Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 43–52
DOI: 10.15439/2022F36
ISSN 2300-5963 ACSIS, Vol. 30

A novel link prediction approach on clinical knowledge graphs utilising graph structures

Jens Dörpinghaus*†§, Tobias Hübenthal‡§, Jennifer Faber†
* Federal Institute for Vocational Education and Training (BIBB), Bonn, Germany,
Email: jens.doerpinghaus@bibb.de, https://orcid.org/0000-0003-0245-7752
† German Center for Neurodegenerative Diseases (DZNE), Bonn, Germany
‡ Department of Mathematics and Computer Science, University of Cologne, Germany
§ These authors contributed equally.

***Abstract*—This paper presents a novel approach towards link prediction in clinical knowledge graphs. They play a central role in linking data from different data sources and are widely used in big data integration, especially for connecting data from different domains. We present a knowledge graph initially built on data from a clinical trial on Spinocerebellar ataxia type 3 (SCA3), which is a rare autosomal dominant inherited disorder. The contributions of this paper are (1) to create a feasible data representation schema capable of handling clinical imaging data in a knowledge graph and to (2) convert the data efficiently into a knowledge graph. Due to the limited amount of patient-nodes usually common methods for link prediction and graph embeddings are problematic and thus we will (3) present a novel approach for link prediction utilising graph structures and Conditional Random Fields. In addition, we present (4) an extensive evaluation underlining the importance of (a) data management and (b) further research on link prediction using graph structures.**

I. Introduction

KNOWLEDGE graphs have been shown to play an important role in recent knowledge mining settings, for example in the fields of life sciences or bioinformatics. Contextual information is widely used for NLP and knowledge discovery tasks since it highly influences the exact meaning of expressions and also queries on data. Here we will present some results on link prediction in knowledge graphs in the field of personalised medicine which aims for matching certain risk groups and possibly yet unknown subgroups to treatments, ultimately optimising patients' responses, mainly to available drugs. For this purpose, collected primary data of the examined persons have to be linked with data from secondary sources like publications or databases in an application-oriented way.

As part of the European Spinocerebellar Ataxia Type 3 Initiative (ESMI), SCA3 mutation carriers, their first-degree relatives, and healthy controls were prospectively studied using standardised clinical assessment as well as MRI imaging and biosampling.

Spinocerebellar ataxia type 3 (SCA3) is a rare autosomal dominant inherited disorder. The onset of the disease is in adulthood. Patients develop ataxia, which is a disorder of coordination of target movements that affects gait, fine motor skills and speech. The disease is progressive and patients in the advanced stages are usually dependent on the use of first a walking aid and later a wheelchair. Not only the gait disorder has a strong influence on everyday activities. Also the independent preparation of meals, tool use of e.g. eating utensils and an increasingly unclear speech severely restrict the patients in their everyday life. Although SCA3 mutation carriers are not yet symptomatic, disease activity is already evident, for example, in atrophy of certain areas of the brain where neuropathological changes are predominant, as well as elevated blood levels of non-specific markers for neuron loss. The data set contains not only patient data but also digital imaging data [1], [2].

The goals of this paper are (1) to create a feasible data representation schema capable of handling clinical imaging data in a knowledge graph, see Figure 1, and to (2) convert the data efficiently into a knowledge graph. Since the overall amount of participants in clinical trials is usually not high, employing common methods for link prediction and graph embeddings is problematic [3]. We will (3) present a novel approach to link prediction utilising graph structures and (4) its evaluation.

This paper is divided into six sections. After an introduction, the second section gives a brief overview of the state of the art, related work and backgrounds used for our novel approach. Therefore, we will refer to both knowledge graphs and dedicated algorithms. In the third section, we present our approaches regarding data integration and data schema. The fourth section describes the novel approach to link prediction, with the experimental results on both artificial and real-world scenarios in the subsequent section.

Our conclusions and outlooks are drawn in the final section. We will propose a novel CRF-field based approach which presents promising performance. While the results at first glance do not seem to be a significant improvement for new algorithms for knowledge discovery on clinical data they clearly show the importance of (a) data management and (b) further research on link prediction using graph structures. We also provide a short outlook for extensions of our work.

II. Related work and Background

Clinical research is more and more relying on data-intensive approaches, thus facing increasingly complex challenges. Expert systems, for example, provide users with several methods for knowledge discovery. They are widely used to find relevant

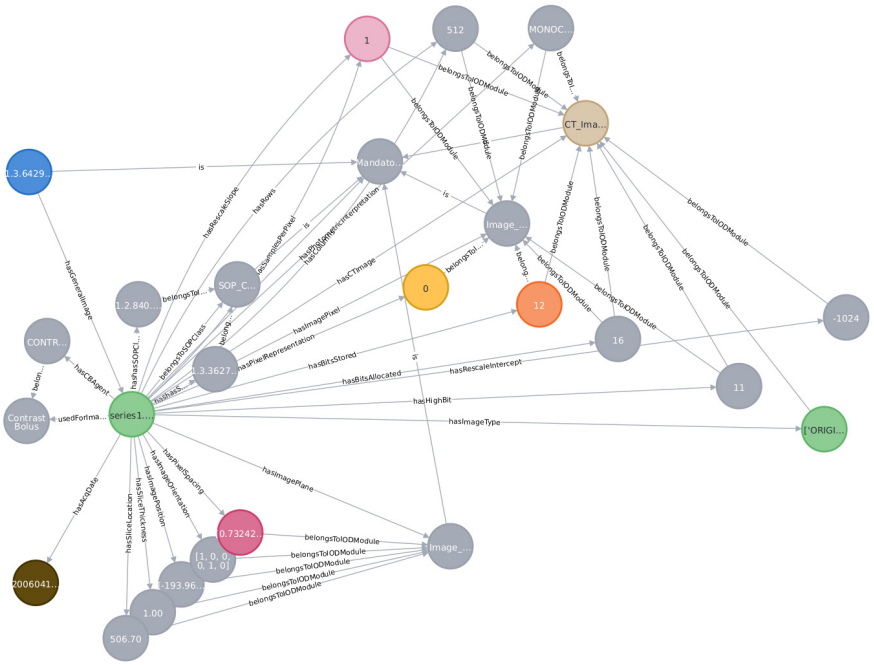

Fig. 1. An example of the nodes and edges in the graph database after import is complete: a `General Image` node and its immediate neighbourhood are shown.

or novel information. A popular example in biomedical research is the attempt to find molecular pathways; controlled reaction mechanisms within biological organisms, which might be misregulated in pathogenic states. Obviously, understanding these cascades, their players and relations to diseases is key to designing and applying drugs in a targeted way.

Being confronted with patients' clinical data and with expert knowledge in the back of their minds, clinical researchers usually consider an initial idea and start integrating external content such as scientific papers. The most common approach is inquiring with a search engine about some terms to find closely related information. Effectively, users most frequently query for additional documents or patient files to adjust the search query. Similarly, for a given set of documents or patients the question might be on commonalities considering a certain topic. Both approaches are heavily related to the context of data points, see for example [4] for *PubMed* data. Topic labelling – or cluster labelling – and longitudinal data are constantly being explored in several research fields, see [5], [6], [7].

Graph structures and in particular knowledge graphs provide several advantages for the integration of knowledge and its targeted re-extraction. According to their generic character such integrative knowledge graphs are important for life sciences, medical research and associated fields, not least by supporting their interconnection on a formal level. Considering systems medicine applications, knowledge graphs provide grounds for holistic approaches unravelling disease mechanisms. In these and other common settings pathway databases play an important role. As a basis, biomedical literature and text mining are used to build knowledge graphs, see [8]. As part of the studies on integrative data semantics within clinical research, data on patients suffering from certain diseases have been collected by various institutions. Some studies also integrate data from several databases and ontologies which can implicitly form a knowledge graph. For example *Gene Ontology*, see [9], *DrugBank*, see [10] or [11] cover large amounts of relations and references which other fields can refer to.

Link prediction on graphs, for example on knowledge graphs and social networks, is usually done using embeddings which form a low-dimensional representation of the graph. The main assumption is that they provide an accurate reconstruct of the graph, see [3], [12], [13]. In general, factorisation, random walk and deep learning approaches are used, see [14]. While these approaches have also been considered for applications in other domains, see for example [15], a general approach towards learning links solely based on graph structures is yet missing. Some researchers have tried to propose features based on graph structures and found promising results based on a large amount of features modelling "different aspects of the graph structure", see [16]. According to our knowledge no more work has been carried out in this field.

In [17] 27 real world clinical questions and queries in scientific projects were collected to test the performance and output of the knowledge graph. The authors revealed that the performance of several queries was very poor and some of them did not even terminate. Their research is based on biomedical knowledge graphs as described in [8] containing mainly scientific documents and extracted data from *PubMed*. In [18] this approach

was generalised to achieve interoperability with clinical data.

In our case, the first step taken is the integration of data from a very complex clinical trial. We will provide a data integration schema in the next section. The data schema should be capable of further data integration, for example *Gene Ontology* or data from scientific documents like *PubMed*. The software importer should be as generic as possible to work on multiple data sources. This helps to provide experimental results on data which is not affected by data protection regulations.

The experimental results are carried out using a Neo4j graph database on a HPC environment utilising parallel learning on several machines. We have provided a generic importer capable of handling different data sources. It makes use of a configurable ini-file which offers a predefined structure and is read-in by the generic importer. All software is available online[1].

III. Data integration and data schema

The actual data schema for the graph on which this work is based is presented in Figure 2. For this purpose, the data used was first considered, taking into account the underlying data structure. This data structure is formalised and published in the *Registry of DICOM Data Elements*[2]. There the different categories of objects within the *DICOM* metadata are listed, described and linked. The underlying tree structure of the information object definitions (IOD) and their sub-trees consisting of other IODs, modules and attributes is very well displayed in the *DICOM Standard Browser*[3]. Our data schema was significantly influenced by these sources and represents the inherent data structure using nodes, edges and attributes. The given selection and arrangement of the individual nodes has been made by the author as an exemplary instance. By adjustments in the configuration file also other schemes arise. However, for the graph used in this work it was necessary to decide on a schema. First, it was important to keep the four-level hierarchy of the DICOM data. This can be observed in Figure 2 in the middle strand. Each patient has his or her own node linked to his or her studies. These in turn contain the associated series, which then contain the images. In addition to this main strand within the schema, additional information is then annotated. All modules classified as mandatory (M) are included. In addition, at least one module from the classes conditional (C) and user optional (U) was also used. For (C) the class `Contrast/Bolus` is chosen, for (U) we decided on `Patient Study`. Within the data schema, the IOD modules used, which form their own node groups, are highlighted in yellow for visualisation, and the attributes in red. The blue line *Node Group = True* implies that the nodes listed below belong to the node group of the heading. These are represented as triangles in the graph, but are shown here as node groups for clarity. As an example, the `Manufacturer` node can be considered. It belongs to the node group `General Equipment` and forms a triangle in the graph with the node `General Equipment` and the node `General Series`. An example is shown in Figure 3.3. In contrast, the `General Series` node, for example, has attributes such as `Modality`, `Series Instance UID`, and others stored as node-owned attributes rather than as separate nodes.

However, such specifications can be freely designed and modified via the configuration file, as explained in the section before. In addition to the data contained in the *DICOM* files, two more nodes have been added. `Source` specifies the source of the data. For the given data, this is stored under the tag (0013, 1010). `File` stores the file name of the image and therefore serves as a kind of provenance, allowing the nodes to be uniquely assigned to a respective file. To ensure that said two nodes can be included, it is important for the configuration of the importer that the patient is contained in the graph as a node. This means a minor restriction in the sense of free configuration, however, such a graph without patients should be difficult to justify in terms of content.

Due to data protection rules, we will present results using a second data source, which is open-source and also supports the generic usability of the importer. The *SIMBA Image Management and Analysis System*[4] is used as our source. From the projects listed there, the *ELCAP Database* and from it again the *Zero-Change Dataset* were selected. The data comes from the *Public Lung Database to Assess Drug Response*, as can be seen from the website. A second configuration file named `dev2.ini` is created for them, which partly contains different nodes from the first one. Since the only purpose is to show conceptually that the script works for other data sets and configurations, only a much smaller total number of node types is used in the configuration file.

IV. Link Prediction

A. Scores based on the topology of the graph

Link prediction belongs to the field of computational analysis of a network, where the nodes represent persons or entities and the edges represent relations. These networks are dynamic and change over time. The link prediction problem deals with a section of such a network at a time t_0 and asks for the most accurate predictions possible for edges that do not yet exist at time t_0 and will be added at a later time t. Among other things, the network's own topology plays a crucial role. To be able to quantify this topology different neighbourhood measures from graph theory and their relative effectiveness are investigated.

In [13] a so-called score is used for the measure of this effectiveness. It is calculated in different ways. Examples are:

- Common Neighbours: Given a graph $G = (V, E)$, $Score(x, y) := |\Gamma(x) \cap \Gamma(y)|$ describes the number of common neighbours of two nodes $x, y \in V$. Here, $\Gamma(v)$ denotes the direct neighbourhood of a node $v \in V$. [13]

[1]See https://github.com/TbsHbnthl/master-s-thesis-link-prediction-on-large-scale-knowledge-graphs.

[2]See https://dicom.nema.org/medical/dicom/current/output/chtml/

[3]See https://dicom.innolitics.com/ciods

[4]See Simba database - public lung database. http://www.via.cornell.edu/visionx/simba/.

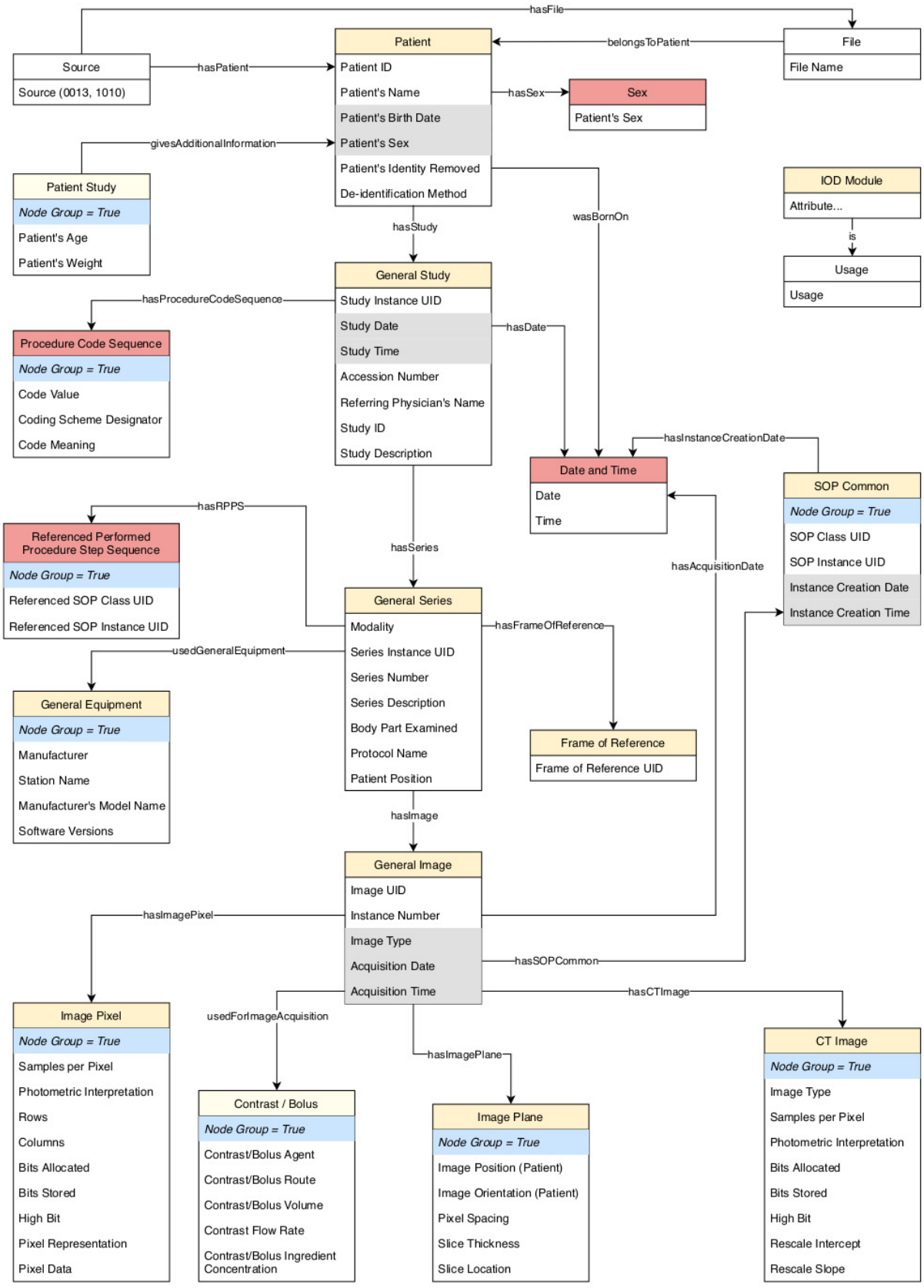

Fig. 2. Data schema for the import of *DICOM* files.

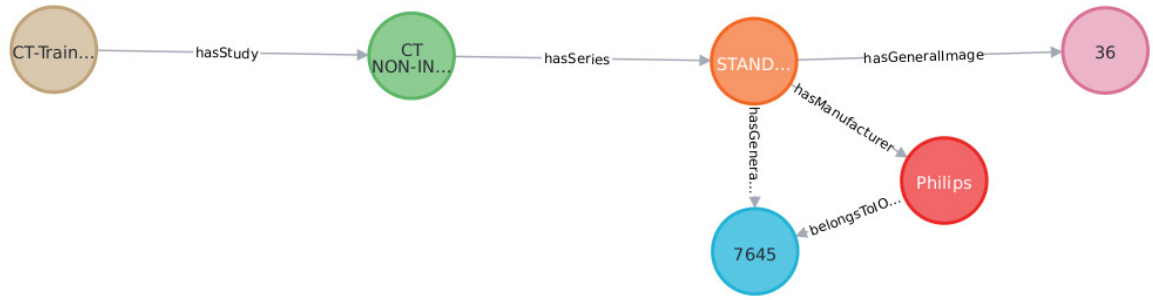

Fig. 3. Partial section of the graph: Exemplary triangle in the graph between the named example nodes `Manufacturer` (red), `General Equipment` (blue) and `General Series` (orange).

- Preferential Attachment: Given again a graph $G = (V, E)$. The underlying premise is the assumption that the probability that a new edge contains the node $x \in V$ is proportional to $|\Gamma(x)|$. Since the measure was originally conceived for predicting future collaborations between two authors, this yields $Score(x, y) := |\Gamma(x)| \cdot |\Gamma(y)|$. This builds on the idea that nodes with many edges have a higher probability of even more edges. [13]
- Adamic/Adar: The coefficient found here originally yields a measure that two homepages are strongly connected. For this purpose, features z are computed from a feature base set F of the two nodes, here web pages, and the commonality is defined as:

$$\sum_{z:\text{features shared by } x,y} \frac{1}{\log(frequency(z))}$$

This gives less weight to more frequent features than to less frequent ones. If features are to be left out and only the topology of the graph is to be considered, the following score is used for two nodes $x, y \in V$ of a graph $G = (V, E)$:

$$Score(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log(|\Gamma(z)|)}$$

These measures belong to methods based on node adjacency. [13]

They are presented in the Neo4j database in two ways as the basis of link prediction within the graph used there. First, there is the possibility of making the addition of a new edge conditional on whether the above score exceeds a pre-specified bound. If it does, the edge is added. On the other hand, the scores can be combined with supervised learning: They are used as features to train a binary classifier. This then predicts whether a particular pair of nodes will be connected by an edge with high probability in the future. To train and evaluate the classifier, the graph used is divided into training, testing and validation sets. Then training is performed within the training graph and the result is applied to the test graph. During validation, promising results are shown for the use case. With this work, as will be explained later, a different approach is taken, but one that also uses these scores as features or as a criterion for choosing a label.

B. Link prediction for paths based on node attributes

The approach adopted in this paper makes use of Conditional Random Fields. Therefore, their origin is briefly examined here and an introduction is given.

a) Markov chain: First, a simple Markov chain of order n is considered. The idea is to be able to calculate the probability of future states occurring. The order indicates on how many previous states the next one depends. In a first-order Markov process, the next state depends only on the current state. At the beginning, the system is in the initial state. [19]

Definition IV.1. *A Markov process is understood to be a tuple (S, A, δ). Here S describes the finite set of states, A the set of possible actions, and δ the state transition function. [19]*

For each pair (s_t, a_t) with $s_t \in S, a_t \in A$ the state s_t transitions via $\delta(s_t, a_t)$ to the state s_{t+1}. The transitions in this case are usually given in probabilities. The choice of action depends on the current state and can be represented as a function $\pi : S \to A$; $\pi(s_t) = a_t$. It is also called a strategy. [19]

b) Hidden Markov models: Hidden Markov models are used to represent probability distributions over sequences of observations. A distinction is made between the observation X_t and the state Z_t at time t. The latter is hidden, hence the name of the model. Here, as in the 1-step Markov chains, the so-called Markov property is assumed: Z_t at time t depends only on Z_{t-1} at time $t - 1$. An example of this can be seen in Figure 4. The time t need not be an explicit time and can also be implicitly considered as a location within the sequence. The overall probability distribution of a sequence of states and observations can be expressed as an equation as follows:

$$P(Z_{1:N}, X_{1:N}) = P(Z_1)P(X_1|Z_1)\prod_{t=2}^{N} P(Z_t|Z_{t-1})P(X_t|Z_t)$$

Since the states are hidden and only the observations are considered, which in turn depend on the states, the probability of an N-element sequence is represented by a product of

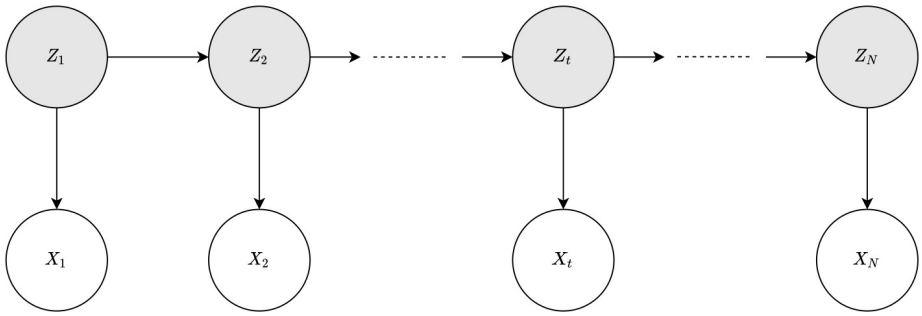

Fig. 4. Example of a hidden Markov model: Z_t describes the state and X_t the observation dependent on it at time t.

conditional probabilities. Moreover, except for the initial state, each state depends on the previous one. [20], [21], [22]

According to [20], [21], there are five elements that characterise a hidden Markov model:

- the number K of states that can be assumed in the model. The states are represented as $K \times 1$ vectors with binary values such that the k-th state at time t takes the value 1 in the k-th row and 0 everywhere else.
- the number Ω of distinct observations that can be observed in the model. Analogous to the states, an $\Omega \times 1$ vector is used.
- the state transition model A: This is also called the state transition probability distribution and describes the probability of changing from a state $Z_{t-1,i}$ to a state $Z_{t,j}$ within one time step. Here $i, j \in 1, ..., K$. This can be formulated as follows:
$$A_{i,j} = P(Z_{t,j} = 1 | Z_{t-1,i} = 1)$$
Each row of A sums up to 1 in this case.
- the observation model B is an $\Omega \times K$ matrix whose elements $B_{j,k}$ give the probability of making the observation $X_{t,k}$ given the state $Z_{t,j}$:
$$B_{j,k} = P(X_t = k | Z_t = j)$$
- the initial state distribution π is a $K \times 1$ vector with $\pi_i = P(Z_{1,i=1})$.

The model is often abbreviated in literature as $\lambda = (A, B, \pi)$. [20], [21]

c) Markov Random Fields: Let $G = (V, E)$ be an undirected graph. The nodes $v \in V$ correspond to the random variables which can assume the states. Here, these depend only on the states of the random variables u of their Markov cover $B_v := \{u : (v, u) \in E\}$. This is expressed in the following equation:

$$P(x_1, ..., x_n) = \frac{1}{Z} \prod_{c \in C} F_c(x_c)$$

Here C is the set of maximal cliques of the graph. The functions F are non-negative and depend on the variables within a clique c. For normalisation, a function $Z = \sum_{x_1,...,x_n} \prod_{c \in C} F_c(x_c)$ is used so that the distribution sums up to 1 overall. [23]

d) Conditional Random Fields: Conditional Random Fields are a special case of Markov Random Fields and belong to the field of supervised learning. Instead of only considering the probability for a label sequence y, here the probability of a label sequence y, conditioned by an observation sequence x, is determined:

$$P(y|x) = \frac{1}{Z(x)} \prod_{c \in C} F_c(x_c, y_c),$$

$$Z(x) = \sum_{y \in Y} \prod c \in C F_c(x_c, y_c)$$

The normalisation function $Z(x)$ now also depends on x.

In other literature, the definition of a (linear chain) Conditional Random Field is the conditional probability

$$p(y_{1:n}|x_{1,n}) = \frac{1}{Z} \exp\left(\sum_{n=1}^{N} \sum_{i=1}^{F} \lambda_i f_i \left(y_{n-1}, z_n, x_{1:N}, n\right) \right).$$

Within the exponential function, the first sum is over $n = 1, ..., N$, which indicates the position of a word, or here a node, within the sequence. The second sum iterates the features f_i weighted by the scalars λ_i, $i = 1, ..., F$. The values for the weights must be given or learned by the CRF model. They ensure that certain labels are preferred or even avoided. [24] For a given sequence, several features can be active at the same time, i.e., not equal to 0. This is called overlapping features. This can happen because, unlike in hidden Markov models, it is also possible to look at subsequent or previous elements of the sequence. [24] To train, fully labelled training sequences $(x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)})$ are required, where $x^{(i)} = x^{(i)}_{1:N_i}$ $\forall i \in 1, ..., m$. Thus, the conditional probability of the training data is maximised:

$$\sum_{j=1}^{m} \log p\left(y^{(j)} | x^{(j)}\right)$$

This is computed by default employing algorithms that use the gradient descent method. [24]

To assess the quality of the prediction, the F1-score (also balanced F-score or F-measure) is used. This can be regarded

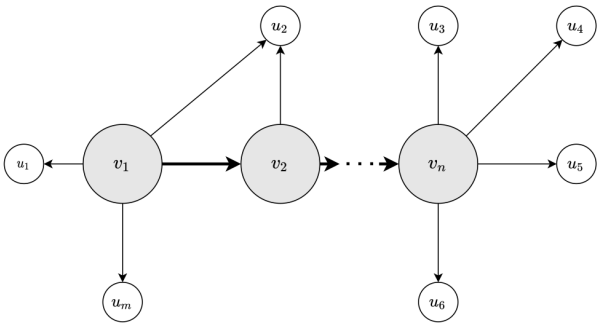

Fig. 5. The input path p consists of the nodes v_i, $i = 1, ..., n$ highlighted in grey. The white nodes u_j, $j = 1, ..., m$ serve as labels of the nodes of p.

as a weighted average of the precision and the recall. The best value is 1 and the worst value is 0. The formulas used for this are:

$$F1 = 2 \cdot \frac{\textit{Precision} \cdot \textit{Recall}}{\textit{Precision} + \textit{Recall}},$$

$$\textit{Precision} = \frac{TPR}{APR},$$

and:

$$\textit{Recall} = \frac{TPR}{APS}.$$

Here *TPR* means true positive results, *APR* means all positive results and *APS* are all samples that should have been identified as positive.

e) Learning Paths: Link prediction is used to predict possible, initially non-existent edges for the previously constructed graph. For this purpose, the graph is imported from *Neo4j* into *Python* via the *py2neo* library[5]. Then, using a query, paths are read in from the graph to be used as input for the conditional random fields and link prediction. The paths are converted to NER-compatible (named entity recognition) form. Thus, a path p is considered first. Then, for each node v contained in p, all neighbouring nodes $u \in \Gamma(v)$ are taken as possible labels. Thus, each node can be used both as a node of a path and as a label for other nodes, see Figure 5.

In the next section, we describe and evaluate different scenarios.

C. Creating one-node paths

The simplest form offers a path of length one, i.e. a single node and its direct neighbourhood. For this purpose, these one-node paths are read from the graph. It is specified which node type is considered, e.g. patients or images. Then a graph query is used to find the direct neighbourhood $\Gamma(v)$ of these nodes $v \in G$ and Γ is stored as a set of labels $l(v)$ for v. Since the CRF library can only assign one label to each node v at a time, criteria must be used for selection. For this purpose, section IV-A is used here to select nodes with, for example,

[5]see https://py2neo.org/2021.1/

TABLE I
EXCERPT FROM THE OUTPUT OF QUERY Q1. THE TERM SCORE REFERS TO THE VALUE CALCULATED FOR THE NODE AND ITS LABEL BY NEO4J'S COMMON NEIGHBOURS ALGORITHM.

patientNode	labelNode	score
CT-Training-BE001	SPIE-AAPM Lung CT Challenge	251.0
CT-Training-BE001	1.2.840.113704.1.111.2112.1167842143.1	2.0
CT-Training-BE001g	Patient_Study	2.0
CT-Training-BE001	073Y	1.0
CT-Training-BE001	1-001.dcm	1.0
...		

the highest score in one of the link prediction algorithms available in *Neo4j*. Later, alphabetical sorting is also given as an alternative. The choice of the method for probing the labels on the one hand influences the result and on the other hand also the runtime of the queries. First, single patient nodes are considered. As their label the neighbour with the highest score first at Common Neighbours and then at Total Neighbours is chosen. Afterwards we consider two other nodes, namely `General Image` and `Date`. The queries used for this are the following (Since some queries did not terminate they are left out):

(Q1) `MATCH (p:Patient)-[]-(a) RETURN p.nodeUID as patientNode, a.nodeUID as labelNode, gds.alpha.linkprediction.commonNeighbors(p,a) AS score ORDER BY p.nodeUID,score DESC,a.nodeUID`

(Q2) `MATCH (p:Patient)-[]-(a) RETURN p.nodeUID as patientNode, a.nodeUID as labelNode, gds.alpha.linkprediction.totalNeighbors(p,a) AS score ORDER BY p.nodeUID,score,a.nodeUID`

(Q4) `MATCH (p:General_Image)-[]-(a) RETURN p.nodeUID as imageNode, a.nodeUID as labelNode ORDER BY p.nodeUID,a.nodeUID`

(Q5) `MATCH (p:Date)-[]-(a) RETURN p.nodeUID as dateNode, a.nodeUID as labelNode ORDER BY p.nodeUID,a.nodeUID`

See table I for an example output for query Q1. The algorithm we use for applying the *CRFs* to the paths from the graph consists of the following steps:

Algorithm 1 INTEROPERABLE-DATA

Require: Graph `G` in *Neo4j*
Ensure: Label prediction, Measurement of prediction success
1: readNodePathsFromGraph(`G`)
2: splitValidationAndTrainingData()
3: for all `P` in `AP`:
4: assignFeaturesToNodesInPaths(`N(P)`)
5: assignLabelsToNodes(`N(P)`)
6: trainUsingCRFs(`AP`)
7: evaluateResultByComparingToValidationData()
8: **return** predictionVector, F1-Score, Precision, Recall

Here `N(P)` denotes the set of nodes in path `P` while `AP` denotes the set of all paths read from the graph. The set of output values consists of a prediction vector as well as the F1 score, the precision and recall.

TABLE II
DETAILS FOR QUERY Q1

node	precision	recall	f1-score	support
SPIE-AAPM Lung CT Chall...	1.0000	1.0000	1.0000	14
accuracy			1.0000	14
macro avg	1.0000	1.0000	1.0000	14
weighted avg	1.0000	1.0000	1.0000	14

TABLE III
DETAILS FOR QUERY Q2

node	precision	recall	f1-score	support
1-414.dcm	1.0000	1.0000	1.0000	0.0
1.2.840.113704....	1.0000	1.0000	1.0000	0.0
...				...
060Y	1.0000	0.0000	0.0000	2.0
061Y	1.0000	0.0000	0.0000	1.0
063Y	1.0000	1.0000	1.0000	0.0
...				...
accuracy	0.0000	0.0000	0.0000	8.0
macro avg	0.9730	0.8108	0.7838	8.0
weighted avg	1.0000	0.0000	0.0000	8.0

TABLE IV
DETAILS FOR QUERY Q4

node	precision	recall	f1-score	support
-0.10	1.0000	1.0000	1.0000	0
...				...
-100.70	1.0000	1.0000	1.0000	0
-1000	1.0000	0.0000	0.0000	653
-1000.00	1.0000	1.0000	1.0000	0
...				...
-1024	0.8417	1.0000	0.9141	3786
accuracy	0.8417	0.8464	0.8441	4473
macro avg	0.9988	0.7664	0.7658	4473
weighted avg	0.8660	0.8464	0.7737	4473

V. EVALUATION

A. Runtime

This subsection deals with the consideration of the achieved runtimes of the link prediction programmes. First, the runtime result of the queries Q1 - Q5 is presented. Within the programme, times are measured for all individual sections. The problematic part of the programme is the `sent2features()` method. For illustration the runtime of the time-relevant parts is shown in Figure 6. All other parts of the programme have a negligible very small runtime. This is especially evident for Q4. In this query, the `General Image` node is in the centre, which compared to other nodes such as `Patient` has already got a lot of neighbours due to the structure of the graph. The runtime, which is almost completely generated by `sent2features()`, amounts to a total of slightly less than 11 hours.

B. Quality

The first attempts at link prediction are carried out with single-node paths. Here the focus is initially on the patient. For Q1 link prediction shows the association of the data with the associated study *SPIE-AAPM Lung CT Challenge*. In this case a F1-score of 1 is obtained. This prediction is very accurate, however this is not surprising given the data. The patient node has a very limited type and number of neighbours. Sorting by number of common neighbours leaves only the source. This is also reflected in the detailed look at the labels, as can be seen in Table II.

In these tables, available labels are shown under the heading node. Precision, recall and the F1-score are shown to the right. The value at support indicates the frequency of the find. The opposite results are obtained for sorting by Total Neighbours. Here a F1-score of 0 is obtained. Thus, the prediction has completely failed here. The result can be seen in Table III. Again, the actual result is not surprising considering the data. The patients have different ages and due to the small group of individuals, clustering is unlikely.

The penultimate one-node path query is Q4. Here labels for the node `General Image` are being examined. The label selection is based on alphabetical order. From a biological point of view, a different weighting may be more appropriate, but several methods of label selection should be tried for scientific reasons. For Q4, a nominally very good value of 0.7737 was obtained for the F1-score. The detailed consideration of the result is presented in excerpts in Table IV. It shows that different labels were selected for the images in the prediction, with priority given to the label -1024. For the last query Q5 there is only a limited set of available nodes and edges of the graph due to the node selection and the given data. The programme nominally returns a very high value with an F1-score of 0.9819. The label predicted for the node `Date` is `SOP_Common`. The values of precision and recall compared to the F1-score are shown for the queries Q1 - Q5 in Figure 7 and Figure 8. The former relates precision and recall to each other. The contour lines provide a visual impression of the corresponding F1-score. The latter shows the three values for precision, recall and F1-score side by side,

VI. CONCLUSION AND OUTLOOK

Our studies pursued several goals. The first and second were to create a feasible data representation schema capable of handling clinical imaging data in a knowledge graph and the generic approach for importing imaging data into a graph. *Neo4j* provides an easy way to import large amounts of data with bulk import and we provide the source code of our solution online. This can be individually configured by the user with the help of the script presented here and the associated configuration file. The design of the graph can be very much defined by the user. For the combination with already existing graphs and data systems an interface can be formed with few lines of code. To do so, only the possibly overlapping node types have to be identified. The corresponding CSV files of the programme presented here can be read in a subsequent programme and the node IDs can be stored in sets. Thus, our solution could also be integrated in analysis workflows, for example utilising text mining.

The third goal was to present a novel approach for link prediction utilising graph structures and applying NER and

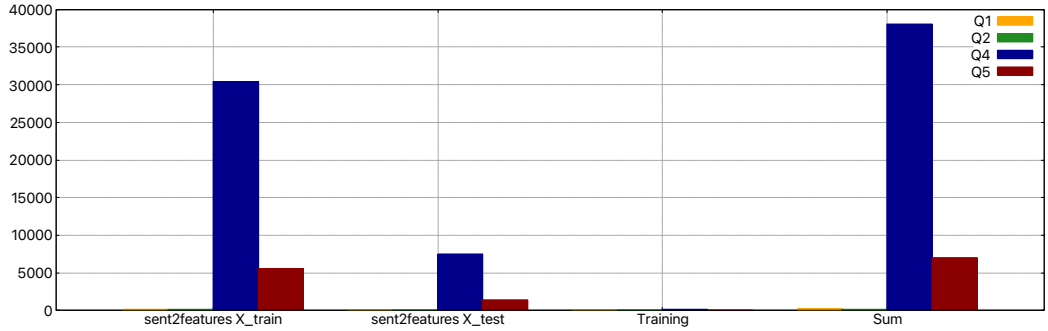

Fig. 6. Average runtime (in seconds) of the relevant parts of the queries Q1 - Q5.

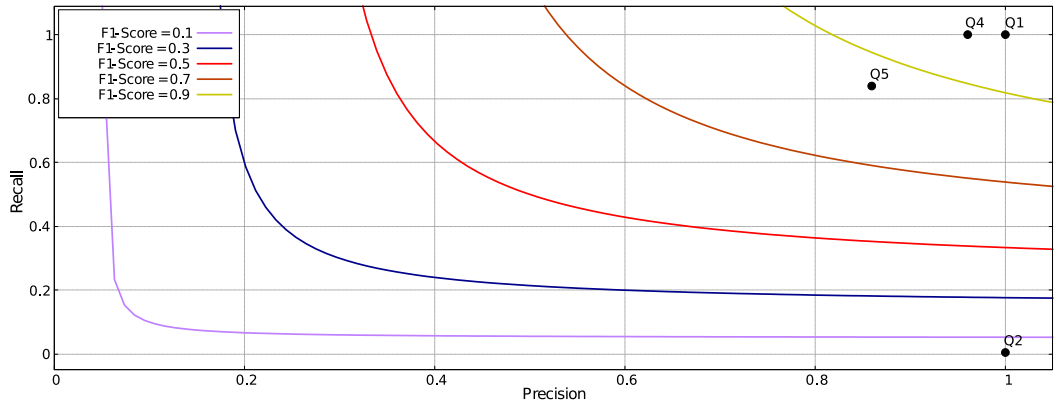

Fig. 7. Precision recall diagram for queries Q1 - Q5.

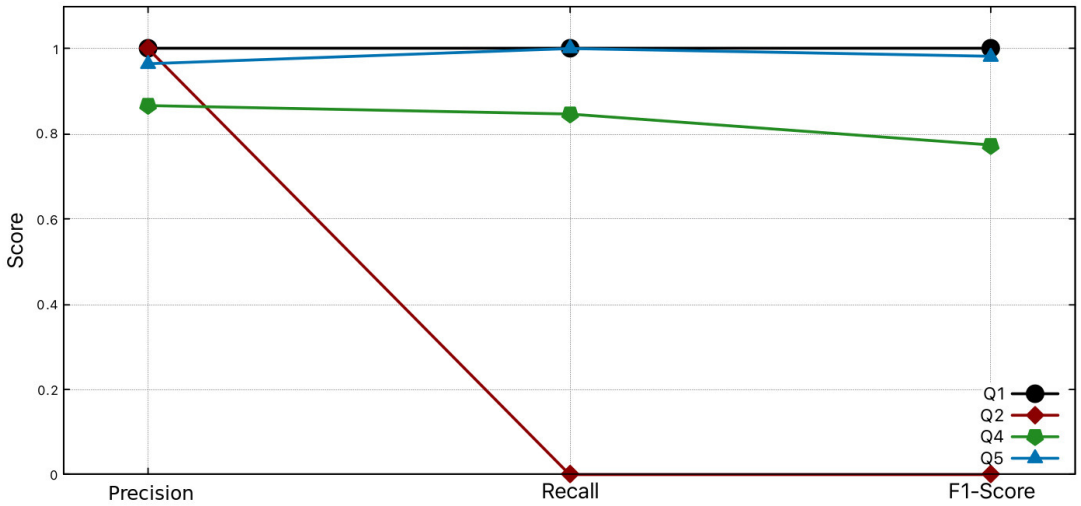

Fig. 8. Comparison of precision, recall and F1-score of queries Q1 - Q5.

CRFs to paths from a graph. For single-node paths, excellent results were obtained for the selected nodes. But we could also show the importance of data management and further research on link prediction using graph structures. For Q1 we could provide trivial results and this clearly underlines the need for data literacy, understanding the structures is essential. Our proposed approach also states the importance of an evaluation with state-of-the art graph embedding technologies to prove the advantage of keeping graph structures for AI approaches on graphs as [16] proposed.

The next step would be considering multi-node paths which will show an increasing runtime for large data sets. Querying features from the graph in our experimental setting turned out to be very time consuming and scales accordingly with the amount of data. The second problem is the increasing runtime for machine learning as the number of nodes used in the input path grows. At the same time, the requirements for the available main memory also increase enormously. However, both are related not only to the length of the input path, but also to the local environment of the paths. We assume that sparsely populated locations of the graph allow better predictions and provide faster results.

While our proof of concept is both functional and generic, extending the knowledge graph, e.g. with data from text mining on scientific documents, is feasible and just a matter of modelling connectors to the relevant sources since the software is prepared for running in a workflow.

ACKNOWLEDGMENT

We thank the Regional Computing Center of the University of Cologne (RRZK) for providing computing time on the DFG-funded (Funding number: INST 216/512/1FUGG) High Performance Computing (HPC) system CHEOPS as well as support.

REFERENCES

[1] O. Dössel and T. M. Buzug, *Medizinische Bildgebung*. Walter de Gruyter GmbH & Co KG, 2014.

[2] D. Peck, "Digital imaging and communications in medicine (dicom): a practical introduction and survival guide," 2009.

[3] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[4] C. S. Burns, R. M. Shapiro, T. Nix, J. T. Huber *et al.*, "Examining medline search query reproducibility and resulting variation in search results," *iConference 2019 Proceedings*, 2019.

[5] A. Callahan, V. Polony, J. D. Posada, J. M. Banda, S. Gombar, and N. H. Shah, "Ace: the advanced cohort engine for searching longitudinal patient records," *Journal of the American Medical Informatics Association*, vol. 28, no. 7, pp. 1468–1479, 2021.

[6] X. Xu, X. Xu, Y. Sun, X. Liu, X. Li, G. Xie, and F. Wang, "Predictive modeling of clinical events with mutual enhancement between longitudinal patient records and medical knowledge graph," in *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2021, pp. 777–786.

[7] Hulpus, Ioana and Hayes, Conor and Karnstedt, Marcel and Greene, Derek, "Unsupervised Graph-Based Topic Labelling Using Dbpedia," in *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, 2013, pp. 465–474.

[8] J. Dörpinghaus and A. Stefan, "Knowledge extraction and applications utilizing context data in knowledge graphs," in *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2019, pp. 265–272.

[9] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig *et al.*, "Gene ontology: tool for the unification of biology," *Nature genetics*, vol. 25, no. 1, p. 25, 2000.

[10] D. S. Wishart, Y. D. Feunang, A. C. Guo, E. J. Lo, A. Marcu, J. R. Grant, T. Sajed, D. Johnson, C. Li, Z. Sayeeda *et al.*, "Drugbank 5.0: a major update to the drugbank database for 2018," *Nucleic acids research*, vol. 46, no. D1, pp. D1074–D1082, 2017.

[11] K. Khan, E. Benfenati, and K. Roy, "Consensus qsar modeling of toxicity of pharmaceuticals to different aquatic organisms: Ranking and prioritization of the drugbank database compounds," *Ecotoxicology and environmental safety*, vol. 168, pp. 287–297, 2019.

[12] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[13] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[14] M. Xu, "Understanding graph embedding methods and their applications," *SIAM Review*, vol. 63, no. 4, pp. 825–853, 2021.

[15] M. Simonovsky and N. Komodakis, "Graphvae: Towards generation of small graphs using variational autoencoders," in *International conference on artificial neural networks*. Springer, 2018, pp. 412–422.

[16] W. Cukierski, B. Hamner, and B. Yang, "Graph-based features for supervised link prediction," in *The 2011 International Joint Conference on Neural Networks*. IEEE, 2011, pp. 1237–1244.

[17] J. Dörpinghaus, A. Stefan, B. Schultz, and M. Jacobs. (2020) Towards context in large scale biomedical knowledge graphs. [Online]. Available: http://arxiv.org/abs/2001.08392

[18] J. Dörpinghaus, V. Weil, S. Schaaf, and T. Hübenthal, "An efficient approach towards the generation and analysis of interoperable clinical data in a knowledge graph," in *2021 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 2021, pp. 59–68.

[19] J. Frochte, *Maschinelles Lernen: Grundlagen und Algorithmen in Python*. Carl Hanser Verlag GmbH Co KG, 2019.

[20] Z. Ghahramani, "An introduction to hidden markov models and bayesian networks," in *Hidden Markov models: applications in computer vision*. World Scientific, 2001, pp. 9–41.

[21] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[22] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[23] A. Blake, P. Kohli, and C. Rother, *Markov random fields for vision and image processing*. MIT press, 2011.

[24] X. Zhu, "Cs838-1 advanced nlp: Conditional random fields," *Technical report, The University of Wisconsin Madison*, 2007.