

Induction of a Large-Scale Knowledge Graph from the *Regesta Imperii*

Juri Opitz Leo Born Vivi Nastase

Institute for Computational Linguistics

Heidelberg University

69120 Heidelberg

{opitz,born,nastase}@cl.uni-heidelberg.de

Abstract

We induce and visualize a Knowledge Graph over the Regesta Imperii (RI), an important large-scale resource for medieval history research. The RI comprise more than 150,000 digitized abstracts of medieval charters issued by the Roman-German kings and popes distributed over many European locations and a time span of more than 700 years. Our goal is to provide a resource for historians to visualize and query the RI, possibly aiding medieval history research. The resulting medieval graph and visualization tools are shared publicly.

1 Introduction

We describe here the process of inducing and visualizing a Knowledge Graph (KG) that structures information from the Regesta Imperii (RI), an important large-scale resource for medieval history research. Having important information from the RI in a structured format makes it easier to visualize, and possibly aid medieval history research.

The *Regesta Imperii* (RI) comprises documents, *regests*, that can be seen as abstracts of charters issued by Roman-German emperors and popes, starting from the Carolingian dynasty to Maximilian I. The project was initiated in 1829 by a German librarian, *Johann Friedrich Böhmer*, who started to collect and summarize the charters. Today, more than 175,000 regests have been converted to Unicode and are stored in a publicly available and continuously increasing online database¹ due to the efforts of various research projects.²

We extract relations and entities from the regest texts and meta information, and build a large-scale knowledge graph that covers approximately 83% of the documents (Sections 2-3). Information about the entity types and about the events themselves reveal interesting observations about the behaviour of emperors and popes with respect to their subjects. The graph can be explored through a web-based visualization tool (Section 4).

2 Constructing the RI Knowledge Graph

The RI corpus. With regard to the referenced charters, the RI are unevenly distributed over a large time span (Figure 1). Many regests reference charters from the later medieval times issued by the emperors Karl IV (1316-1378, 15,595 regests), Friedrich III (1415-1493, 21,477 regests) and Maximilian I (1459-1519, 22,153 regests) and very often consist of one, often complicated sentence, describing an action performed by the issuer (usually an emperor or pope) towards one or several of his subjects, as can be seen

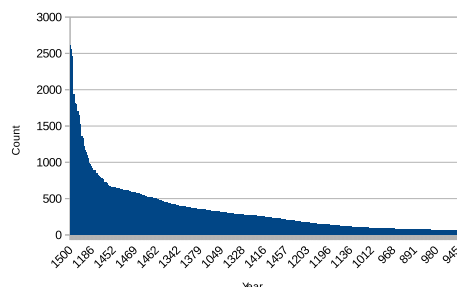



Figure 1: Distribution of regests over time with respect to year of charter creation.

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: <http://creativecommons.org/licenses/by/4.0/>

¹<http://www.regesta-imperii.de/en/home.html>.

²Under the umbrella of the *German commission for the handling of the Regesta Imperii e. V. (Deutsche Kommission für die Bearbeitung der Regesta Imperii e. V.)*.

 [rel]verspricht] dem Kurfürsten [oa_{rel} Ludwig von der Pfalz] für dessen allenfallsige Wahlstimme ihm alle seine [oa_{oc} Privilegien Reichspfandschaften] (Oppenheim, Gauodernheim, Ober- u. Nieder-Ingelheim, Winterheim, Dexheim, Nierstein, Schwabsburg, Kaiserslautern, Barr, Ortenberg, Offenburg, Gengenbach, Zell, Sels) u. s. w. zu [oc]bestätigen]. Mitsiegler Burggraf Friedrich VI v. Nürnberg.



 [rel]promises] the Elector Palatine [oa_{rel} Ludwig of the Palatinate] that if he gets his vote, that he [oc]confirms] all the Palatine Elector’s [oa_{oc} privileges earnests] (Oppenheim, Gauodernheim, Ober- u. Nieder-Ingelheim, Winterheim, Dexheim, Nierstein, Schwabsburg, Kaiserslautern, Barr, Ortenberg, Offenburg, Gengenbach, Zell, Sels) etc. Also sealed by castle-count Friedrich VI of Nuremberg.

Figure 2: Regest summarizing a charter issued by emperor Sigmund in 1410 and (our) translation. *rel* is the main relation, *oa* (accusative object) indicates *whom* the emperor is promising something, *oc* (object clause) indicates *what* the emperor is promising. Green/orange indicate named entity persons/locations found by the automatic tool, which are all accurate in the example.

 Kg. Ruprecht gibt seinem kuchenschriber Hermann von Mülin und dessen erben für seine treuen dienste sein haus und den stall daran zu eigen, “daz gelegen ist in unßer Stad Heidelberg by unsem marstalle off ein syte an der apoteken und off die andern syten an Zengels des smyts (...)”


 King Ruprecht gives to his kitchen-bills-accountant Hermann von Mülin and his heirs for his faithful services his house and barn into his possession “which lies in our city of Heidelberg at the Marstall on one side the pharmacy and at the other side the blacksmith (...)”

Figure 3: German text and (our) English translation of a regest summarizing a charter issued by king *Ruprecht* in 1404. Code is switched to an older German variant in the middle of the text.

in the regest³ in Figure 2. While this regest is syntactically quite complex, it contains only conventional German, where *conventional* means that syntax and spelling conform with contemporary German. Other regests, having been created earlier or by different authors, contain words with non-conventional spelling or nouns in lowercase, which is highly unconventional in contemporary German. An example is displayed in Figure 3.⁴ *erben* (heirs) and *dienste* (services) are nouns currently written with an uppercase starting letter; *kuchenschriber* (kitchen bills accountant) is a (very rare) noun, spelled *Küchenschreiber* in contemporary German. In the middle of the text the code switches and we find a much older variant of German, a quotation from the original charter of 1404, where spelling and syntactic constructions differ significantly, e.g. *Stadt* (city) is written *Stad* and *Seite* (side) is spelled *syte*.

Corpus statistics are summarized in Table 1.⁵ We find more than 16 million tokens and almost 2 million named entity mentions in the RI. Because the RI stem from different authors and times, the spelling of words can be very variable. For example, there are many spelling variants of the Austrian city *Innsbruck* (*Ynnsbrug*, *Ynsbrug*, *Innsbruck*, *Insprugk*, etc.). Code-switching is ubiquitous not only from author to author but

| element | sum | types | μ | $\tilde{\mu}$ | σ |
|-----------|------------|---------|-------|---------------|----------|
| regests | 179,320 | - | - | - | - |
| issuers | 179,320 | 419 | - | - | - |
| locations | 179,320 | 13,656 | - | - | - |
| tokens | 16,525,042 | 488,619 | 92.2 | 56 | 150.7 |
| NPs | 4,097,632 | 903,383 | 22.9 | 14 | 34.7 |
| NEs | 1,977,866 | 363,162 | 11.0 | 7 | 15.7 |

Table 1: Corpus statistics.

also inside a regest, where the text often contains quotes in Latin and medieval German, taken from the original charter. Named entities in the RI can be rather complex, for example *Adelheid, tochter weiland Ulrichs von Minzenberg* (English: “Adelheid, daughter of former Ulrich of Minzenberg”) includes a sub-named entity (*Ulrich von Minzenberg*). *tochter*, as a noun, should start with an uppercase letter and *weiland* is an outdated and uncommon German adverb, used in the sense of “formerly”.

³URI: http://www.regesta-imperii.de/id/1410-08-05_4_0_11_1_0_4_4.

⁴URI: http://www.regesta-imperii.de/id/1404-06-26_1_0_10_0_0_3588_3584.

⁵Linguistic annotation was performed with spacy’s German model v2.0.0: <https://spacy.io/models/de>.

Preprocessing. We use a state-of-the-art German preprocessing pipeline. The preprocessing model was trained on contemporary German texts, so it naturally makes more errors on the RI. We manually examine the outputs for 100 randomly chosen regests, and find that, particularly on the first sentence, all steps yield reasonably good results on named entity recognition, and the syntactic dependency parse is often correct in detecting the finite main verb and its object, or parts of it.⁶

Some errors are caused by non-conventional spelling (e.g. lowercased nouns tagged as verbs). We will address this type of errors in future work, mainly because countless rare nouns and a high rate of spelling variations make classical normalization techniques very challenging. Applying such techniques, e.g. techniques based on lexical lookups and minimum edit distance computations (Ristad and Yianilos, 1998; Yujian and Bo, 2007), may even introduce many new errors and manipulate the historic data in unwanted and difficult-to-detect ways.

We find, however, a specific type of error which is due to an eager sentence splitting model: e.g. a sentence split often is performed on title prefixes. In the RI a title – abbreviated with a period – often prefixes a named entity: e.g. *Gf.* – *Graf* (count), *Bf.* – *Bischof* (bishop), *Eb.* – *Erzbischof* (archbishop).

We introduce three processing steps to minimize this type of error: (1) iterate through 15,000 randomly sampled regests and aggregate the tokens occurring directly in front of named entities; (2) compute their frequencies; (3) filter instances starting with an uppercase letter, ending with a period, having a minimum frequency of 5 and a maximum length of 5 characters. Taking into account only the prefixes starting with an uppercase letter has the advantage of increased precision (verbs are not counted). Recall can be increased afterwards by taking into account the prefixes’ lowercase variants as well (sometimes bishop was abbreviated *bf.* and at other times *Bf.*). After obtaining the possible abbreviations a historian helped us in manually filtering out 250 common abbreviations in the RI (false positives such as Roman numerals – e.g. *XI.* – were discarded in this step). After applying these steps, we obtain an interesting vocabulary of historic German abbreviations (Table 2). The amount of erroneous sentence splits was reduced by 10%.

Edges and vertices. Based on the assumption that the first (and often only) sentence contains the relevant relation information, we process only this sentence. We rely on the fact that the main relation (edge) between a ruler and its subject entities (the vertices) very frequently occur in a subject-verb-object format with respect to the first finite verb. As arguments of the relation we consider the first occurring named entity and the named entity which acts as the accusative or dative object of the first finite verb. Named entities in a comma-separated list or appearing in *and*-conjunctions of which at least one has an object dependency relation to the main verb, are all added to the graph separately. The source is the issuer or emperor, which we assume to be the subject of the first finite verb and which we extract from the metadata as it is often omitted in the regest text (cf. Figure 2). The KG will thus contain two types of vertices: (i) the issuers (emperor or pope) and (ii) named entities found by the NLP tool.

Edges are weighted: if a named entity is mentioned first after the finite verb *or* has an object dependency to it, the weight of the edge is increased by 0.5; if a named entity is mentioned first after the main verb *and* lies in

| abbreviation | title/function | freq. |
|--|----------------|-------|
| <i>Kgin.</i> | queen | 7 |
| <i>Kg., Ks.</i> | king, emperor | 1,125 |
| <i>Hz., Hz., Hzg.</i> | duke | 369 |
| <i>Hzn., Hzin.</i> | duchess | 8 |
| <i>Mgf., Pfgf., Ldgf., Pfgf., Mggf., Bggf.</i> | various counts | 252 |
| <i>Eb., EB., Bf., BF., Erzb.</i> | (arch-)bishop | 121 |

Table 2: Some of the automatically collected medieval title abbreviations and their frequency in a random sample of 15,000 regests.

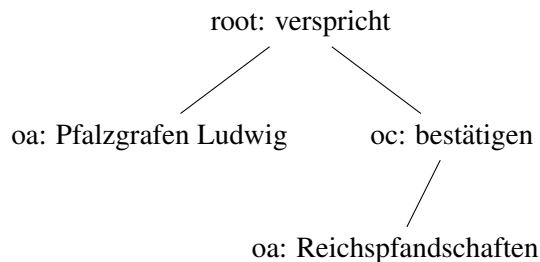


Figure 4: Dependency parse for Figure 2.

⁶The first two authors are German native speakers.

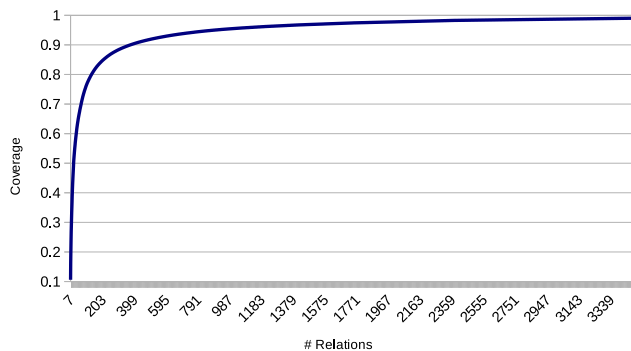


Figure 5: 400 out of 4,948 relations cover over 90% of the relations induced into our KG.

| relation | translation | count | coverage |
|------------------|-------------|--------|----------|
| <i>bestätigt</i> | confirms | 15,811 | 0.11 |
| <i>befiehlt</i> | commands | 8,408 | 0.06 |
| <i>verleiht</i> | bestows | 5,796 | 0.04 |
| <i>schreibt</i> | writes | 5,427 | 0.04 |
| <i>schenkt</i> | donates | 2,652 | 0.02 |
| <i>fordert</i> | demands | 1,619 | 0.01 |
| <i>gewährt</i> | permits | 1,425 | 0.01 |
| <i>bittet</i> | asks | 1,004 | 0.01 |

Table 3: The most frequent relations in our KG, with (our) English translations.

the object dependency of it, the weight is incremented by 1.0. For example, from Figure 2 (dependency tree: Figure 4) we retrieve the tuple (*Sigmund*, *promises*, *Ludwig of the Palatinate*), and insert the subject and object into the graph (if not already contained). If the relation *promises* already exists between these entities in the KG, we increment the edge weight by 1.0, since, as can be seen in the example, *Ludwig of the Palatinate* was detected both as object of *promises* (+0.5) and as the first named entity (+0.5).

The more *promise*-events are instantiated between these two entities, the higher the weight of the edge. This implies that an edge or relation r with weight w between an emperor and one of its subjects means that in the data there were $1 \leq n \leq w$ instantiations of r detected between these entities and thus the edge weight can be interpreted both as a reliability measure of the relation and an indicator for how many times the relation was instantiated.

While we induced 4,948 distinct relations in the KG, the 400 most common relations cover 90% of the relations induced in the KG (Table 3 and Figure 5). Edge quality will be analyzed in the next section. Regests that do not contain verbs in the first sentence are discarded, e.g. *Geburt Karls, des jüngsten Sohnes Kaiser Ludwigs (d. Fr.) und seiner zweiten Gemahlin Judith* (English: “Birth of Karl, the youngest son of emperor Ludwig (the Pious) and his second wife Judith”).⁷ Applying these steps, more than 149,203 regests yield a relation, thus 83% of the RI have found their way into our KG.

Edge attributes. Each edge (relation) in the KG can be seen as representing one or more event(s) between an emperor and one of his subjects. We can associate an attribute list to each edge: date, time and location of creation of the event are taken from respective fields in the XML of the regests. We are also interested in key phrases associated with each event. For example, at one time king Sigmund may have promised *bestowal of land* to duke Ludwig and at another time he might have promised him *privileges* or *financial help*. To find associated key phrases, we formulate a text classification task, where we predict the main relation of a regest (i.e. the finite verb of the first sentence) using a bag of all contained phrases represented as a binary vector. We consider all phrases tagged as noun chunks or verbs (except the finite verb) as features. We only consider relations that occur more than 25 times, leaving us with 397 classes plus one catch-all class for the less frequent relations. Having fitted a logistic regression model to the task in a one-vs.-rest setting, we can rank the phrases describing an instantiation of a relation according to the learned weights for this relation. For example, given that the relation extracted from the example regest *Sigmund*→*promises*→*Ludwig of the Palatinate* was instantiated only one time, a list associated with this edge contains the description of one specific event. The event took place in 1410 at the location *Ofen* and, after having queried the model’s coefficients for the relation *promise*, we see that highly associated key phrases of the event are the noun chunks *Privilegien* (privileges) and *Reichspfandschaften* (earnests). If there was another regest with a *promise*-relation between those same entities, we add another event description to the edge attribute list and increment the weight of the edge accordingly. The key phrase ranking relation-prediction approach is evaluated and examined more closely in the next section.

⁷URI: http://www.regesta-imperii.de/id/0823-06-13_1_0_1_2_1_1_1.

Kg. Ruprecht_{NOUN→PROPN} gibt seinem kuchenschreiber_{ADJ→NOUN} Hermann_{NOUN→PROPN} von Mülin und dessen erben_{VERB→NOUN} für seine treuen dienste_{ADJ→NOUN} sein haus und den stall_{X→NOUN} daran zu eigen.

[Kg.]_{PERSON} [Ruprecht]_{ORG} gibt seinem kuchenschreiber [Hermann von Mülin]_{PERSON} und dessen erben für seine treuen dienste sein haus und den stall daran zu eigen.

Figure 6: Part-of-speech (top) and NER annotations (bottom). Severe errors are marked in red. An arrow indicates the correct tag.

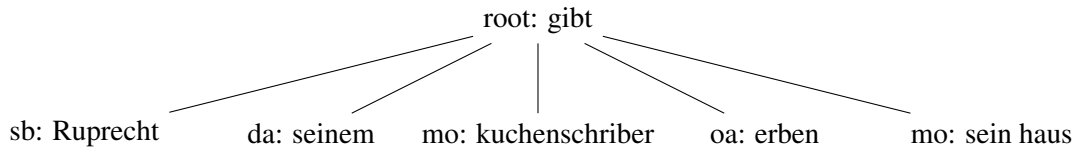


Figure 7: The erroneous dependency parse for Figure 3. *kuchenschreiber* (kitchen bills accountant) is mislabeled as the relation modifier (mo) and *erben* (heirs) is mislabeled as accusative object (oa) – *sein haus* (his house) would have been the correct accusative object.

3 Problem Analysis

Despite syntactic parse errors due to the text complexity, we find that many named entities are being captured and the syntactic parse is often partially correct with respect to detecting the direct object of the finite verb, as can be seen in Figure 4. In some cases, however, we obtain very erroneous parse trees.

The regest in Figure 3 gets an erroneous dependency tree (Figure 7). The main reasons for this seem to be errors from preceding pipeline steps (Figure 6): even when disregarding the quote in the older variant of German, the text is annotated with false part-of-speech tags, propagating errors through the rest of the pipeline and into the dependency parse. Nevertheless, the relation is correctly induced with a weight of 0.5 because *Hermann von Mülin* is correctly detected as the first named entity after the verb. On the other hand, the regest in Figure 2, mostly written in today’s conventional German, is correctly parsed with respect to the main relation, its accusative object and even the distant object clause.

For future work we want to experiment with fine-tuning the preprocessing models for the RI data. This is a non-trivial task because manual annotation would require domain experts. Other means such as bootstrapping the NER-pipeline by filtering and crawling part-of-speech patterns may introduce new errors, possibly even worsening the overall performance of the system, which is difficult to assess without manually labeled data. For further graph refinement one could consider disambiguating the named entities and merging their nodes. The KG could also be improved by identifying the (minority of) cases where the regest is not an event description of the emperor unidirectionally interacting with other entities but vice-versa (e.g. an entity gives a present to the emperor).

Inspection and Evaluation of the Relation Prediction Model. To investigate the performance of the model on unseen data, we divide the RI into a training set of 100,000 instances and 79,320 test instances, and use key phrases, issuers, locations and dates as binary features to predict one of the 398 relations (397 most common with KG-induction frequency of > 25 and one rest-class). This classification task is difficult because the many classes are also unevenly distributed, varying in frequency from 44,303 to 26, with an average of 450.5 and a median of 76 instances per class. The results are shown in Table 4. The relation classifier achieves an accuracy of 0.487 and macro f1 of 0.141, significantly higher than the baselines (majority baseline: 0.003 f1). Both baselines, majority voter and random baseline (drawing classes according to estimated class probabilities in the training data), lag behind not only the full-feature model but also a logistic regression model which uses only location or issuer features. We conclude that certain issuers and locations are more associated with specific relations than others. Our model’s learnt coefficients can also be used to investigate these associations. For example, we find that the coefficients for emperor Friedrich III. have the highest value for the relations *quittirt* (signs), *legitimirt* (legitimizes),

| feature type | #features | acc. | macro f1 |
|---------------|-----------|--------------|--------------|
| all | 447,028 | 0.487 | 0.141 |
| key phrase | 436,508 | 0.461 | 0.113 |
| issuer | 401 | 0.267 | 0.007 |
| year | 821 | 0.258 | 0.005 |
| location | 9,298 | 0.267 | 0.007 |
| maj. baseline | - | 0.247 | 0.001 |
| ran. baseline | - | 0.008 | 0.003 |

Table 4: Evaluation of the 398-class relation prediction task.

| scenario | micro error | macro error |
|------------------|-------------|-------------|
| all | 0.083 | 0.934 |
| confident (@397) | 0.052 | 0.174 |
| @1 | 0.0 | 0.0 |
| @5 | 0.0 | 0.0 |
| @25 | 0.0 | 0.0 |
| @125 | 0.03 | 0.104 |

Table 5: (Pessimistic) estimates of weighted (micro) and unweighted error (macro) of induced relations due to false part-of-speech tags. @n: considering the n relations in the KG which were induced most often.

gibt (gives), *gebietet* (dictates), *befiehlt* (commands), *erlaubt* (permits), while for emperor Maximilian I., there is a different picture and we find many financial relations such as *schuldet* (owes), *verbucht* (books) and *bezahlt* (pays). Maximilian I. was well known for his debts he accumulated due to many wars and a rather extravagant lifestyle and we can see that this is also statistically reflected in the RI.

Inspecting the fitted model’s coefficients for the relation *verhängt* (imposes), we see that among highly associated phrases are *die Reichsacht*, *die Acht*, *die Aberacht*, *Klage*, *den Bann*, *die Reichsaberacht*, *das Anathem*, which are mostly outdated German words for banning (excommunicating) somebody (*Acht*, *Aberacht*, *Bann*) and also anathema (*Anathem*). Highly weighted terms for the relation describing *verleiht* (bestow) are *das Pallium* (a religious clothing), *Regalien* (regalian rights), *ein Wapen*⁸ (a crest), *Lehens- und Landesrecht* (land rights), *Gerichtsstandsprivileg* (court-right), *Doktorwürde* (doctorate) and *Halsgericht* or *Blutbann*, both old German terms in the context of the relation *bestows* referring to the right to speak the death sentence (a right which was granted only to selected cities). We conclude that our binary, bag-of-phrases representation for relation classification is a robust and straightforward way not only to rank key phrases describing medieval events but also to gain deeper insights into the RI.

Quality of Graph Edges We evaluate the quality of graph edges (first detected finite verb in a regest): (i) to assess the overall quality of edges and (ii) from the point of view of a user or historian, who might be interested in the quality of high-confidence edges, e.g. he choses to consider only relations with weight $> w$. The relation extraction experiments indicate that a falsely induced relation is due to false positive verb errors from POS tagging. We manually checked the 397 most commonly induced relations and found 69 to be incorrectly POS tagged. Examples of errors include *wittwe* (widow) and *archidiacon* (archdeacon). Everything that was erroneously tagged as a verb was labeled an error – an exception were deverbal nouns.⁹ We assume a pessimistic scenario of all relations with count equal to or less than 25 having been misclassified with false POS tags. This scenario is very pessimistic because we not only find many rare and generally uncommon German verbs with varied spellings in this subset (e.g. *vidimiert*, *vidimirt* – a very rare word meaning “he witnesses”) but also deverbal nouns which can be useful. Thus, given n distinct relations $i = 1, \dots, n$ and f_i returns the total number of times this relation was induced in the KG, and g_i returns 1 if the relation was likely falsely inserted and 0 if it likely was correctly inserted (regarding POS annotation), we estimate the weighted micro error:

$$\frac{\sum_{i=1}^n f_i \cdot g_i}{\sum_{i=1}^n f_i}. \quad (1)$$

For the unweighted macro error, $f_i = 1 \forall i$. The results are shown in Table 5. The macro error, i.e. when all relations have the same significance, is estimated at 0.17 for scenario (ii), when the manually annotated relation subset of the most frequent 397 relations was used. When taking into account as errors

⁸*Wapen* is not a mistake – the word is spelled *Wappen* in contemporary German but we find both forms in the RI.

⁹Specifically, those nouns were *belagerung* (siege), *Eintreffen* (arrival) and *belehnung* (mortgage).

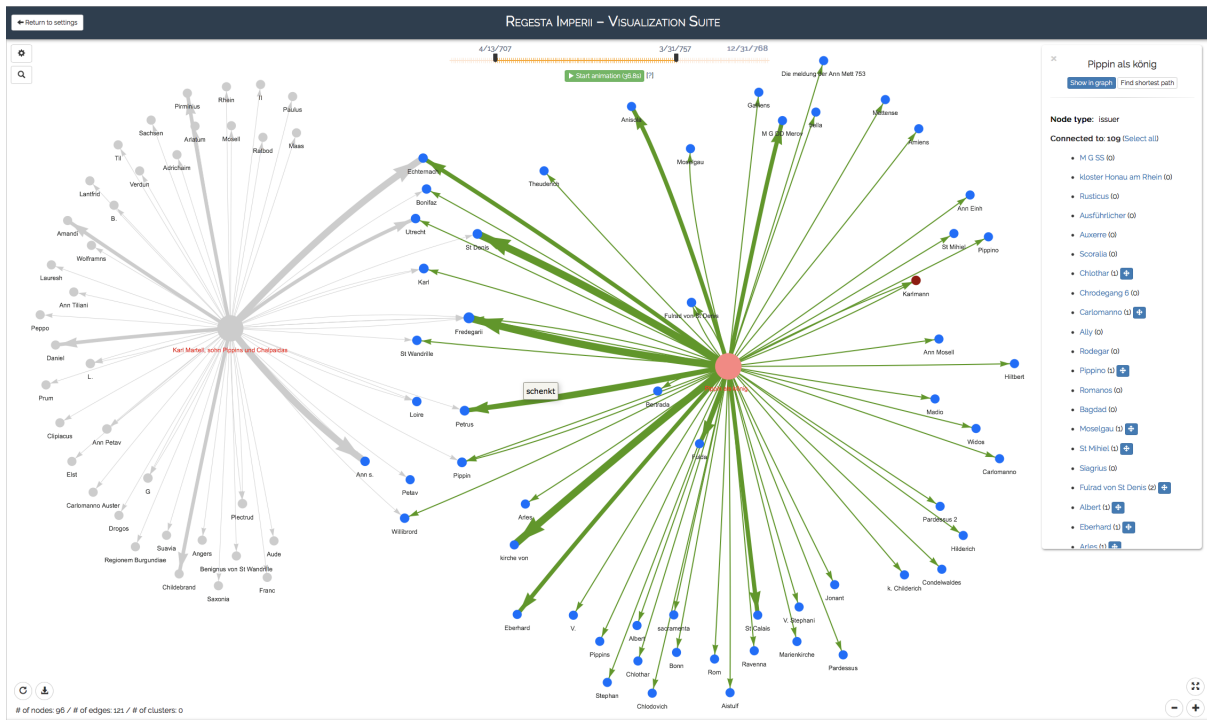


Figure 8: Example graph of the two issuers king *Pippin* (selected) and *Karl Martell* (father of Pippin) between 4/13/707 and 3/31/757 (96/148 nodes, 121/196 edges). Highlighted is an edge for the relation *schenkt* (endow). Edge thickness indicates edge weight and nodes are scaled by degree centrality. The figure is best seen in color.

all the other, less frequent relations, the macro error is < 0.93 , which of course is a very pessimistic estimate. However, in most cases, assessing the weighted micro-error would be of much more interest – some relations cover a much larger proportion of induced relations. For example, a historian who is interested in the most frequent 125 relations (@125 in Table 5) would expect a weighted error of 0.03 and pessimistically an unweighted error of at most 0.104. When choosing the most common 25 relations (e.g. @25 in Table 5), the error is zero for both micro and macro calculations with regard to falsely induced relations due to POS tag errors.

4 Visualization: Diving into European Medieval Times

Visualization of textual (meta)-data is crucial both to make sense of the data itself and also to distribute the information in a more accessible way. Figure 9 displays our induced graph, where modularity clusters are colored differently. Inspecting the main clusters shows that some clusters roughly represent the universes of single emperors (violet: Friedrich III. (14.3% of all nodes), blue: Sigmund (9.54%), black: Karl IV. (6.14%)), while other clusters encapsulate multiple emperors (pink: the Palatine emperors Ruprecht I., Ruprecht II., and Ruprecht III. (7.45%)) and additionally there appear to be clusters which subsume the universes of less prevalent and especially earlier emperors (orange: Otto III., Otto I., Karl the Great, etc. (7.14%)).

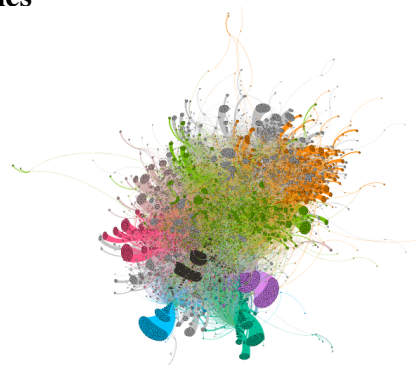


Figure 9: RI-graph with colored modularity clusters (Brandes et al., 2008) after application of ForceAtlas2 (Jacomy et al., 2014) in Gephi (Bastian et al., 2009).

Interactive visualization is a methodological tool to assist researchers in exploring the data, allowing for non-destructive data manipulation and filtering. For the Regesta Imperii we aim to provide an easy-

to-use exploration tool of the extracted knowledge graph for historians. We built a web application to visualize the data. The application is JavaScript-based and uses AngularJS 1 as the frontend framework. Visualization is done using *vis.js*,¹⁰ an open-source tool aimed at network and timeline visualizations.

Our KG contains 68,726 nodes and 154,797 edges. The average outdegree of nodes is 380 and average indegree is 2.65, showing that the KG is highly uni-directional with few central nodes (i.e. issuers) and most peripheral nodes having few incoming and almost no outgoing edges. Structurally speaking, the knowledge graph is weakly connected, consisting of 14 components with one large component containing 68,692 and the remaining ones containing < 5 nodes. In the largest component, of the 25 nodes with highest betweenness centrality, 7 are issuers and 18 are persons. This indicates that in terms of shortest paths between all node pairs, persons are especially important in establishing the connectivity of the graph. However, when looking at the 70 cut vertices of the component that are named entities,¹¹ we see that 55 are persons, but 11 denote place or region names (e.g. *Frankreich* or *Reutlingen*), indicating that certain regions also play a critical role in connecting multiple sub-networks.¹²

An example network is shown in Figure 8. Although the sheer size of the KG prohibits it from being visualized in its entirety, we use optimizations and ad-hoc querying to visualize sub-networks of the KG. With these, we were able to create interactive visualizations for networks of up to 12,000 nodes in practice. On the settings page, a user can choose one or more issuers. Based on the selected issuers, a network with all other nodes with which they interacted gets created. Furthermore, there are a number of additional settings, which can be divided into **limiting constraints** and **network enhancements**.

Limiting constraints. These limit the resulting network on an ad-hoc basis. Specifically, we allow to set a threshold X ($\leq X$; $= X$; $\geq X$) for the number of connections to display for a named entity and the number of relation instances. With these settings, specific requirements can be made, e.g. “show the network of Friedrich III. with relations that appear at least 40 times”. A combination of constraints is also possible but per default, we set all constraints ≥ 1 so that the full graph will be rendered.

Network enhancements. These apply to the way the network gets rendered and can be interacted with. The first option visualizes the intensity of edges by their associated edge weights. The second option visualizes the importance of persons by means of a selected node centrality measure. Currently, we support degree, eigenvector, and betweenness centrality. The last option enables the *time slider* that allows for filtering the network by event dates. Per default, all three options are disabled.

To further reduce rendering speed, we implemented a decay factor to the number of iterations of the underlying physics solver used to calculate the graph layout. Starting with a network containing more than 150 nodes or 250 edges, the default number of iterations (1,000) is reduced by a factor of 0.1 for every additional 150 nodes or 250 edges.¹³ We set the lower-bound to 100 iterations to give the solver some time to calculate the layout. Using this decay factor and the above-mentioned constraints drastically reduces the load times for very large networks. For example, the network of Karl IV. containing 7,359 nodes and 13,760 edges takes a couple of minutes to render. With the decay factor and the constraint that named entities have exactly 2 edges, the graph is reduced to 785 nodes and 1,568 edges, and is rendered in 80 seconds with 576 iterations.¹⁴ Furthermore, we allow to save graphs as JSON files that can later be re-uploaded, further reducing load times. These are promising results as in a later production stage, the application can be supplemented by a database containing pre-calculated network layouts.

5 Related Work

Historical Text Processing. Piotrowski (2012) gives an overview of the many challenges arising when applying NLP to historical documents. The idea of normalization is often explored, yet we encountered the same problem as the author who reports that the effectiveness of normalization to a large degree depends on text type and language – most satisfying results are achieved only on more recent texts. For

¹⁰visjs.org.

¹¹We exclude issuers from this analysis as cutting them almost always increases the number of components because of their central role.

¹²The remaining four are generic entities such as Roman numerals.

¹³Furthermore, if the node-edge difference is > 900 , we consider the edge number to be determining the decay factor.

¹⁴40 seconds of that are needed to process the data from our KG file, the remaining time is the rendering time.

this reason, we applied little and careful normalization to the regests. We concur that “the highly variable spelling found in many historical texts has remained one of the most troublesome issues for NLP” (p. 83), a fact that may be especially true with regard to the RI. Due to these issues, the research conducted in processing of (German) historical texts has been diverse and very task-specific, see i.a. (Massad et al., 2013; Meroño-Peñuela et al., 2015; Seemann et al., 2017; Hench, 2017; Schulz and Keller, 2016).

Regesta Imperii. Not much NLP research has been conducted on the RI. Kuczera (2015), in an example experiment, projects attributes and relations between entities from the times of Friedrich III. (i.e. a subset of the RI) into a graph database. He applies no NLP in these steps, but relies on the manually created person registers for the universe of Friedrich III. While currently only the registers for Friedrich III. are available, we think that they may be used in future work to fine tune the NER system to achieve better performance not only on the sub-corpus of Friedrich III. but also on the whole RI. A caveat is that the regests of the other emperors differ significantly not only in named entities but also in linguistic variety and thus such a system may fail to generalize. Opitz and Frank (2016) manually labeled 500 randomly sampled regests with 12 medieval themes and players of interest (e.g. *nobles*, *spiritual institutions*, *war and peace* or *justice*) and trained binary classifiers to in the end label all regests and compute statistics about the importance of the medieval themes and players with regard to time.

Relation Extraction For relation extraction from the Regesta Imperii we work in an “extreme” environment – no annotated data; few language resources (particularly because of the German language variations used in the corpus); no tools like named entity taggers or chunkers that would help in identifying relations and relation arguments; and low or no redundancy in relation instances (for computing extraction reliability scores). Because of this, the inspiration for the applied methods comes from open information extraction (Open IE), particularly work where relations are described through POS and grammatical patterns (Banko et al., 2007), and using the assumption that binary relations often appear in a subject-verb-object format. Wu and Weld (2010) extend a previous distantly supervised system that learns using Wikipedia infobox relations, and describe relations (through POS and dependency patterns) as phrases consisting of at least one verb and/or a preposition. To counter noisy extractions (often phrases that are too long and too specific to constitute a relation), Fader et al. (2011) introduce ReVerb, where lexical and syntactic constraints on relation expressions serve to produce cleaner extractions, with less uninformative or incoherent expressions. To increase recall, Mausam et al. (2012) expand the patterns by allowing relational nouns (e.g. Bill Gates, *co-founder* of Microsoft ...). Patterns are gathered and generalized (e.g., from words to parts-of-speech), which boosts recall. All these relation extraction methods rely on redundancy in the data to verify the relation patterns and the extracted candidates through various reliability scores. This cannot be applied to the RI corpus. We rely instead on the relatively simple structure of the regest texts and the grammatical information obtained from a parser.

6 Conclusions

We induced a Knowledge Graph from the medieval Regesta Imperii corpus. Nodes represent medieval named entities, the (weighted) edges indicate relations between those entities, which may have occurred at multiple times in history. High linguistic variation and code switching within single regests pose challenges for modern-day NLP systems, despite their seemingly simple structure. We added preprocessing heuristics to prevent erroneous sentence splitting, and in the process acquired a list of nobility titles. We explored relation classification as multi-class classification based on a binary bag-of-phrases representation of the regests. This not only gives us an option to inject further useful information into the KG describing the encapsulated events more closely, but also allows us to explore phrases highly associated with specific relations. The resulting Knowledge Graph was embedded in an isolated web application, enabling the visualization and querying of the data. The data and application are shared publicly.¹⁵

For future work we are planning to further refine the Knowledge Graph, introduce more structure using nobility titles and entity types and develop and evaluate it together with historians to adjust visualization and querying methods to their needs.

¹⁵<https://gitlab.cl.uni-heidelberg.de/born/ri-visualization>.

References

- Michele Banko, Michael Cafarella, Stephen Sonderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the Web. In *Proceedings of the 22nd Conference on the Advancement of Artificial Intelligence*, Vancouver, B.C., Canada, 22-26 July 2007, pages 2670–2676.
- Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: An Open Source Software for Exploring and Manipulating Networks.
- Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hofer, Zoran Nikoloski, and Dorothea Wagner. 2008. On Modularity Clustering. *IEEE Trans. on Knowl. and Data Eng.*, 20(2):172–188, February.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, UK, 26-29 July 2011, pages 1535–1545.
- Christopher Hench. 2017. Phonological Soundscapes in Medieval Poetry. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 46–56, Vancouver, Canada, August. Association for Computational Linguistics.
- Mathieu Jacomy, Tommaso Venturini, Sebastien Heymann, and Mathieu Bastian. 2014. ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PloS one*, 9(6):e98679.
- Andreas Kuczera. 2015. Graphdatenbanken für Historiker. Netzwerke in den Registern der Regesten Kaiser Friedrichs III. mit neo4j und Gephi. *Mittelalter. Interdisziplinäre Forschung und Rezeptionsgeschichte*.
- D Massad, E Omodei, C Strohecker, Y Xu, J Garland, M Zhang, and LF Seoane. 2013. Unfolding History: Classification and analysis of written history as a complex system.
- Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing*, Jeju Island, Korea, 12-14 July 2012, pages 523–534.
- Albert Meroño-Peñuela, Ashkan Ashkpour, Marieke van Erp, Kees Mandemakers, Leen Breure, Andrea Scharnhorst, Stefan Schlobach, and Frank van Harmelen. 2015. Semantic technologies for historical research: A survey. *Semantic Web*, 6(6):539–564.
- Juri Opitz and Anette Frank. 2016. Deriving Players & Themes in the Regesta Imperii using SVMs and Neural Networks. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 74–83. Association for Computational Linguistics.
- Michael Piotrowski. 2012. *Natural Language Processing for Historical Texts*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.
- Eric Sven Ristad and Peter N. Yianilos. 1998. Learning String-Edit Distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(5):522–532, May.
- Sarah Schulz and Mareike Keller. 2016. Code-Switching Ubiquitous Est - Language Identification and Part-of-Speech Tagging for Historical Mixed Text. In *Proceedings of the 10th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 43–51, Berlin, Germany, August. Association for Computational Linguistics.
- Nina Seemann, Marie-Luis Merten, Michaela Geierhos, Doris Tophinke, and Eyke Hüllermeier. 2017. Annotation Challenges for Reconstructing the Structural Elaboration of Middle Low German. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 40–45, Vancouver, Canada, August. Association for Computational Linguistics.
- Fei Wu and Daniel S. Weld. 2010. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, 11-16 July 2010, pages 118–127.
- Li Yujian and Liu Bo. 2007. A Normalized Levenshtein Distance Metric. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):1091–1095, June.