SEMANTIC SEARCH BASED ON NATURAL LANGUAGE PROCESSING – A NUMISMATIC EXAMPLE

Abstract: Iconographic representations on ancient artifacts are described in many existing databases and literature as human readable text. We applied Natural Language Processing (NLP) approaches in order to extract the semantics out of these textual descriptions and in this way enable semantic searches over them. This allows more sophisticated requests compared to the common existing keyword searches. As we show in our experiments based on numismatic datasets, the approach is generic in the sense that once the system is trained on one dataset, it can be applied without any further manual work also to datasets that have similar content. Of course, additional adaptions would further improve the results. Since the approach requires manual work only during the training phase, it can easily be applied to huge datasets without manual work and therefore without major extra costs. In fact, in our experience bigger datasets generate even better results because there is more data for training. Since our approach is not bound to a certain domain and the numismatic datasets are just an example, it could serve as a blueprint for many other areas. It could also help to build bridges between disciplines since textual iconographic descriptions are to be found also for pottery, sculpture and elsewhere.

Keywords: *Natural Language Processing, Ontology, Numismatics, Iconography*

Patricia Klinger
Sebastian Gampe
Karsten Tolle
Ulrike Peter

Johann Wolfgang Goethe-University of Frankfurt
Berlin-Brandenburgische Akademie der Wissenschaften
klinger, gampe, tolle@dbis.cs.uni-frankfurt.de
peter@bbaw.de

DOI: 10.14795/j.v5i3.334
ISSN 2360 – 266X
ISSN–L 2360 – 266X

1. INTRODUCTION

Iconographic representations on ancient artifacts played an important role within their time. They also served as an information medium due to the lack of alternatives like daily newspapers. This is especially true for groups of artifacts that were produced in bigger quantities and were standardized, like pottery or coins. Therefore, finding and analyzing persons and objects based on their iconographic representation is highly important for archaeologists. Images are used to preserve and digitalize iconographic representations; however, it is still difficult to search for specific items or persons on the images themselves. We are also working in this direction, but it is difficult since the images are generally not labeled, and such labels are needed for the current supervised learning approaches. In this respect, our work can also support this to some extent.

One very common approach is to describe the iconographic representations in natural language that is stored as text. Queries to find iconographic representations could then be based on these text entries. One of the easiest approaches for this would be to search for a specific set of characters, like a name of a person, across all image descriptions. We would call this a *keyword search*. With the growing amount of data that is linked together based on Linked Open Data (LOD) approaches, the limits of such keyword searches gets obvious. On the one hand one gets too many results and on the other hand, one cannot be sure if all relevant data was retrieved.

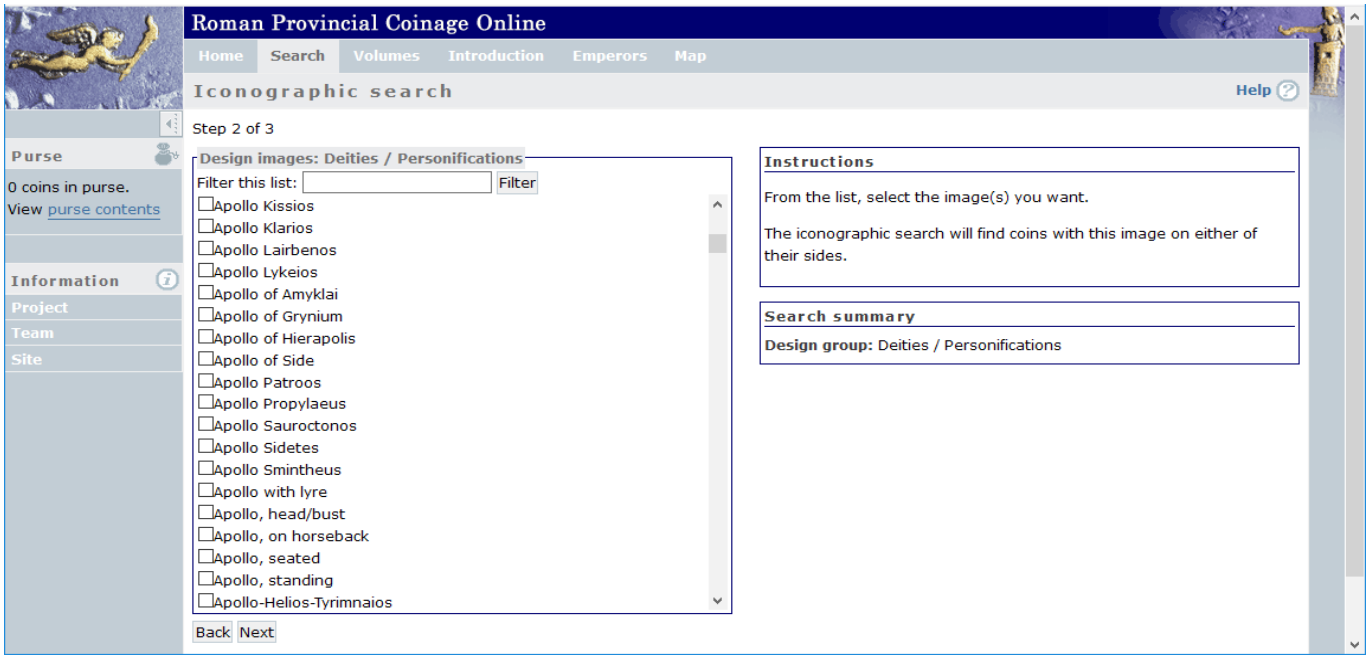

Fig. 1. Roman Provincial Coinage Iconographic search, step 2, list of deities.

The goal of our work was to come up with a solution that is generic enough to work with different textual descriptions that are not always standardized, in order to extract their semantics. This means the solution should work on already existing data without the need to rework it or recreate everything from scratch. The extracted semantics can then be used to run more sophisticated queries with a better fitting result set.

Within this paper, we concentrate on use-cases in the field of numismatic textual descriptions of iconographic representations on coins or types of coins. However, our approach is not bound to this field and the principle idea and steps should be applicable to many other cases where we want to search within many short textual descriptions.

We based our work on the coin data that had been collected within the *Corpus Nummorum Thracorum (CNT)* project[1]. The CNT database contains a virtual meta-collection of ancient coins of Thrace, a region which covers modern Bulgaria, Northern Greece and European Turkey, and consists of data about Thracian coins located in museums and private collections from all over the world. By analyzing these coins, individual coin dies have been reconstructed where possible.

The goal of CNT is to generate a typology of Thracian coins. The iconography of these coins is described in human language. A description of the obverse and reverse of each coin is given both in English and German. Based on the CNT database dump from January 9, 2018, there are 3.628 iconographic descriptions, in the following called *designs*. One example of such a design is:

"Apollo seated left on omphalos, holding bow in right hand."

In this sentence, *Apollo* has a semantic relation (*seated*) to *omphalos* and another semantic relation (*holding*) with *bow*. In a keyword search one could ask for "*Apollo omphalos*", however, it might also be that both words are inside a description but do not have any semantic relation. With our approach, we are able not only to enforce a relation; we can even distinguish between a set of important ones like "holding" or "seated". In addition, we included thesauri (for semantic clustering) and hierarchies for entity classes like persons and objects and relations. This way one could query: *Retrieve all iconographic descriptions where deities are holding arms*. This query would also return the description above.

This paper is structured as follows: In section 2 existing and related work is analyzed. In section 3 the details of the implemented Natural Language Processing (NLP) pipeline and how we worked and trained it with CNT-data are shown. The hierarchies for the persons, objects and other entity classes are further explained in section 4, before section 5 describes how the output of the pipeline and the hierarchies are translated into Resource Description Framework (RDF). We evaluated our trained model with a set of SPARQL queries executed on the CNT data. We also run our system on a different data set from Online Coins of the Roman Empire (OCRE). The results are evaluated in section 6 before we close the paper with a brief conclusion.

2. RELATED WORK

Many numismatic databases and services provide their users with an advanced search to access their data. For example *Mantis: A Numismatic Technologies Integration Service (MANTIS)*[2] and *Online Coins of the Roman Empire (OCRE)*[3] can be queried by specific key words. The *Roman Provincial Coinage (RPC)*[4] and the *Digital Iconographic Atlas of Numismatics in Antiquity (DIANA)*[5] furthermore offer

1 Corpus Nummorum Thracorum, https://www.corpus-nummorum.eu. Accessed 02 August 2018.

2 MANTIS: A Numismatic Technologies Integration Service, http://numismatics.org/search/search. Accessed 02 August 2018.

3 Online Coins of the Roman Empire, http://numismatics.org/ocre/search. Accessed 02 August 2018.

4 Roman Provincial Coinage, http://rpc.ashmus.ox.ac.uk/coins/. Accessed 02 August 2018.

5 Digital Iconographic Atlas of Numismatics in Antiquity, http://ww2.

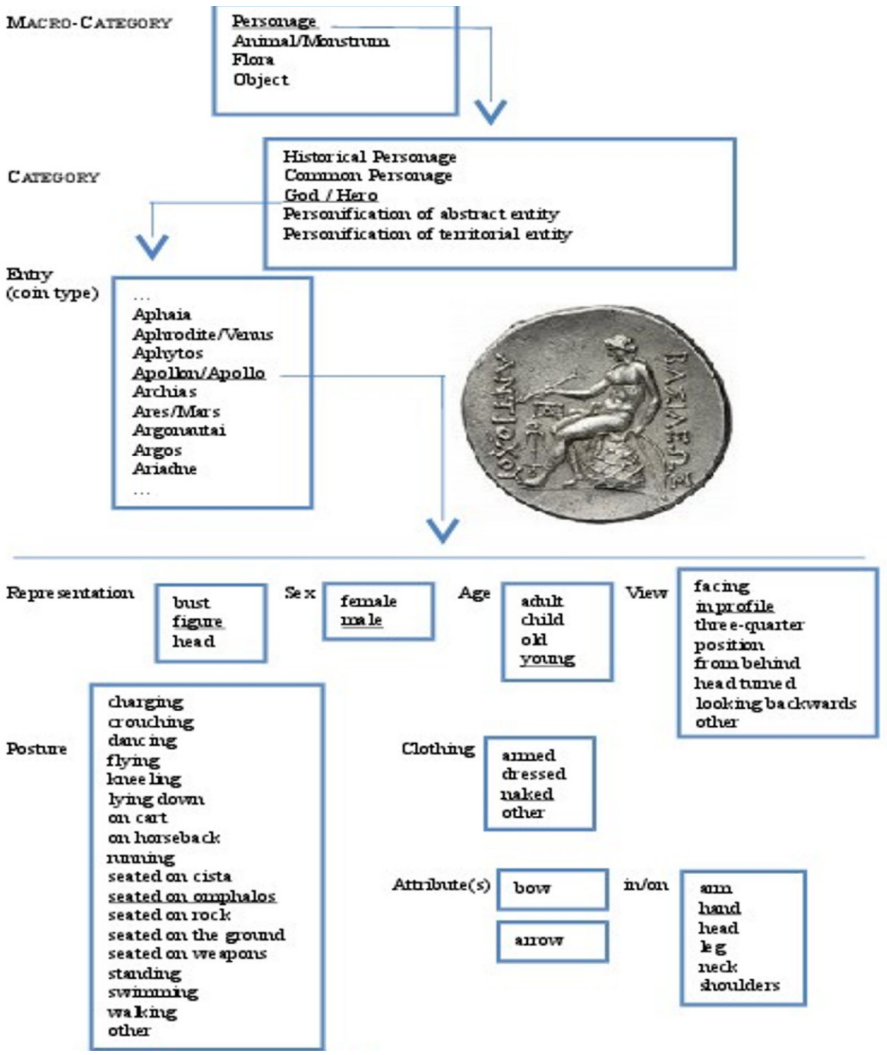

Fig. 2. The Digital Iconographic Atlas of Numismatics in Antiquity (DIANA) enables a hierarchical search. Here shown for: Personage – God/Hero – Apollo/Apollon – figure – male – young – seated_on_omphalos[6].

specialized iconographic searches which we will focus on in the following.

In the RPC Iconographic search[7] seven design groups are distinguished: animals, architecture, deities/personifications, games, heroes/famous persons, imperial family and objects. These design groups serve as a kind of pre-filter. After selecting one, one gets a list of predefined possibilities, e.g. for deities/personifications or key phrases like "Apollo with lyre", as shown in figure 1. Finally, relevant provinces can be selected from an analogously structured list, e.g. "Thrace". However, it must be stressed that the lists gets very long and confusing.

DIANA has a strong focus on the iconographic subjects displayed on coins. It distinguishes four iconographic types: personages, animals/mythical creatures, flora and objects. This classification is based on numismatic indices and lexicons[8]. DIANA enables a hierarchical search starting with the macro-category "personage" over "god/hero" to a proper name, e.g. "Apollo" as shown in figure 2. Furthermore, attributes like age, posture and clothing can be selected. For example, the posture "seated on omphalos" takes the relation of the person "Apollo" and the object "omphalos" into account. Attributes are based on lists within each category.

Of the search approaches discussed, DIANA's is the most sophisticated in respect to query semantics: it is structured hierarchically, distinguishes different iconographical types and takes certain relations between these types into account. However, this is enabled by manual work during the process of entering the coin data and their semantics manually. Therefore, this approach does not scale, nor does it improve automatically by learning new entries within the lists in the case that one wants to adopt the approach to new areas. Nevertheless, it was a great inspiration for our work, and the system of DIANA and our approach could even be combined in future.

Beside the numismatic approaches there are other possibilities of querying databases with ancient iconographic data. The *Lexicon Iconographicum Mythologicae Classicae* (LIMC) offers a basic online search tool for about 55,000 entries[9]. The available databases of the *Beazley Archive* at the University of Oxford[10] and the *Corpus Vasorum Antiquorum* (CVA) database[11] also provide good possibilities for querying ancient iconography. What these databases have in common is that they don't have an elaborated hierarchy of query terms. Furthermore, in few cases there is of course still the opportunity to consult the printed issues of these databases.

3. NATURAL LANGUAGE PROCESSING (NLP) PIPELINE

Our approach to enhancing numismatic search in terms of automation, scalability and flexibility is based on *Natural Language Processing (NLP)*. NLP is a field at the intersection of computer science, engineering and artificial intelligence. It deals with the challenges of teaching machines how to find relevant information in human language[12].

To make the most relevant information in a text accessible, two major tasks in information extraction have to be taken into account: *Named Entity Recognition (NER)*

unime.it/diana/. Accessed 02 August 2018.

6 CELESTI *et alii* 2017, 23.10 fig 6.

7 Roman Provincial Coinage, http://rpc.ashmus.ox.ac.uk/search/icono/. Accessed 02 August 2018.

8 CELESTI *et alii* 2017.

9 Lexicon Iconographicum Mythologicae Classicae (LIMC), http://www.limc.ch/public/Default.aspx?lang=en-US. Accessed 02 August 2018.

10 Classical Art Research Centre, http://www.beazley.ox.ac.uk/index.htm. Accessed 02 August 2018.

11 Corpus Vasorum Antiquorum (CVA), http://www.cvaonline.org/cva/default.htm. Accessed 02 August 2018.

12 SARKAR 2016.

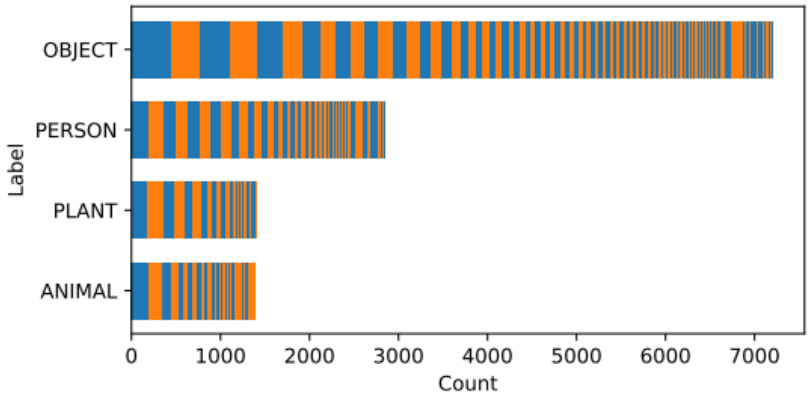

Fig. 3. The number of OBJECTs is substantially higher than for any other label. Each colour block corresponds to a unique term which occurs with different frequencies. The most frequent PERSON is "Apollo", the most frequent OBJECT is "bust".

and *Relation Extraction (RE). Named entities (NEs)* are definite noun phrases, e.g. names like "Apollo", that refer to specific types of individuals, e.g. persons, organizations or locations[14]. Relation extraction (RE) by definition means building triples of the form (NE_1, α, NE_2)-triples where α is the string of words intervening between the two NEs[15]. The relations we are interested in represent verbs connecting the subject with an object in a design.

Based on the categorization introduced by DIANA (cf. section 2), in a first step we distinguish four types of NEs: PERSONs, OBJECTs, ANIMALs and PLANTs. We started our work with the relations between PERSONs and OBJECTs because these labels are the most important ones for most numismatists and occur most frequently in the CNT designs: NEs of type OBJECT occur 7.201 times, whereas NEs of type PERSON occur 2.844 times. Figure 3 gives a rough overview of the distribution of NEs in the CNT designs.

The NLP approaches we use are based on supervised learning. For this, one needs data with already correctly assigned/labelled results (NEs or relations in our case). This data was split into training data used by the system to learn, and testing data used afterwards to test if the training was successful. The testing phase also provides the possibility of generating key indicators like accuracy, recall and precision[16]. Only if these key indicators provide good results should the trained system be used in practice. As mentioned above, our approach consists of the two steps NER and RE, which are coupled together in an NLP pipeline. The next subsection will provide an overview, before each of them will be explained in detail, including implementation aspects.

3.1 Overview of the NLP pipeline

The NLP pipeline consists of two steps: the NER identifies entities within the designs while possible relations between them are identified in the RE step (see figure 4). In the first step the NEs are assigned to one of four possible labels PERSON, OBJECT, ANIMAL or PLANT. The second step automatically identifies one of ten possible relations, e.g. "holding", "wearing", "resting_on" (cf. table 1), between pairs of (PERSON, OBJECT) NEs if appropriate.

Let's consider the NLP pipeline's workflow by means of the following example design:

"Apollo seated left on omphalos, holding bow in right hand."

After passing the NER step, "Apollo" is labelled as a PERSON and "omphalos" and "bow" are each labelled as OBJECTs:

"Apollo PERSON seated left on omphalos OBJECT, holding bow OBJECT in right hand."

After combining all NEs labelled as PERSONs or OBJECTs into possible pairs (cross-product), the (PERSON, OBJECT) pairs are taken into account as a candidate for the RE step:

[("Apollo seated left on omphalos, holding bow in right hand.", "Apollo", "omphalos"), ("Apollo seated left on omphalos, holding bow in right hand.", "Apollo", "bow")].

After passing the RE step, the output ideally would be as follows:

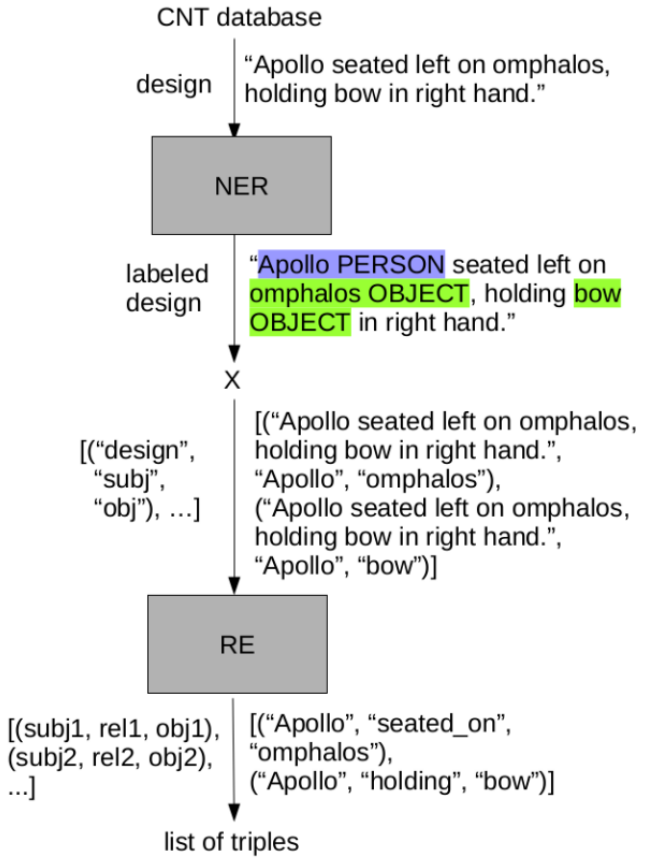

Fig. 4. Named Entity Recognition (NER) and Relation Extraction (RE) are implemented in two sequential steps in the NLP pipeline. The final output contains the subjects and objects, as well as the relations that exist between each (PERSON, OBJECT) pair in a design.

13 In some cases we deviate from the DIANA approach due to our focus on semantically linked entities. The "bust" is classified by DIANA as representation while we use it as 'OBJECT', because it should be linked with a person and should not be recognized as the person itself, which would generate results such as "bust wearing cuirass".

14 BIRD/KLEIN/LOPER 2009.

15 Ibidem.

16 SARKAR 2016.

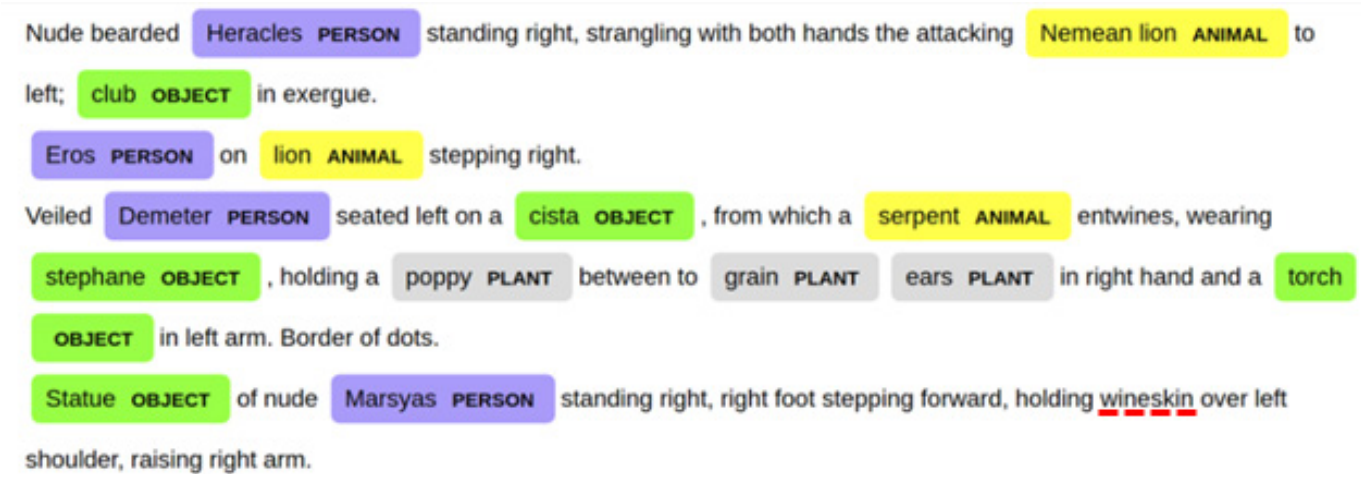

Fig. 5. Visualization of labelled designs. Each colour corresponds to a label: Purple represents PERSON, green OBJECT, yellow ANIMAL and grey PLANT. The object "wineskin" is a missing prediction that has not been recognized as OBJECT.

[("Apollo", "seated_on", "omphalos"), ("Apollo", "holding", "bow")].

"seated_on" and "holding" are the relations that exist between the person "Apollo" and the objects "omphalos" and "bow". The output of the NLP pipeline shows the subjects and objects of a design as well as the relations that exist between each (PERSON, OBJECT) pair.

3.2 Named Entity Recognition

3.2.1 Overview

In the NER step, the pipeline is trained on numismatic designs with automatically annotated labels.

The basis for these annotations are manually compiled lists of persons, objects, animals and plants. We created these lists by screening the CNT designs for synonyms, homonyms, alternative spellings and spelling errors. For example, "Apollo" occurs in the data also as "Apollo Citharoedus" or "Apollo Sauroctonos". These alternative names are also taken into account and will finally refer to the same entity.

After training, the pipeline is able to automatically recognize new, unseen NEs and label them into an corresponding entity-class. Suppose that "Apollo" is not part of the training data. After training, there is a high probability "Apollo" would be recognized as a PERSON.

The labelling of the designs can be visualized with the tool displaCy[17] (see figure 5). While depicting the correctly labelled designs, it also shows that the NLP approach might not attain 100% correctness. Figure 5 also shows one missing object: "wineskin" has not been recognized by the system. However, by improving the lists used for annotating the designs automatically for the training process, the probability of predicting NEs and correct labels can be increased.

3.2.2 Technical Details

We implemented NER by employing the NLP library spaCy[18]. To train the NE recognizer[19], a *supervised learning* approach was chosen[20]. Supervised learning algorithms take the annotated input data, i.e. the input data, together with the label to which the coherent data point shall be assigned, and return the trained classifier. After training, the classifier is tested with annotated input data which have not been involved in the preceding training process. This annotated data for testing is also called ground-truth. In our case the generation of the annotated data for the NEs was based on lists we had produced manually and had been improved by manual checking of the resulting output of the trained NER step. There had been several cases where name of a NE was missed or where spelling mistakes were encountered. This loop is illustrated in figure 6.

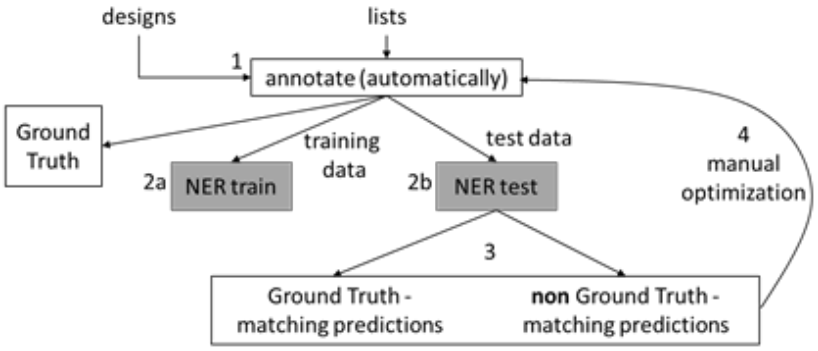

Fig. 6. Workflow of the training process and manual improvements for Named Entity Recognition (NER).

Viewed in more detail, the NER workflow consists of four steps. In **step 1**, the NE recognizer obtains the manually compiled lists of persons, objects, animals and plants which are used for annotating the CNT designs automatically.

17 displaCy Named Entity Visualizer, https://demos.explosion.ai/displacy-ent/. Accessed 02 August 2018.

18 spaCy: Industrial-Strength Natural Language Processing, https://spacy.io/. Accessed 02 August 2018.

19 spaCy: Industrial-Strength Natural Language Processing, https://spacy.io/api/entityrecognizer. Accessed 02 August 2018.

20 In comparison to supervised learning problems, *unsupervised learning* algorithms are used if the training data consists of a set of input vectors without any corresponding target values. For further discussion cf. BISHOP 2006.

Before training, the data was split into a training and a test set in order to evaluate the NE recognizer's performance after training.

In **step 2a** of the NER workflow, the NE recognizer is trained on the training data.

Afterwards, the trained NE recognizer receives the unseen, unlabelled test set and assigns the corresponding labels PERSON, OBJECT, ANIMAL or PLANT for the recognized NEs (**step 2b**). The NE recognizer's output is distinguished in ground truth (GT)-matching and non-ground truth (GT)-matching predictions in **step 3**. Predictions match the ground truth if the predicted labels equal the ground truth annotations and are non GT-matching otherwise. In **step 4**, the lists of persons, objects, animals and plants can be improved by reviewing the predictions manually.

To evaluate the NE recognizer's performance, we chose the evaluation metrics *accuracy*, *precision* and *recall*. *Accuracy* measures the percentage of annotated input data that has been labelled correctly by the classifier. *Recall* indicates how many of the relevant items were identified and *precision* indicates how many of the identified items are relevant, always compared to GT[21].

For the CNT design data, the NE recognizer achieves an *accuracy* of 88%, a *recall* of 97% and a *precision* of 98%. However, while these figures are very good, what figures are needed for which situation is another question.

The NER step was also tested on external data sets from *Online Coins of the Roman Empire (OCRE)*[22] – as mentioned above – and on data sets from *Coinage of the Roman Republic Online (CRRO)*[23] with new and different coin descriptions such as those for the coinage of the Roman Republic. The NER step was trained both on a CRRO and on an OCRE data set without any further adaptations, i.e. the lists were not adapted/improved for the external data sets, and the NER setup was the same as with the CNT data set. On the CRRO data set with 1.691 designs, the NEs are labelled correctly with a probability of 75%. On the OCRE data set with approximately 100.000 designs, the NEs are labelled correctly with a probability of 90%. This shows that the NER step generalizes from the CNT designs towards other numismatic data sets, and it suggests that the NER step gets better the more data is available for training.

It should also be mentioned that a figure of 100% is not easy to reach and any attempt to reach this could cause overfitting to the training data and therefore the trained model would not work well on new and different datasets. In section 6 we provide some results on applying our approach and how far it can help us.

3.3 Relation Extraction

3.3.1 Overview

After recognizing NEs in step one, we now concentrate on the relations between NE pairs. The question is: Which relations exist between a pair of NEs in a given design? Common methods for Relation Extraction (RE) can be distinguished into *knowledge-based methods* and *supervised methods*[24].

Knowledge-based methods are used for domain-specific tasks in which a fixed set of relations are to be extracted. Usually these methods are based on pattern-matching rules. Annotations and training are not necessary. However, knowledge-based methods usually do not generalize to other data sets and require a lot of manual labour[25].

Supervised methods automatically learn how to extract relations and therefore easily adapt to other data sets. The drawback of supervised methods is that a large amount of training data is needed and it takes a lot of manual labour to produce the labelled data required for training[26].

A challenging trade-off in Relation Extraction is the one between manual labour and automatic learning. In the case of CNT, on the one hand the data set was too small for a supervised learning approach and it was impracticable to manually annotate the whole data set. On the other hand, a knowledge-based approach would not have generalized to other data sets. All things considered, the existing approaches did not suit our needs.

Therefore we selected a different approach based on a fixed number of relations. If the number of relations is known, instead of asking: What relations exist between a pair of NEs?, the question can be rephrased into: Does a pair of NEs have a certain relation or not? This way RE can be understood as a *multiclass classification* problem where each relation represents one class[27]. Creating training data then becomes comparatively easy because only the relations have to be annotated. However, the drawback is that new relations cannot be learned as the number of relations is fixed.

By building semantic clusters, we identified ten relations in the CNT designs (see table 1). These relations are the classes with which the RE step is trained to automatically recognize whether a pair of NEs has a certain relation or not. After training, the percentage of relevant items that were identified and the percentage of identified items that were relevant were both high. For the CNT designs, the percentage of relevant items that were identified was 84% and the percentage of identified items that were relevant was 92%.

3.3.2 Technical Details

For the Relation Extraction within our NLP pipeline, the numismatic designs together with the PERSONs and OBJECTs detected in the NER step serve as input. In order to train this Relation Extraction classifier we used the workflow as shown in figure 7.

In **step 1**, the designs are annotated manually with relations between (PERSON, OBJECT) pairs. For the CNT data it took about four working days (32 hours) of manual work to perform this task. In **step 2** of the RE training workflow, features are extracted from the annotated relations (see figure 7). These features serve as input for the training process. A feature is a unique, measurable attribute or property for each data point in a data set[28]. In

21 BIRD/KLEIN/LOPER 2009.

22 Online Coins of the Roman Empire, http://numismatics.org/ocre/. Accessed 02 August 2018.

23 Coinage of the Roman Republic Online, http://numismatics.org/crro/. Accessed 02 August 2018.

24 KONSTANTINOVA 2014.

25 KONSTANTINOVA 2014.

26 KONSTANTINOVA 2014.

27 BISHOP 2006.

28 SARKAR 2016.

Table 1. The eleven relations between (PERSON, OBJECT) pairs represent semantic clusters. no_ existing_ relation is used for designs which lack any such relation. Words in brackets, e.g. "drawing" (arrow), indicate what the verb in the semantic cluster is commonly used with.

Relation	Semantic Cluster
holding	"holding", "carrying" (garment), "brandishing" (spear), "shouldering" (rudder), "playing" (lyre, aulos), "raising" (shield), "cradling" (torch)
wearing	"wearing", "covered with" (lion-skin)
resting_on	"resting on", "leaning on"
seated_on	"seated on", "seated in"
standing	"standing in" (biga), "driving" (biga), "standing on" (galley)
drawing	"drawing" (arrow)
stepping_on	"stepping on" (helmet)
grasping	"scooping" (gold), "reach out for" (person), "plucking" (chiton)
lying	"lying on"
hurling	"hurling" (thunderbolt)
no_existing_relation	

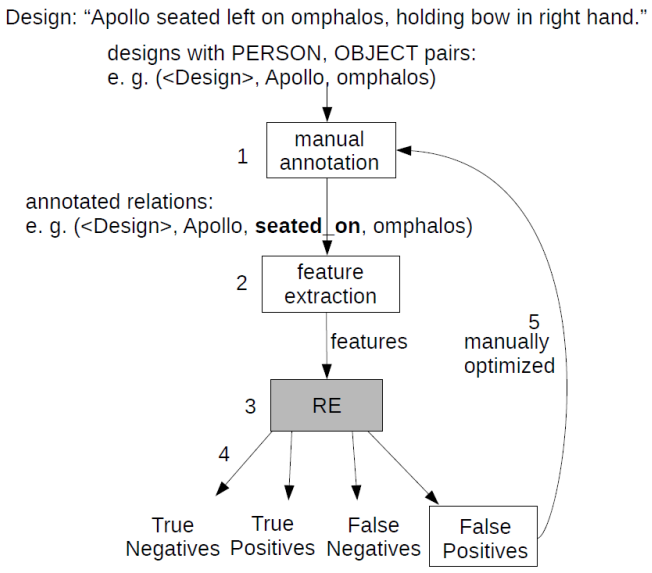

Fig. 7. In the Relation Extraction (RE) training workflow, annotated relations are transformed into features for training the Relation Extraction classifier. The False Positives can be analysed for further optimizing the annotated relations.

step 3, the extracted features are used as input for training the RE classifier. The output of the best feature-classifier-combination is split in *True Positives*, *True Negatives*, *False Positives* and *False Negatives* in **step 4** of the workflow. If we let *p* denote the positive (correct) and *n* the negative (wrong) class, then the *True Positives* are the number of instances correctly predicted and belonging to *p*. The *True Negatives* also belong to *p*, in our case where no relation was found and this being correct. The *False Negatives* are defined as the number of instances falsely predicted and belonging to *n*. In our case this means that the relation was not identified by RE even if there was a relation. The *False Positives* are the number of instances falsely predicted compared to our manual annotation and belonging to *n*[29]. However, this last class of results represents the cases where there is a relation and RE found it but our underlying annotation is wrong. This happens in cases which were forgotten or something was mistaken in the manual annotation. The goal of this distinction is to analyse the *False Positives* in order to correct these errors in **step 5**.

By performing a cross-validated grid search with five repetitions (S = 5), we identified the best feature-classifier-combinations. We repeated the process five times in order to avoid bias. Each run took about one hour on a normal desktop machine.

In a *grid search*, different parameter combinations are probed over a grid. The performance of each parameter combination is evaluated by cross-validation[30]. For *cross-validation*, the data set is split into S equally sized parts. A proportion (S – 1)/S is used for training and the remaining proportion 1/S for testing. This procedure is repeated for all S possible choices of the combination of test and training proportions[31].

The best performing classifiers in our case were either Logistic Regression or Support Vector Machines with a linear kernel. We implemented a feature that extracts the path on the parse tree between two NEs, i.e. between the subject and the object. The best feature-classifier-combinations use this feature together with the vectorizer bag of words, which counts the frequency of each word appearing in a text[32]. The bag of words is used with 2- or 3-grams, i.e. sequences of tokens of length two or three.

To evaluate the RE classifier's performance, the metrics *precision*, *recall* and *F1 score* were chosen. The *F1 score* is the harmonic mean of *precision* and *recall*[33]. For the CNT designs, the RE classifier achieved a *precision* of 92%, a *recall* of 84% and an *F1 score* of 88%. It must be stressed that the RE performance is also limited by the NER performance because the steps are implemented sequentially.

Given the RE classifier has been trained on the annotated CNT designs, relations can be predicted for external data sets. We predicted relations on the OCRE data set with approximately 100.000 designs without any further adaptations, i.e. the NLP pipeline setup was the same as with the CNT data set. Due to the lack of annotated training data, it is not possible yet to evaluate the RE classifier's performance in terms of evaluation scores. However, the

29 Definition according to SARKAR 2016.
30 PEDREGOSA *et alii* 2011.
31 BISHOP 2006.
32 SARKAR 2016.
33 SARKAR 2016.

predictions show that the RE step generalizes from the CNT designs towards other numismatic data sets. Further more practical observations on the OCRE data can be found in section 6.

4 HIERARCHIES OF NAMED ENTITIES

With the output of the NLP pipeline, one can retrieve all designs containing triples of the form of "subject -- relation --> object", e.g. "Apollo -- holding --> bow". By defining a hierarchy within each NE-type more sophisticated queries like "deity -- holding -->arms" are allowed. The outcome of this query should contain all designs where a deity is holding a weapon or a shield. This would include the design example of Apollo from the previous section. Through the designs one then could retrieve the appropriate coins.

The main advantage here is that, in contrast to exploring data by querying, where semantically sharpened queries might bring too few results, with the possibility of climbing up the hierarchies for subjects or objects in the query, one can get more results and a better overview of the content of the data.

Let us compare this with existing systems like OCRE. The OCRE search tool offers the possibility of searching for keyword and/or fixed categories like "deity"[34]. But there is no common category for something like "arms", so the keyword search has to be used and the query specified with the term "bow" or something similar. The outcome here will be limited and more complex queries have to be employed. Our approach offers the possibility of climbing within the treelike hierarchy graph to the node "arms", and thus achieving a better overview with more results.

There are further issues with keyword searches like that used in OCRE due to potentially missing semantic relations (cf. section 6). We transform the output of the NLP pipeline and the NEs hierarchies into RDF[35] (cf. section 5), which can be queried very flexibly with SPARQL[36]. In this way a query for designs containing "emperor and club" like in a keyword search is also still possible.

4.1 Hierarchies spreadsheet and its transformation to database tables

The first step in generating a hierarchy for our CNT data was a manual search by our domain experts for NEs in the CNT database. The results were recorded in corresponding spreadsheets. Subsequently, our domain experts applied an already existing hierarchy from the CNT project to the entries. During this process the hierarchy was further supplemented. In the next step the spreadsheets were converted into tables in a relational database. These tables were also the basis for training the NER and RE pipelines (cf. section 3).

Beside the NEs lists, the classes and its subclasses are stored in a separate hierarchy table. This structure represents one tree for each NE type, meaning each class has exactly one super class. The only exceptions are the root classes for each NE type, like "person" or "object", which do not have a super class defined. This offers the possibility climbing up or down the tree depending on your type of query.

In contrast to the tree structure of the class hierarchy, the NEs themselves can be assigned to more than one class. As illustrated in figure 8 the NE *Apollo* is assigned to three classes of the person hierarchy.

To implement the instantiation of the NEs to multiple classes, we designed the appropriate tables as follows: Every table representing a NEs type and containing entries for the appropriate NEs contains a variable number of category columns. Each of these columns represent a set of classes of the hierarchy that should be disjoint, e.g. *cnt:Male* and *cnt:Female*. This is represented in figure 8 by the different colours of the classes.

4.2 Hierarchy examples

As an example of our hierarchy and its structure we take the following design, which was introduced above (cf. section 1 and 3.1):

"Apollo seated left on omphalos, holding bow in right hand."

The design was labelled by the NER pipeline this way: "Apollo" is a PERSON, "omphalos" and "bow" are OBJECTS (cf. section 3.1). We take these labels as a base for applying our hierarchy. Let us start with the person "Apollo". He is the Roman equivalent of the Greek god "Apollon" and one of the twelve Olympian gods. On this account we assigned the classes "Male", "Roman" and "Olympian" (see figure 8)[37]. The further super classes "Deities" and "Person" are inferred by the class hierarchy tree.

The categorization of the objects "omphalos" and "bow", which reside in the object table of our database, are simpler. Both items are assigned to only one further category: "omphalos" is a "Stone" object, because of its Greek mythological origin as stone which marks the "navel" of the world at Delphi, where the oracle of "Apollo" is based[38]. As patron deity of archery, the god is armed with a bow[39]. Thus, the "bow" object is a part of the "Arms" class. Furthermore, the RE pipeline extracted the two relations which associate the PERSON and OBJECT entities. These relations are "holding" and "seated on", and both link "Apollo" to one of the objects.

5. TRANSLATION TO *RDF*

This section concentrates on how we translated the output of the NLP pipeline (cf. section 3) and the hierarchies (cf. section 4) into the *Resource Description Framework* (RDF)[40]. This was done in order to use *SPARQL* to query the results[41].

For the translation of our relational tables to RDF a mapping file was needed, which was developed via the D2RQ Mapping Language[42] and tested with a D2R server[43].

34 Online Coins of the Roman Empire, http://numismatics.org/ocre/search/. Accessed 09 May 2018.

35 Resource Description Framework, https://www.w3.org/RDF/. Accessed 16 May 2018

36 SPARQL Protocol And RDF Query Language, https://www.w3.org/TR/rdf-sparql-query/. Accessed 16 May 2018.

37 GRAF 2009, 14, 27.

38 MAAß 1993, 1.

39 GRAF 2009, 13-14, 27-28; LAMBRINUDAKIS et alii 1984.

40 Resource Description Framework, https://www.w3.org/RDF/. Accessed 16 May 2018.

41 SPARQL Protocol and RDF Query Language, https://www.w3.org/TR/rdf-sparql-query/. Accessed 16 May 2018.

42 D2RQ, http://d2rq.org/d2rq-language. Accessed 19 June 2018.

43 D2RQ, http://d2rq.org/d2r-server. Accessed 19 June 2018.

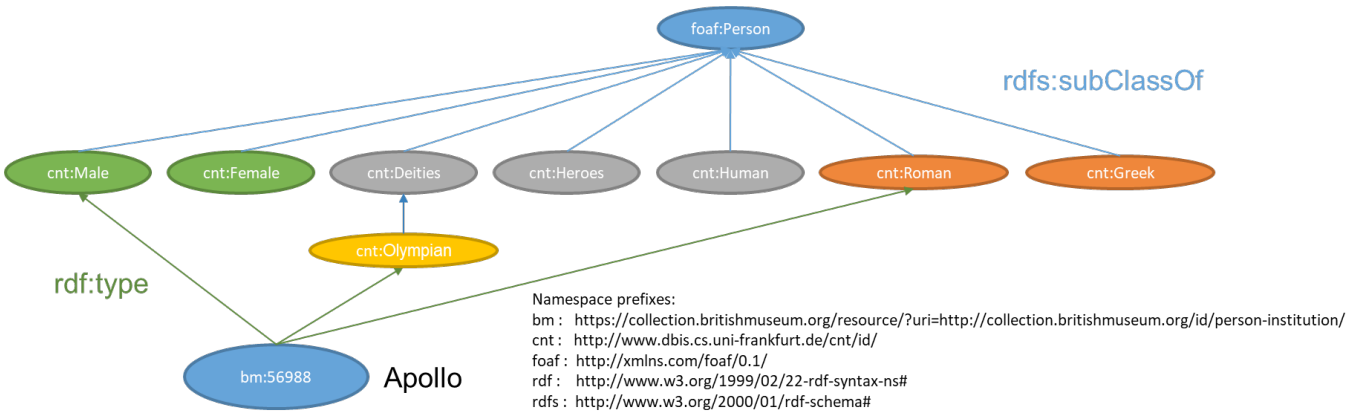

Fig. 8. The class hierarchy defined by *rdfs:subClassOf* and instantiation to classes with *rdf:type*. All classes are represented by their RDF names (cf. section 5) with namespace prefixes. The example shows: *Apollo* as an instance of the class *cnt:Olympian*, which is a subclass of *cnt:Deities*. Moreover, *cnt:Deities* is a subclass of *foaf:Person*. By inference *Apollo* is also an instance of these super classes.

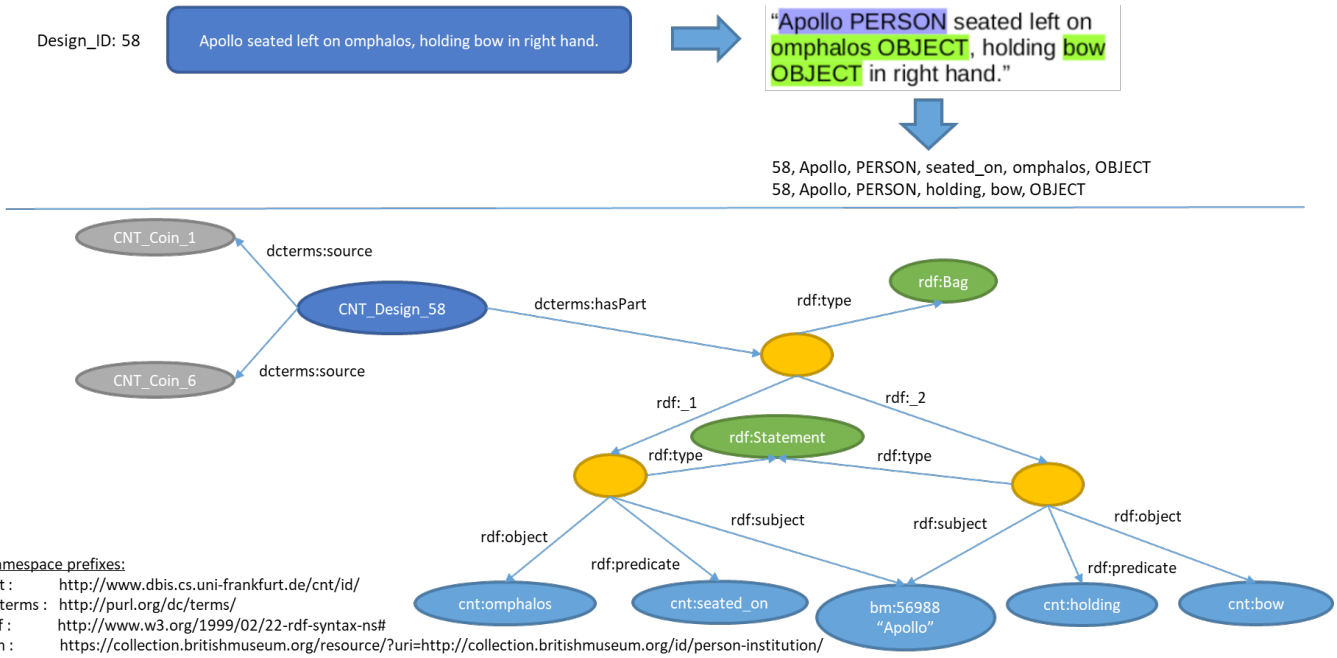

Fig. 9. Shows the translation of the NLP pipeline output into RDF, together with its links to the design and the coins using this design.

Furthermore, via the mapping file the D2R server produced a RDF dump of our approach's results.

The translation of the hierarchy is visualized basically in figure 8. The classes are linked by rdfs:subClassOf and the instantiation by rdf:type properties. Multi inheritance of the instances is supported in RDF.

More complex was the translation of the output of the NLP pipeline. This output already represents triples that are assigned to a certain coin design. This means a design can have a set of triples consisting of a subject, an object and a matching relation assigned between them. Therefore, we chose the rdf:Bag container to store this set. For the triples themselves we used the standard reification vocabulary of RDF[44], where each statement is represented by a blank node with links to the subject, predicate and object of the statement. An example of this translation is shown in figure 9.

With an addition to our pipeline we also store all identified NE for each design in the database, since otherwise NEs that do not have a relation would be lost. They are translated to RDF into a second rdf:Bag. This way, one can find results for coin descriptions which consist of only one NE or have no relation between several identified NEs.

6. EVALUATION AND FIRST RESULTS ON *OCRE*-DATA

One of our goals was to generate a generic approach that can easily be adopted to other data sets. Therefore, we performed a number of experiments on the OCRE data. For these experiments, we used our NLP pipeline that was trained purely on CNT data (cf. section 3) on it. The OCRE data is already available in RDF format with Nomisma.org ontology on the ANS web portal[45]. The results were transformed into

44 Resource Description Framework, https://www.w3.org/TR/rdf-schema/#ch_reificationvocab. Accessed 04 July 2018.

45 American Numismatic Society, http://numismatics.org/ocre/nomisma.rdf. Accessed 19 June 2018.

RDF and loaded to the Apache Jena Fuseki SPARQL server[46]. Then we executed three experiments with different goals (cf. Table 2) and analysed the results:

Table 2. Overview of the experiments and goals to be achieved.

Experiment	Query type	Goals
1	Basic - does it work?	Comparison between the OCRE search tool and our approach
2	Higher semantical precision	Can our approach be more precise and in this way provide more focused result sets?
3	Hierarchical flexibility	Can our approach be more generic and in this way provide more high-level result sets?

Experiment 1: Does it work? – Comparison between OCRE search and our approach

The goal in this experiment was a direct comparison of the results between the OCRE search tool[47] and our approach. Therefore, we defined a query that could be executed on both systems with no further adaptations: retrieve all coin types containing "Apollo" and "Bow". OCRE returned twelve results (coin types) for this query. More results are available (subtypes) by clicking on the individual entries. All in all, we had a resulting sum of twenty-six types and subtypes. After linking our RDF file to the OCRE data, we used a SPARQL query on our model that looks for "Apollo" as *rdf:subject* and "Bow" as *rdf:object* in the appropriate RDF file, in this way allowing any relation between the two. The outcome contained twenty-four types and subtypes of the OCRE search. There were two missing types, which show Apollo and Diana with a bow in front of them, 'RIC V Postumus 263' and 'RIC V Postumus 338'. Our model assigned the bow to Diana as the textual description does, and therefore these types were not included in the result. All results of both systems were relevant. We conclude that for this case our system works satisfactorily and as expected.

Experiment 2: Higher semantical precision

One weakness of the OCRE search tool is that the search is performed on the complete type description, including obverse and reverse descriptions at the same time. A query for depictions of the emperor Caracalla and a club together on one face of the coin cannot be done with the existing OCRE keyword search (however, it could be done by a SPARQL query on the RDF dump of the OCRE data). The current outcome of the OCRE keyword search for "Caracalla" and "club" consists of fourteen entries, but all of them show Caracalla on the obverse and Hercules with a club on the reverse, e. g. 'RIC IV Caracalla 74A (denarius)' and 'RIC IV Caracalla 206b'. A SPARQL query with these terms together on one face of the coin on the results of our model did not return any results. As the first query has shown, there are no depictions of the emperor Caracalla with a club on a single face of a coin in the OCRE data. This was confirmed also by our approach. This means our model can focus on the design on one face of the coin for searching subjects, objects and relations which are semantically linked, so we can retrieve fewer results with a higher relevance. This possibility is important in cases where otherwise have too many results would be returned. For example, the reverse design of 'RIC IV Caracalla 13A (denarius)' shows the emperor Caracalla beside a military trophy. There is no semantical link between them and hence our approach does not provide a result for that.

Experiment 3: Hierarchical flexibility

The last experiment should show that our approach allows using a hierarchy within the query terms. Instead of using well-defined keywords for searching the OCRE data, we employ our hierarchy terms (c.f. section 4). Therefore, the following query-terms were used: "emperor -- holding --> arms". The results should include Roman emperors who are holding a weapon or a shield. Since these hierarchies are not part of OCRE, this cannot be executed within the OCRE search tool, nor with a SPARQL query on the available OCRE dump. In contrast, our model gives us around 1200 results in 100,000 coin designs (around 52,000 coin types in Scenario 0, cf. Table 3). Since it would be too time consuming for our domain experts to check all of them, we restricted the evaluation to two scenarios where we added time range conditions in order to gain a manageable result set. To enable this, we linked our resulting RDF file with the OCRE RDF dump in Fuseki (by simply uploading the dump. Since the URLs of the OCRE types are identical, the linkage is done automatically by the graph database). Both scenarios cover a period in the reigns of two successive emperors:

Scenario 1: Trajan and Hadrian, AD 107-127.

This scenario contains 1025 coin types and provides 37 results for "emperor -- holding --> arms". After a manual check we can say that 27 of the 37 results are correct and all the relevant results were found. Thus, we got a precision of 73% and a recall of 100%. A depiction of soldiers and officers together with an emperor is responsible for the ten wrong results. The problem here is that the emperor's military escort hold their own arms such as spears and shields, and that this wasn't recognized by the NER.

Scenario 2: Elagabalus and Alexander Severus, AD 219-224.

For this timespan there are 639 coin types that produced 41 results for our query. Again, our domain experts did a manual check on all 639 coin types. The 41 results contain all relevant results, and only 3 coin types where false positive (meaning the expert would not include them to the result), which corresponds to a precision score of 92.7% and a recall of 100%. Again, all relevant types were found and an incorrect assignment to the emperor of a shield on the ground is responsible for the three errors.

Example types:

Correct result: Scenario 1: 'RIC II Hadrian 1008 (dupondius)'; Scenario 2: 'RIC IV Elagabalus 181'

Wrong result: Scenario 1: 'RIC II Trajan 662'; Scenario

46 Apachi Jena Fuseki, https://jena.apache.org/documentation/fuseki2/index.html. Accessed 20 June 2018.

47 Online Coins of the Roman Empire, http://numismatics.org/ocre/search/. Accessed 14 June 2018.

Table 3. Overview of all scenarios and their particular query results before and after the manual check by domain experts with the resulting precision and recall score.

Scenario	Number of Coin Types	Time Span	Query Results	Correct Results (manual check)	Precision	Recall
0	51,555	31 BC - 491 AD	1,205	-	-	-
1	1,025	107 - 127 AD	37	27	73%	100%
2	639	219 - 224 AD	41	38	92.7%	100%

2: 'RIC IV Severus Alexander 555'

These two scenarios demonstrate the potential of our approach to facilitate more flexible queries on a higher level of hierarchy. Especially in cases where a distinction between groups of persons and/or objects is desirable, this is very helpful. Thus, more relevant results can be obtained than with a simple keyword search. However, it must be stressed that the NLP pipeline was only trained on CNT data, and were the model further adopted to the OCRE use case, the results probably would improve. However, it also shows that a 100% *precision* score is difficult to reach (c.f. Table 3).

RECAPITULATION

As can be seen from our experiments, the model trained on CNT data can be used on the OCRE textual design descriptions with no further adaptations. It also allows a higher semantical precision in cases where there are too many results, and a hierarchical flexibility in cases one wants to query on a higher level in order to gain more results.

However, there is still work to be done. Due to the fact that our annotation step focuses on the CNT data several subjects, such as the emperor Vespasian or groups of persons like soldiers, aren't found yet in the OCRE data. The same is true for objects and relations. This also has an impact on the fluctuating recall score in experiment three. Further work on the underlying lists and training the model with a custom OCRE annotation would surely stabilize the score on a high level and enable searching for previously unknown entities.

7 Conclusion

Our work, and especially the evaluations, show that our NLP approach generates results that can be the basis for analysing research questions and hypotheses. With the semantic relations we extract from existing text, and the implemented hierarchies for known persons and objects, we enable a semantic search that goes beyond the possibilities of normal keyword searches. One current limitation is the fixed number of relations our approach currently can handle. This is partly due to the fact that extensive annotated corpora with annotations of relations and entities for the numismatic domain do not exist.

We also demonstrated that our approach is generic enough to be applied to new datasets, as is shown with OCRE data. Of course, it is helpful if the datasets to be analysed are semantically similar to the data used to train the NLP pipeline. Currently we are improving our lists in order to train the model on the union of both datasets, CNT and OCRE, since our experience also showed that bigger training sets generate better results. We therefore hope to improve the quality of our results. It must be stressed at this point that the output of the NLP pipeline might contain false positive and false negative results (errors). However, a manual extraction of the relevant data would simply be too time consuming and could contain mistakes as well.

So far, we generated the data from the NLP pipeline and transformed it into RDF. This allowed us to query the data via SPARQL queries. For domain experts like numismatists or archaeologist, a more intuitive query interface needs to be constructed. DIANA currently provides in our view the most advanced graphical user interface (GUI) for semantic queries on iconographies, at least for the numismatic space. By mapping our results into the internal model of DIANA its GUI could be easily adopted.

Another interesting next step we plan is to apply our approach on datasets outside the field of numismatics, e.g. pottery, not only to answer research questions in other domains, but also to build bridges between these domains and to answer questions that combine these different fields.

ACKNOWLEDGEMENTS

This work was partially funded by the Deutsche Forschungsgemeinschaft (DFG) under the project title: "Corpus Nummorum Thracorum. Klassifizierung der Münztypen und semantische Vernetzung über Nomisma.org". More details under: https://www.corpus-nummorum.eu/.

REFERENCES

BIRD/KLEIN/LOPER 2009
Bird, St./ Klein, E./ Loper, E., *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit* (Sebastopol: O'Reilly).

BISHOP 2006
Bishop, Chr. M., *Pattern Recognition and Machine Learning* (New York: Springer).

CELESTI *et alii* 2017
Celesti, A./ Salamone G./ Sapienza A./ Spinelli M./ Puglisi M./ Calatabiano M., An Innovative Cloud-Based System for the Diachronic Analysis in Numismatics. DOI: 10.1145/3084546 [*Journal on Computing and Cultural Heritage* 10/4, 23.1– 23.18, October 2017].

GRAF 2009
Graf, F., *Apollo* (Abingdon: Routledge).

KEMKES 2013
Kemkes, M., Caracalla - Kaiser, Tyrann, Feldherr. In: Archäologisches Landesmuseum Baden-Württemberg (ed.), *Caracalla. Kaiser, Tyrann, Feldherr* (Darmstadt), 7-32.

KONSTANTINOVA 2014
Konstantinova, N., Review of Relation Extraction Methods: What Is New Out There?. DOI: 10.1007/978-3-319-12580-0_2. [In: Ignatov, D./ Khachai, M. Yu./ Panchenko, A. / Konstantinova, N./ Yavorskiy, R. E. (eds.), *Analysis of Images, Social Networks and Texts. Third International Conference, AIST 2014, Yekaterinburg, Russia, April 10-12,*

2014 Revised Selected Papers (New York), 15–28].

MAAß 1993

Maaß, M., *Das antike Delphi. Orakel, Schätze und Monumente* (Darmstadt: Wissenschaftliche Buchgesellschaft).

LAMBRINUDAKIS *et alii* 1984

Lambrinudakis, W. et al, s. v. Apollon, Lexicon Iconographicum Mythologiae Classicae, II/1, Zürich/ München, 183–326

PEDREGOSA *et alii* 2011

Pedregosa, F./ Varoquaux, G./ Gramfort, A./ Michel, V./ Thirion, B./ Grisel, O./ Blondel, M./ Prettenhofer, P./ Weiss, R./ Dubourg, V./ Vanderplas, J./ Passos, A./ Cournapeau, D./ Brucher, M./ Perrot, M./ Duchesnav, E., Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research* 12, 2825–2830.

SARKAR 2016

Sarkar, D., *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data* (New York: Apress)