond line, indicating the cumulative difference to the baseline, is decreasing almost everywhere, except in 1000 - 1100 CE and (marginally) in 1400 - 1500 CE.

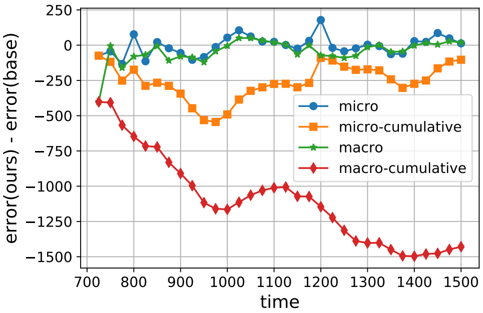
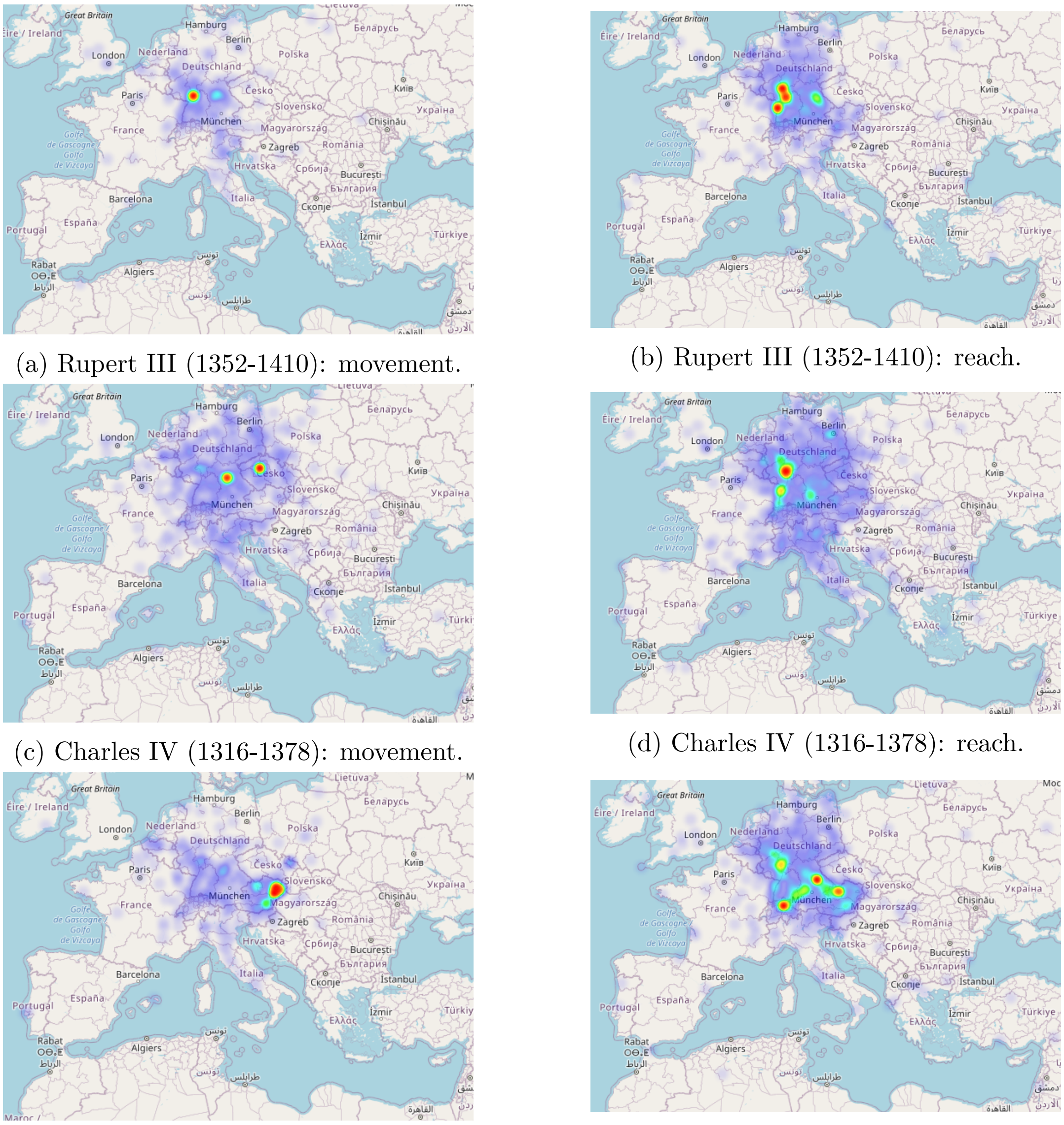


Figure 3: Plotting the error vs. the baseline (Eq. 8) over bins of 20 years.

4. Data analyses and an outlook on knowledge graph construction

4.1. Data analysis

Spatial movement and interactions of Rupert III, Charles IV and Frederick III We assess three powerful medieval rulers with respect to their geo-spatial movement and interaction: Rupert III (1352 – 1410), Charles IV (1316 – 1378) and Frederick III (1415 - 19 August



(a) Rupert III (1352-1410): movement.

(b) Rupert III (1352-1410): reach.

(c) Charles IV (1316-1378): movement.

(d) Charles IV (1316-1378): reach.

(e) Frederick III (1415-1493): movement.

(f) Frederick III (1415-1493): reach.

Figure 4: Movement (left) and reach (right) of three influential emperors as estimated by the method. Most frequent actions: **(a,b)**: bestows (332), presents (164), commands (128), confirms (116). **(c,d)**: commands (492), confirms (482), writes (223), bestows (214). **(e,f)**: bestows (506), confirms (390), commands (296), gives (270).

1493). Charles and Frederick were Holy Roman emperors, while Rupert III reigned as king of Germany. Thus, Rupert III resided slightly lower in the hierarchy, which we find being reflected in a smaller action radius (Figure 4a), that focuses on West Germany. Still, his reach to other entities covers almost all of Germany, notably also Strasbourg (Figure 4b, bottom-left red spot), with whom he formed an alliance in 1408. His most frequent actions were rather ‘benevolent’ ones: *bestowal* (332 times) and *presents* (164 times).

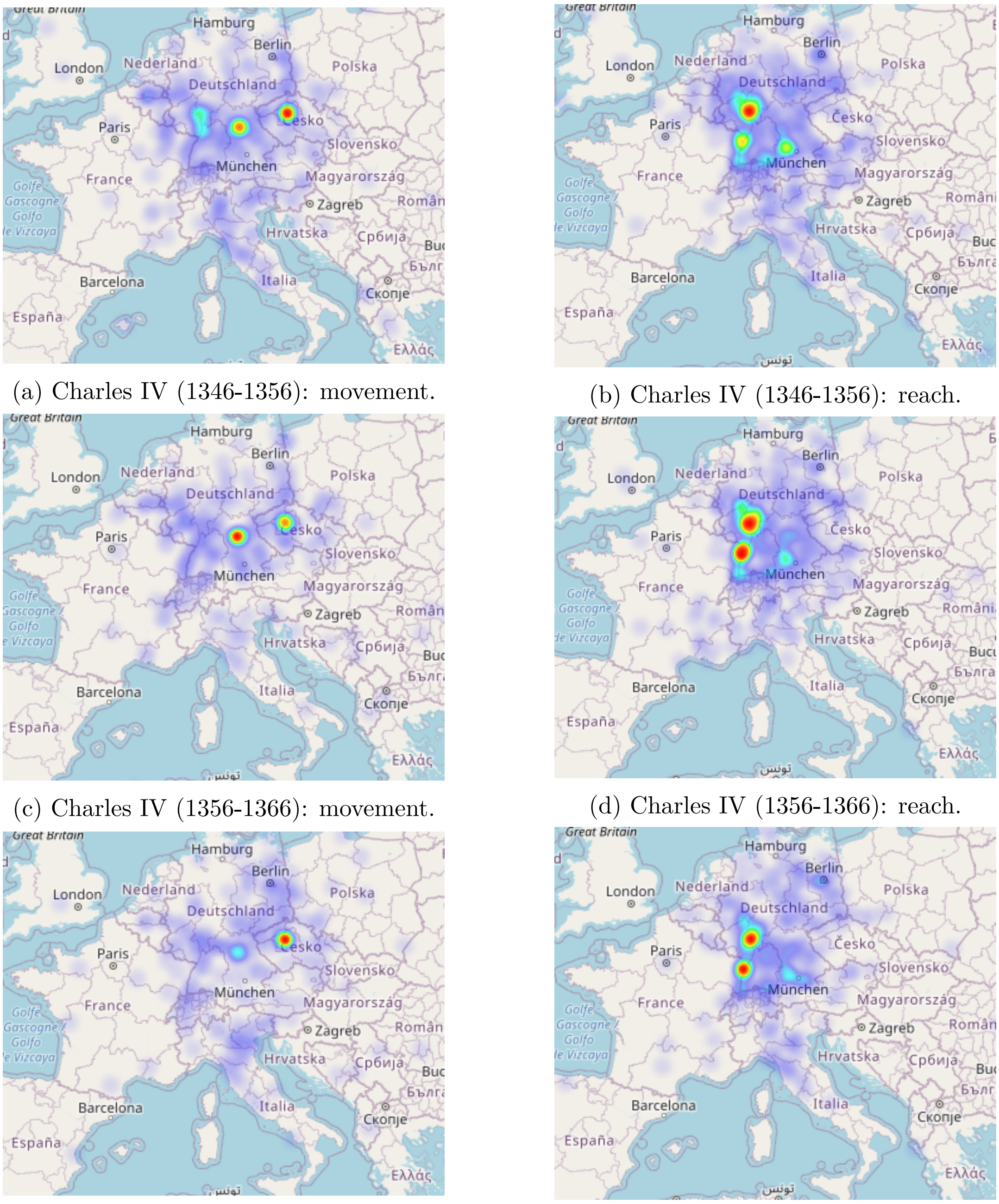
The actions of Charles IV seem slightly less ‘benevolent’, he most frequently *confirms* (482 times) or *commands* (128 times). His action radius is larger than that of Rupert III and focuses on Prague and Bavaria (Figure 4c). The radius of his reach is also of great extent, covering all of Germany and also parts of Italy (Figure 4d). Frederick III shifts the center of his action away from Prague, again to West Germany, but also (and foremost) Austria (Figure 4e). He appears to be not as concerned with Italian businesses as it was the case with Charles IV, but focuses on the Alps, more precisely, Switzerland and Austria (Figure 4f).

Reach of Charles IV over three decades of his reign Charles IV was born 14 May 1316 and crowned King of the Romans in 1346. Figure 5 displays some of his spatial and interaction patterns from 1346 onwards, over the course of three decades. For example, we see that an important spot during the earlier days of reign was Prague. However, in the middle days of his reign, Nuremberg was a more frequently visited place. In the later days of reign, Prague again appears to become a highly frequented residence. However, we also see that Prague was never the spatial center of his *reach*. His main focus of interaction lied in the west – according to our predictions specifically on the cities Mainz and Strasbourg (Figures 5b, 5d, 5f). Another facet of his reign is reflected in the movement and reach with respect to Italy. Namely, in 1354, Charles traveled to Italy to receive the Imperial Crown. However, he returned quite immediately, essentially abandoning his imperial rights in Italy [25]. Our analysis indicates that this may have also resulted in a loss of influence in Italy (5a vs. 5c and 5b vs. 5d). His second travel to Italy (1368) is also indicated in the map and seems to come with a partial restoration of his Italian reach, especially in Liguria (5d vs. 5f).

KG outline It is possible to aggregate the results of this work in a single knowledge graph (KG). Here, we want to give a brief outline on how this could be achieved. The technical implementation, investigation and, perhaps more importantly, the assessment of the usefulness of such a large (and noisy) KG for history studies, will have to be studied in future work.

Let our set of vertices V contain indices that refer to (i) regests as events and (ii) medieval entities, either emperors who issued charters or entities mentioned in the charter text. Then we have edges $E \subseteq V \times V \times L$, where L is a set of edge labels, which is used to represent semantic relationships between the nodes. The key result of this work can be modeled with a dedicated *:predictedPlaceId* edge: this edge connects an entity (e.g., an event or a person) to a geo-spatial entity. If this edge connects a place node with a regest node, this means that the place node contains the predicted place of charter creation. Otherwise, such an edge associates a place node with a medieval entity that we detected in the text of the charter.

The charter id-nodes may be connected to the entities mentioned in the text with shallow semantic or syntactic relations. For example, we may only extract entities that are subject, dative object or accusative object. This would cover many actions found in the RI, such as *bestows, commands, gives, confirms, writes*, etc. I.e., we would like to capture the question *who bestows/commands/gives... what (on)to whom?* along with the predicted locations where the specific entities (who/what/whom) resided. E.g., consider in charter i, issued by emperor *issuer* of year *year* with predicted place p a detected entity e and its associated place p', where the e occurs in the text with the dependency relations *bestows* → *dativeObject* → *Entity*. Then, we may insert five triples into the graph: $< i,$ `bestows:subject,` $issuer >$, $< i,$ `year,` $year >$, $< i,$ `predictedPlaceId,` $p >$, $< i,$ `bestows:dativeObject,` $e >$ and $< e,$ `predictedPlaceId,` $p' >$.



(a) Charles IV (1346-1356): movement.

(b) Charles IV (1346-1356): reach.

(c) Charles IV (1356-1366): movement.

(d) Charles IV (1356-1366): reach.

(e) Charles IV (1366-1376): movement.

(f) Charles IV (1366-1376): reach.

Figure 5: Movement and reach of Charles IV in three different decades of his reign.

5. Limitations, related work and background

5.1. Limitations

Working with the RI means working in an extreme environment. This is due to several factors, such as, i.a., the great word and place name spelling variation (e.g., Table 1), unknown places (and unknown place names). Moreover, the large spatial dimension (bridging Europe and its Asian and African crossroads) and temporal dimension (800 years) points to a large and complicated entity universe, containing some persisting entities (e.g., the city *Rome*) and many others emerging and vanishing (e.g., persons). This implies that there are several limitations of this work, here we will give an outline of the most obvious ones.

Assumptions and heuristics First, to allow resolution of place names in the corpus via our method, we used *assumptions and approximations*: e.g., Assumption I and II in Section 2 or the traveling-cost heuristic. It is unclear how well the incorporated assumptions hold in the specific cases. Moreover, the backbone of the traveling cost heuristic is the vincenty distance, which measures the length of a line between two points on a spherical surface. Naturally, it neglects geographical hindrances such as rivers or mountain passes, that may increase the traveling cost. In modern day route planning, such hindrances are taken into account in most applications. However, modern route planning has access to accurate and frequently updated street networks. In this work, we covered a time span of several hundreds of years. Over these centuries streets changed, bridges were built and vanished, etc. In other words, for reconstruction of our medieval places and itineraries, we cannot rely on many of the conveniences of modern route planning. Nevertheless, it may be possible to significantly improve the cost heuristic by incorporating additional contextual information.

Prediction error Our evaluation indicates that there is ample room for improvement of the predictions. While the random baseline is outperformed by large margins, both micro- and macro errors are still considerable. We believe that an (incremental) improvement of the predictions could be achieved by mending some of the weaknesses in our hyper-parameter configurations. For instance, we used geonames as our data-base. While geo-names covers many historical sights, we know only little about its general coverage of places that occur in the RI. This leads to empty candidate sets or candidate sets that do not contain the correct place (in the latter case, our method has no choice but to make an error). To alleviate this issue, we could enrich geonames with places from other data-bases. For example, Simon et al. [32] have created a data base that is dedicated to the representation of historical sights, however, its general coverage is quite small, when compared to geonames. Additionally, encyclopedic sources could help in determining the origin of charters. E.g., from wikipedia articles of emperors we could attempt to automatically retrieve helpful auxiliary signals about their preferred residences and relations to other places.

Finally, we lack knowledge about the performance of the NER system that we used to extract place names from the charter texts and the dependency parser that we used to mine the semantic relations. Since automated language processing systems are known to produce more errors when confronted with historic (mixed) text [26], it could prove valuable to assess (i) whether the most recent NER method[6] provides significantly enhanced performance, or (ii) whether methods that are specifically tailor-cut to historic German text [30, 17]. Alternatively, one may attempt to normalize historic spellings with methods from machine translation [33] before performing named entity recognition and dependency parsing.

Expressiveness of the gold standard As discussed previously, the gold standard only contains name level resolutions. This means that it cannot capture situations where a place name, in different circumstances, can refer to different places. However, it is difficult to assess how often

[6] E.g., *stanza* [28]

such phenomenon occurs. Furthermore, the gold standard may contain some human labeling errors [23], since a considerable amount of the place names are difficult to resolve in general, even for humans.

5.2. Related work and background

Historic itinerary research Historic itinerary research investigates traveling paths of historic entities to determine their influence and reach. So far, most related work has focused on specific itineraries related to one person of interest [13, 10, 8, 7, 20, 21, 11, 1], for example, the study of Senatore [31] assesses the military itinerary of King Ferrante (1458-1465). A general discussion of this research branch can be found in, e.g., Opll [24].

Geo-coding itineraries While our mechanism to resolve places was tailor-cut to the RI, there have also been other works that target the itinerary resolution for other application cases. For example, Blank and Henrich [3, 2] develop a depth-first branch-and-bound algorithm for resolution of historic itineraries from year 1563. Additionally, Moncla et al. [19] propose a spanning tree algorithm to resolve hiking routes and Wen [35] aims at suggesting new routes to travelers, based on texts describing the routes traveled beforehand.

Computational work based on the Regesta Imperii John et al. [12] work towards visualization of itineraries extracted from the RI, where they assume the coordinates as given. This could be valuable to our project. In an ideal application, we could follow the path of an emperor, and, at each step, see connections to other places with whom he interacted at this time (additionally labeled with the semantic type of interaction). Kuczera [15, 16, 14] and Opitz et al. [22] and Born et al. [6] extract semantic graphs from the RI. In all of those cases, it is straightforward to inject our predicted coordinates into these graphs.

Background The investigation of large textual data of historic events may require new means beyond manual analysis. A potential corpus that may be suitable for exploring such new means are the Regesta Imperii, which were intitalized in 1829 by a librarian named Johann Friedrich Böhmer, who began to collect and document the charters issued by medieval European rulers. Since then, many other people have contributed to the project.

For example, let us consider the history of our regest-example discussed throughout this paper, which summarizes a charter issued by Konrad I (Figure 1). This regest was released as part of a book in the 19th century [9]. Later, in the age of computers, it has become part of a growing collection of Unicode documents, freely accessible in an online data base.[7]

Usually, when the RI are used as sources for historic research, a historian selects a small subset of regests from which they believe that they contain relevant information that may aid in answering their research question. E.g., Lenel [18] investigate the history of Verona and Padua in the 13th century and Pope [27] analyses change in relations between rural and urban elites in the 15th and 16th century in upper Germany.

However, the information contained in the RI spreads over almost 1,000 years of European history and the whole European continent. This makes the data also suitable for 'macro'-level research questions, such as the profiling of emperors [29] or other historic entities like

[7] RI I n. 2077, in: Regesta Imperii Online, URL: http://www.regesta-imperii.de/id/0912-04-12_3_0_1_1_0_4456_2077 (accessed July 2020).

cities. Here, one can make out two main branches: (i) research that is explorative and has the potential to mine new research questions or previously unknown patterns, and (ii) targeted information extraction with the goal of answering a specific research question. Under some circumstances it could be appropriate to aim at finding a middle ground between those two branches.

6. Conclusion

We have engaged the problem of automatically resolving place names mentioned in medieval charters. The final resource bridges approximately 800 years of European medieval history and contains more than 1 million predictions for medieval places.

References

[1] J. Barrow. “Way-Stations on English Episcopal Itineraries, 700–1300”. In: *The English Historical Review* 127.526 (2012), pp. 549–565.

[2] D. Blank and A. Henrich. “A depth-first branch-and-bound algorithm for geocoding historic itinerary tables”. In: *Proceedings of the 10th Workshop on Geographic Information Retrieval*. 2016, pp. 1–10.

[3] D. Blank and A. Henrich. “Geocoding place names from historic route descriptions”. In: *Proceedings of the 9th Workshop on Geographic Information Retrieval*. 2015, pp. 1–2.

[4] J. F. Böhmer. *Regesta imperii*. Vol. 1. Wagner, 1908.

[5] J. F. Böhmer. *Regesta Imperii: Die Regesten des Kaiserreichs unter den Karolingern, 751-918*. Vol. 1. Wagner’sche Universitäts-Buchhandlung, 1889.

[6] L. Born, J. Opitz, and V. Nastase. *A knowledge graph from the Regesta Imperii: Construction, visualization and macro-level analyses*. Galway, Ireland, Jan. 1, 2018. URL: https://pdfs.semanticscholar.org/d023/74114678056cf69feb956e2c8d03baa8ebd3.pdf. published.

[7] J.-M. Cauchies. “Widder (Ellen). Itinerar und Politik. Studien zur Reiseherrschaft Karls IV südlich der Alpen.” In: *Revue belge de Philologie et d’Histoire* 77.4 (1999), pp. 1182–1183.

[8] J. F. Edwards et al. “The transport system of medieval England and Wales: A geographical synthesis”. PhD thesis. University of Salford, 1987.

[9] G. für ältere deutsche Geschichtskunde. *Die Urkunden Konrad I., Heinrich I. und Otto I.* Hahnschse Buchhandlung, 1879.

[10] B. P. Hindle. “The road network of medieval England and Wales”. In: *Journal of Historical Geography* 2.3 (1976), pp. 207–221.

[11] R. Hurtienne. “Ein Gelehrter und sein Text. Zur Gesamtedition des Reiseberichts von Dr. Hieronymus Münzer, 1494/95 (Clm 431)”. In: *Erlanger Studien zur Geschichte* 8 (2009), pp. 255–272.

[12] M. John et al. “Interactive Visual Exploration of the Regesta Imperii”. In: *Digital Humanities, Montreal, Canada, August 8-11, 2017* (2017).

[13] H. Krüger. *Das älteste deutsche Routenhandbuch: Jörg Gails" Raissbüchlin"*. Akadem. Druck-U Verlagsanst., 1974.

[14] A. Kuczera. "Die 'Regesta Imperii' im digitalen Zeitalter. Das Regest als Netzwerk von Entitäten". In: *Das Mittelalter* 24.1 (2019), pp. 157–172.

[15] A. Kuczera. *Graphdatenbanken für Historiker. Netzwerke in den Registern der Regesten Kaiser Friedrichs III.* 2015.

[16] A. Kuczera. "Graphentechnologien in den Digitalen Geisteswissenschaften". In: *ABI Technik* 37.3 (2017), pp. 179–196.

[17] K. Labusch, C. Neudecker, and D. Zellhöfer. "BERT for Named Entity Recognition in Contemporary and Historic German". In: *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*. Erlangen, Germany: German Society for Computational Linguistics & Language Technology, 2019, pp. 1–9.

[18] W. Lenel. *Studien Zur Geschichte Paduas und Veronas Im 13. Jahrhundert*. De Gruyter, Incorporated, 1893. ISBN: 9783111133782. URL: https://books.google.de/books?id=3yZaMwAACAAJ.

[19] L. Moncla et al. "Reconstruction of itineraries from annotated text with an informed spanning tree algorithm". In: *International Journal of Geographical Information Science* 30.6 (2016), pp. 1137–1160.

[20] R. Nicholson. "Haunted Itineraries: Reading The Siege of Jerusalem". In: *Exemplaria* 14.2 (2002), pp. 447–484.

[21] A. Oettinger et al. "Making the Myth Real: The Genre of Hebrew Itineraries to the Holy Land in the 12th–13th Century". In: *Folklore: Electronic Journal of Folklore* 36 (2007), pp. 41–66.

[22] J. Opitz, L. Born, and V. Nastase. "Induction of a Large-Scale Knowledge Graph from the Regesta Imperii". In: *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*. Santa Fe, New Mexico: Association for Computational Linguistics, Aug. 2018, pp. 159–168. URL: https://www.aclweb.org/anthology/W18-4518.

[23] J. Opitz et al. "Automatic Reconstruction of Emperor Itineraries from the Regesta Imperii". In: *Proceedings of the 3rd International Conference on Digital Access to Textual Cultural Heritage*. DATeCH2019. Brussels, Belgium: Association for Computing Machinery, 2019, pp. 39–44. ISBN: 9781450371940. DOI: 10.1145/3322905.3322921. URL: https://doi.org/10.1145/3322905.3322921.

[24] F. Opll. "Herrschaft durch Präsenz Gedanken und Bemerkungen zur Itinerarforschung". In: *Mitteilungen des Instituts für österreichische Geschichtsforschung* 117.JG (2009), pp. 12–22.

[25] F. Petrarca and S. Manilius. *Epistolae familiares*. Joannes and Gregorius de Gregoriis, de Forlivio, 1933.

[26] M. Piotrowski. "Natural language processing for historical texts". In: *Synthesis lectures on human language technologies* 5.2 (2012), pp. 1–157.

[27] B. Pope. “Changing Relations Between Rural and Urban Elites Across the Fifteenth and Sixteenth Centuries in Upper Germany”. In: *Die Stadt des Mittelalters an der Schwelle zur Frühen Neuzeit. Beiträge des interdisziplinären (Post-)Doc-Workshop des Trierer Zentrums für Mediävistik im November 2017* (2017), pp. 58–70. URL: https://mittelalter.hypotheses.org/12834.

[28] P. Qi et al. *Stanza: A Python Natural Language Processing Toolkit for Many Human Languages*. 2020.

[29] G. Schwedler et al. *Der Historiker als Profiler: Ueberlegungen zur vergleichenden Analyse spätmittelalterlicher Herrscher*. Böhlau Verlag, 2017.

[30] S. Schweter and J. Baiter. “Towards Robust Named Entity Recognition for Historic German”. In: *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 96–103. DOI: 10.18653/v1/W19-4312. URL: https://www.aclweb.org/anthology/W19-4312.

[31] F. Senatore. *Spazi e tempi della guerra nel Mezzogiorno aragonese: l'itinerario militare di re Ferrante (1458-1465)*. Vol. 10. Carlone, 2002.

[32] R. Simon et al. *Peripleo: a tool for exploring heterogenous data through the dimensions of space and time*. 2016.

[33] G. Tang et al. “An Evaluation of Neural Machine Translation Models on Historical Spelling Normalization”. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 1320–1331. URL: https://www.aclweb.org/anthology/C18-1112.

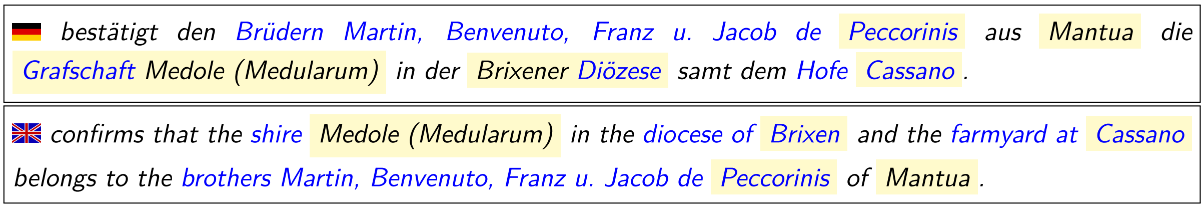
[34] T. Vincenty. “Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations”. In: *Survey review* 23.176 (1975), pp. 88–93.

[35] C. P. Wen. *System for extracting itineraries from plain text documents and its application in online trip planning*. US Patent App. 12/328,768. June 2009.

[36] L. Yujian and L. Bo. “A normalized Levenshtein distance metric”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1091–1095.

Table 6
Performance of different text-place solvers on a subset of 5,000 regests.

method	avg. cost	time
RANDOM	361.4	10s
STEINERAPPROX	266.1	2857s
HILLCLIMBING	259.9	324s



🇩🇪 *bestätigt den Brüdern Martin, Benvenuto, Franz u. Jacob de Peccorinis aus Mantua die Grafschaft Medole (Medularum) in der Brixener Diözese samt dem Hofe Cassano.*

🇬🇧 *confirms that the shire Medole (Medularum) in the diocese of Brixen and the farmyard at Cassano belongs to the brothers Martin, Benvenuto, Franz u. Jacob de Peccorinis of Mantua.*

Figure 6: Regest summarizing a charter issued by Sigmund 1413 in the location of " Udine (Wyden) " and our English translation.

A. Experiment for determining the text place solver

We experiment with two solutions. (i) STEINERAPPROX, a Steiner-tree approximation. It uses the metric closure of the graph induced by the terminal nodes to generate a minimum spanning tree (MST), where the metric closure of G is the complete graph in which each edge is weighted by the shortest path distance between the nodes in G.[8]. (ii) HILLCLIMBING: starting with a random candidate set and one of its points, select as the next point the point from the candidate set which minimizes traveling cost, repeat until all candidate sets have been visited and run a final $resolveItinerary(X, \cdot)$ over the emerged order of place names and their candidates X.

To assess the efficacy of the approaches in order to select among them, we sample a subset of 5,000 regests. The baseline is RANDOM (randomly selecting one point from each candidate set). Table 6 displays the results of this experiment in terms of the avg. cost to travel from each point to each point, per regest instance. It shows that both methods outperform the random baseline, however, they appear to yield similar solutions, reflected in a cost that is almost equal, for all three methods. Therefore, we decide upon the HILLCLIMBER for resolving the text places, since it is more than 8 times faster than STEINERAPPROX.

B. Example studies

B.1. Case study of predictions

Figure 6 displays a randomly selected regest released by Sigmund 1403 in Udine. Here, we go through the exact resolutions, discussing errors and correct predictions.

Charter origin prediction The place name of charter origin, stated as *Udine (Wyden)* is correctly resolved, even though the addition *(Wyden)* could have led to confusion:

```
"1413-05-08_1_0_11_1_0_485_474": {
        "prediction:": {
            "asciiname": "Provincia di Udine",
```

[8] networkx.algorithms.approximation.steinertree.steiner_tree

```
            "geonameid": "3165071",
            "latitude": "46.16408",
            "longitude": "13.17794",
            "name": "Provincia di Udine",
            "population": "535430"
        }}
```

Charter text place predictions The NER and dependency program (almost) correctly identified the entity (two brothers: *"Franz u. Jacob de Peccorinis"*) as the object of the *confirm* action (the error is that this is the dative object and not the accusative object, see *heads*, below). Another error is that it misses that there are actually four brothers and only captures two. But this would have been difficult to detect for the NER system, since *the brothers Martin, Benvenuto, Franz u. Jacob de Peccorinis of Mantua* is a rather complex nested phrase that contains multiple person entities. Our method correctly extracted *Peccorinis* as their location but resolved it, most likely, wrongly:

```
                "heads": [
                    "oa --> best\u00e4tigt & "
                ],
                "label": "PER",
                "prediction": {
                    "asciiname": "Perini",
                    "geonameid": "8976658",
                    "latitude": "45.5643",
                    "longitude": "11.16851",
                    "name": "Perini",
                    "population": "19"
                },
                "text": "Franz u. Jacob de Peccorinis",
                "associated_with_loc": "Peccorinis",
                "used_name": "Peccorinis"
            }
```

It is unclear to us, what the correct location would have been in this case. However, as the following predictions will indicate, this guess may not be too far off from the real place that was meant here (since it predicted coordinates close to the Alps in northern Italy, near to the places of the following correct predictions). Namely, the NER system correctly identified *Mantua* as a location (this time the dependency edge is more correct in predicting the accusative object). This time, the resoultuion worked flawless:

```
"heads": [
             "oa --> best\u00e4tigt & "
                ],
                "label": "LOC",
                "prediction": {
                    "asciiname": "Provincia di Mantova",
                    "geonameid": "3174050",
                    "latitude": "45.16667",
                    "longitude": "10.78333",
                    "name": "Provincia di Mantova",
                    "population": "408336"
                },
                "text": "Mantua",
                "used_name": "Mantua"
            },
```

Similarly, the shire Medole was correctly identified and resolved appropriately:

```
"heads": [
                    "oa --> best\u00e4tigt & "
                ],
                "label": "LOC",
                "prediction": {
                    "asciiname": "Medole",
                    "geonameid": "3173657",
                    "latitude": "45.32588",
                    "longitude": "10.51357",
                    "name": "Medole",
                    "population": "3522"
                }
                "text": "Grafschaft Medole",
                "used_name": "Grafschaft Medole"
```

Likewise, *Brixen* and *Cassano* have been resolved appropriately:

```
                {
                "heads": [
        "oa --> best\u00e4tigt & "
    ],
    "label": "LOC",
    "prediction": {
        "asciiname": "Bressanone",
        "geonameid": "6535887",
        "latitude": "46.70893",
        "longitude": "11.65638",
        "name": "Bressanone",
        "population": "20677"
    },
    "text": "Brixener",
    "used_name": "Brixener"
},
    {
    "heads": [
        "oa --> best\u00e4tigt & "
    ],
    "label": "LOC",
    "prediction": {
        "asciiname": "Cassano Spinola",
        "geonameid": "3179793",
        "latitude": "44.76557",
        "longitude": "8.86228",
        "name": "Cassano Spinola",
        "population": "1648"
    },
    "text": "Cassano",
    "used_name": "Cassano"
    }
```

B.2. Example of a resolution of an entity that is associated with an ancient place name

This prediction is correct, since *Konstantinopel* used to denote the location that is nowadays referred to as *Istanbul*.

```
"associated_with_loc": "Konstantinopel",
"heads": [
    "mnr --> Schicksal & ",
    "nk --> \u00fcber & ",
    "op --> informiert & "
],
"label": "PER",
"prediction": {
    "asciiname": "Istanbul",
    "geonameid": "745044",
    "latitude": "41.01384",
    "longitude": "28.94966",
    "name": "Istanbul",
    "population": "14804116"
},
"text": "Nikolaos von Konstantinopel",
"used_name": "Konstantinopel"
```

B.3. Predictions for the regest in Figure 1

B.4. Fulda

```
"heads": [
                "da --> schenkt & "
         ],
"label": "LOC",
"prediction": {
                 "asciiname": "Landkreis Fulda",
                 "geonameid": "3220993",
                 "latitude": "50.58278",
                 "longitude": "9.76111",
                 "name": "Landkreis Fulda",
                 "population": "221170"
             },
             "text": "Fulda",
             "used_name": "Fulda"
```

B.5. Abbot Huoggi

The NER system correctly detected a *PER*. However, no place name is stated in his name, therefore, we interpolate his location:

```
    "associated_with_loc": "None",
                "heads": [
                    "nk --> unter & ",
                    "mo --> schenkt & "
                ],
                "label": "PER",
                "prediction": {
                    "asciiname": "INTERPOLATION_NONAME",
                    "geonameid": "INTERPOLATION_2850688_2854068_2856971_2873759_2906098_2906726_2934058_3220993_3314094_6550970_6556941",
                    "latitude": 50.696850909090905,
                    "longitude": 9.974371818181819,
                    "name": "INTERPOLATION_NONAME",
                    "population": -1
                },
                "text": "Huoggi",
                "used_name": "None"
```

This is quite accurate, since the interpolated place is not far from the abbot's true stay (Fulda in Hesse) and lies only 30 km to the east.

B.6. Helmershausen

```
"heads": [
                    "nk --> zu & ",
                    "mnr --> k\u00f6nigshufen & ",
                    "cj --> schenkt & "
                ],
                "label": "LOC",
                "prediction": {
                    "asciiname": "Helmershausen",
                    "geonameid": "2906726",
                    "latitude": "50.56325",
                    "longitude": "10.23763",
                    "name": "Helmershausen",
                    "population": "0"
                },
                "text": "Helmershausen",
                "used_name": "Helmershausen"
```

B.7. Hengistdorf

We assume that the following prediction is wrong (since the predicted *Hergisdorf* is known 'only' since 1252, according to the district's webpage[9]). But we do not know if the correct place is known today.

```
            "heads": [
                    "nk --> in & ",
                    "mnr --> Ramuolt & ",
                    "ag --> lehen & ",
                    "cj --> und & ",
                    "cd --> k\u00f6nigshufen & ",
                    "cj --> schenkt & "
                ],
                "label": "LOC",
                "prediction": {
                    "asciiname": "Hergisdorf",
                    "geonameid": "2906098",
                    "latitude": "51.53333",
                    "longitude": "11.48333",
                    "name": "Hergisdorf",
                    "population": "1777"
                },
                "text": "Hengistdorf",
                "used_name": "Hengistdorf"
```

Again though, the prediction may not be too far off from the true place, since it lies more or less in proximity to the other places of this charter, that were correctly predicted.

[9] https://www.verwaltungsamt-helbra.de/gemeinden/hergisdorf-2/