24th European Conference on Artificial Intelligence - ECAI 2020
Santiago de Compostela, Spain

Incorporating Semantic Dependencies Extracted from Knowledge Graphs into Joint Inference Template-based Information Extraction

Hendrik ter Horst and **Philipp Cimiano**[1]

Abstract. In template-based information extraction, a template is described by a predefined set of slots that need to be filled with entities mentioned in a text. Compared to traditional relation extraction that aims at classifying binary relations involving a pair of entities only, template-based slot-filling is of higher complexity as interdependencies between slot-filler values need to be taken into account. To model these dependencies, we tackle the slot-filling task as a joint inference problem and build on factorized distributions as used in conditional random fields. Dependencies are often described by textual features only, but in some cases they are of semantic nature and not directly expressed in text. To exploit this potential, we propose the incorporation of semantic dependencies extracted from knowledge graphs into an information extraction model. Dependencies are extracted from the variable bindings of queries executed over a knowledge graph and capture semantic soft constraints that are weighted as part of model training. We evaluate our approach on five distantly supervised labeled datasets extracted from Wikipedia/DBpedia and compare our approach to a most frequent entity baseline as well as a purely textual model. We show that there is an overall positive impact of integrating factual background knowledge in all datasets, yielding an improvement of up to 10 points in F_1.

1 Introduction

Information extraction (IE) is a research field of natural language processing which aims at extracting relevant information, e.g. key facts, from a given text written in natural language. While most work in the area of information extraction has been concerned with extracting binary relations between two entities, we are concerned with the problem of filling more complex structures comprising multiple relations. In template-based IE, a template describes a set of slots (or properties) that need to be filled with entities mentioned in a text. Consider the following example:

Jack "Jacky" Miller is an English soccer player born 15 December 1977 in Scunthorpe, United Kingdom, now residing in Berlin. He began his soccer career as a youth player for Scunthorpe United.

Usually, each slot of the given template describes a certain aspect of the entity type in question, in this case key facts of a soccer player. An example instantiation of such a template which is filled by information from the given example text is shown in Table 1. We

Table 1. Example IE-template with four slots (the resource type and three properties) describing a soccer player.

slot	type	slot filler
resource	SoccerPlayer	Jack_Miller
birthYear	string	"1977"
birthPlace	PopulatedPlace	Scunthorpe United_Kingdom
team	SoccerClub	Scunthorpe_United

consider two types of slots: i) slots that require free text annotations as slot-fillers (e.g. *birthYear*), and ii) slots that require entities of a specific type as fillers (e.g. *birthPlace*). In the context of domain specific templates, we assume that the assignments of values to slots have mutual dependencies. This motivates us to frame the task of template-based information extraction as a joint inference problem.

In order to model the task as a joint inference problem while remaining tractable, a promising approach is to rely on factorized distributions as used in conditional random fields (CRFs) [14]. CRFs model the conditional probability distribution of a set of output variables, slot variables in our case, by decomposing the joint probability into local factors ranging over a subset of these variables. Factors typically have a log-linear form over weighted sums of indicator functions as features. Such features are often linguistic e.g. describing the textual dependencies between different slot-filler candidates [9, 20].

Besides linguistic features, there are also extra-linguistic or semantic features that can be used to describe dependencies between slot-filler values. For instance, consider the slot-filler values from the previous example in the *birthPlace* and *team* slots. There is a semantic dependency between the birth place of the player, Scunthorpe, and its club, Scunthrope United, in the sense that there is some likelihood that players will play for teams from the place they are born in. While such semantic relations between entities are often not explicitly expressed textually (expecting certain knowledge of the reader), they are usually contained in large knowledge graphs such as DBpedia [1]. Therefore, we propose an approach in which semantic dependencies are extracted from knowledge graphs and incorporated into an extraction model via indicator functions. A clear benefit of conditional random fields is that such indicator functions can be incorporated straightforwardly into the model. In our proposed approach, semantic dependencies are extracted from a knowledge graph via graph queries where the bindings of these queries are turned into indicator functions. As knowledge graphs consist of triples (h, r, t), queries over a knowledge graph can be expressed via triple patterns involv-

[1] CITEC, Bielefeld University, Germany, email: {hterhors,cimiano}@techfak.uni-bielefeld.de

ing a number of query variables that need to be bound during the query evaluation. For instance, the following query:

$\{$*Scunthorpe_United ?r Scuntrhope*$\}$ asks for relations $?r$ that exist between the entity *Scunthorpe_United* and the entity *Scunthorpe*, e.g. binding the variable $?r$ to the value *hometown* as a result of query evaluation. The value *hometown* can then be used to describe the semantic relation between both entities via an indicator function computing binary features. A trained model will assign a learned weight to such an indicator function describing the importance of individual semantic relations between slots. In this way, a model can learn to prefer slot assignments to *birthPlace* and *team* for which certain semantic relations hold.

In this work, we investigate the impact of incorporating such semantic dependencies extracted from knowledge graphs into template-based slot-filling models. We evaluate the model on five datasets that have been automatically extracted by aligning Wikipedia articles and their anchors with DBpedia info boxes.

We assess the impact of incorporating semantic features by comparing our approach to two baseline models: i) an approach that chooses slot-filler values based on their frequency appearing in the document, and ii) a baseline model relying on linguistic features, only. Further, we evaluate the impact of the proposed queries in a query-ablation study. The datasets and evaluation scripts are available under `http://psink.techfak.uni-bielefeld.de/ecai-2019/dbpedia-ecai2019.zip`. The source-code can be found under `https://github.com/hterhors/`.

2 Related Work

Relation Extraction and Slot-Filling: The task of template-based information extraction dates back to the early days of information extraction work in the context of the Message Understanding Conferences (MUC) [8]. Since then, information extraction tasks have received wide attention in the NLP community, leading to many shared tasks such as the Automatic Content Extraction (ACE) Program [7] which was introduced in 2004 as a successor of MUC. Since 2009, several shared tasks on a wide range of relation extraction including slot-filling and knowledge base population have been defined as part of the Text Analysis Conference (TAC) [18]. In spite of the fact that most of these tasks are formulated as template-based extraction tasks, most of the developed systems reduce slot-filling problems to binary relation classification, relying on i) linguistic features [9, 38, 35], ii) matrix factorization techniques [27], iii) kernel-based approaches [36, 5, 3, 21] , and, recently iv) neural networks e.g., LSTMs for collective relation extraction per sentence [30], slot-filling and cold-start knowledge base population [28], but also for extracting relations between entities across multiple sentences [22, 24].

Yet, there are a number of works that have relied on joint inference approaches to extract several variables or types of variables in one joint model, such as the early work of Bunescu et al. on collective information extraction [2]. Our proposed approach relies on conditional random fields to model the interaction between several output variables, as proposed by Lafferty et al. [14]. Other frameworks supporting joint inference include for instance Markov logics [26] and [23].

Distant Supervision for Datasets: With the advent of large collaboratively created knowledge bases such as DBpedia [1], there has been an attempt to leverage such large knowledge bases to automatically label training data by projecting relations from these resources into text documents. This so called *distant supervision paradigm* was proposed in 2009 by Mintz et al. [20]. Along these lines, Hoffmann et al. describe the alignment of Wikipedia articles and DBpedia info boxes to create distantly supervised training data for binary relation extraction [10]. Reschke et al. proposed an alignment of DBpedia infoboxes with newspapers for generating a slot-filling data set [25].

Knowledge Graphs in Relation Extraction: Knowledge graphs have been exploited in the context of relation extraction, mostly focusing on knowledge base completion or population [37, 15], and query expansion [6]. Most recent works rely on embedding methods to encode the information in the graph into entity embeddings that encode the topology of the graph. Weston et al. [33] for instance learn to encode the interactions among entities and relations from the knowledge base in a joint fashion by embedding knowledge graphs and text into the same vector space. Context dependent knowledge graph embeddings were proposed by Luo et al. [16]. They transform knowledge graphs into sequences of words, relying on common training methods such as CBOW and Skip Gram [19]. While context is integrated by ignoring directions of relations, they use this method for triple classification for knowledge base completion. Inspired by the work of Weston et al. (2013), Wang et al. propose to embed text and knowledge graphs into separate vector spaces first and then combine them in a third model [32]. Another way of exploiting a knowledge base was proposed by Wang et al. [31]. They projected both known and new relations into the same relation-topic space in order to generate more training instances.

To the best of our knowledge, there is no work directly encoding knowledge graph relations as features into a probabilistic graphical model as we propose. However, such models are especially suited for incorporating such explicit knowledge as it can be directly integrated into a model via indicator functions. We show in this work how such indicator features can be instantiated directly from the bindings of queries evaluated over the knowledge graph.

3 Methods

3.1 Conditional Random Fields

In this work, we frame the problem of extracting IE-templates consisting of a number of slot variables as a structured prediction problem [29] and rely on conditional random fields (CRFs) [14] to model the (conditional) joint distribution of slot variables given the input text. Probabilistic graphical models, especially CRFs, are widely applied machine learning models when dealing with computationally expensive inference procedures.

Let $\vec{y}$ denote the target structure of interest (e.g. a soccer player IE-template) in that each dimension corresponds to one slot that needs to be filled (e.g. the birth year). Further, let $\vec{x}$ denote a tokenized input document d containing free text, where $\vec{x}_t$ corresponds to the t_{th}-token in d. The conditional probability of a specific assignment of $\vec{y}$ given the input document $\vec{x}$ can be modeled as:

$$p(\vec{y}|\vec{x};\theta), \tag{1}$$

where the conditional probability is parameterized by a (learned) parameter vector θ. The best slot-filling assignment for $\vec{y}$ is found through maximum a posteriori (MAP) inference:

$$\vec{y}\,' = \underset{\vec{y}}{\operatorname{argmax}}\, p(\vec{y}|\vec{x};\theta). \tag{2}$$

Conditional random fields decompose this joint probability into a factorized form consisting of local compatibility functions called factors defined over subsets of variables. The decomposition is typically

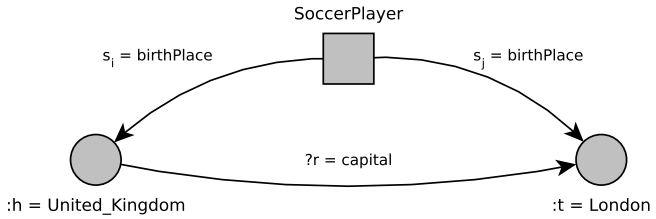

Figure 1. Example variable binding for query Q_1 being evaluated on the knowledge graph. The resulting triple $\langle{:}h = United\_Kingdom, {:}r = capital, {:}t = London\rangle$ is used by indicator functions $f^{\emptyset}$ (cf. Equation (6)) and $f^{\in}$ (cf. Equation (7)) to compute features.

described by a factor graph [13, 12]. A factor graph is a bipartite undirected graph $\mathcal{G} = (V, F)$ consisting of a set of random variables V and factors F that are connected. The set of random variables is defined as the union of input and observed output variables $V = \vec{y} \cup \vec{x}$ with $\vec{y} \cap \vec{x} = \emptyset$. A factor $\Psi \in F$ is a non-negative real-valued function defined over a subset of random variables within its scope $\lambda = \hat{\vec{y}} \cup \vec{x}$ with $\hat{\vec{y}} \subseteq \vec{y}$ and computes a scalar-score reflecting the compatibility of assignments to its output variables:

$$\Psi(\lambda) \rightarrow \mathbb{R}_{\geq 0} \text{ with } \Psi(\lambda) = \exp(\langle f(\lambda), \theta \rangle), \tag{3}$$

with $f(\cdot)$ representing a feature vector and θ a set of weight parameters shared between factors of the same type. In our case, we introduce factors defined for single variables as well as between pairs of variables, so that the overall probability distribution $p(\vec{y} \mid \vec{x})$ can explicitly be written as:

$$p(\vec{y}|\vec{x}) = \frac{1}{Z(\vec{x})} \prod_{y_i \in \vec{y}} \Psi'(y_i, \vec{x}) \prod_{y_j \in \vec{y} \setminus y_i} \Psi''(y_i, y_j, \vec{x}) \tag{4}$$

with $Z(\vec{x})$ denoting the partition function, $\Psi'(\cdot)$, $\Psi''(\cdot)$ denoting factors to single variables and pairs of variables, respectively.

Inference and Parameter Update: We approximate the posterior distribution by performing Markov chain Monte Carlo sampling. Proposal states are generated via Gibbs sampling [4], that is applying atomic changes to the IE-templates. An atomic change is defined as either deleting, changing or adding a slot-filler variable for a single slot. The unrolling of factors over the input is performed using imperatively defined factor graphs as proposed by McCallum et al. [17]. During training, parameter updates are performed with SampleRank [34], which is an online algorithm that learns preferences over hypotheses from gradients between atomic changes. As inference in factor graphs is NP-hard, we limit our search exploration to a fixed maximum number of sampling steps which in that sense limits the runtime to a constant factor. Exact runtime is highly affected by the computation of features, that is mainly querying the knowledge graph.

3.2 Features

All features are modeled as binary indicator functions. We distinguish between intra-textual features and semantic dependency features, where the latter capture the semantic dependencies extracted from a knowledge graph.

Intra-textual Features: Intra-textual features rely on the linguistic and textual structure of a document as commonly used by information extraction systems in the context of NLP.

- We capture the **local context around entities** that are used as slot-fillers in form of n-grams, ranging from one to three tokens around the entity in question.
- **Sentence distance features** measure whether the slot-filler entities of a template are co-located within the same sentence or rather distributed over the document. The feature measures the pairwise distance between all entities.
- **Entity surface forms** describe the surface form of the entity in question via n-gram features.
- **Common slot-filler entity type priors** capturing the prior probability of a given slot filler.
- The **linguistic context between slot-filler entities** is described via n-grams appearing in the text between two slot-fillers, ranging from one to three tokens.
- We capture the **IE-template coverage** by indicating whether a slot or slot pair is empty. For multi value slots, we also indicate the general number of slot-filler entities per individual slot.

Semantic Dependency Features: The main contribution of this work is the integration of background knowledge that captures semantic relations between entities that are not linguistically expressed. This class of features is based on results from evaluating a query on a knowledge graph containing factual background knowledge. A knowledge graph $\mathcal{G}$ stores information in form of triples $(h, r, t) \in \mathcal{I} \times \mathcal{R} \times \{\mathcal{I} \cup \mathcal{L}\}$, where $\mathcal{I}$ is the set of available resources, $\mathcal{R}$ is the set of relations, and $\mathcal{L}$ is the set of literal values. Thus, each triple $(h, r, t) \in \mathcal{G}$ is a composition of the head (or subject) $h \in \mathcal{I}$, the tail (or object) $t \in \mathcal{I} \cup \mathcal{L}$, and the relation (or predicate) $r \in \mathcal{R}$ between h and t. Knowledge is accessible via triple queries, consisting of triple patterns featuring constants denoted by *:x* and query variables *?x* that need to be bound as a result of the query evaluation. For example, the evaluation of $\{$*:Scunthorpe ?r :United_Kingdom*$\}$, that is eval($\{$*:Scunthorpe ?r :United_Kingdom*$\}$) is a set of relations $p \in P$ that *?r* can be bound to and that connect *Scunthorpe* with *United_Kingdom* in the knowledge graph.

In this work, queries were designed to capture meaningful relations between two entities in the knowledge graph while limiting search complexity and reducing feature sparsity. Thus, we limit the query design to consider a maximum of two possible relations connecting entities in the knowledge graph. While the 1-hop queries represent a direct connection between entities in the knowledge graph, 2-hop queries take an intermediate entity into account. We assume that two relations are sufficient to capture typical relations of two entities while arbitrarily complex queries (e.g. multi-hop or complex tree-structured relations) would rather lead to overfitting, exponentially increase the search space and induce feature sparsity. This leads to the design of the following four queries to access factual background knowledge from the knowledge graph:

1. Undirected-1-hop-relation:

$$Q_1(e_1, e_2) = \{{:}h\ ?r\ {:}t\} \cup \{{:}t\ ?r\ {:}h\} \tag{5}$$

Given a pair of slots $\langle s_i, s_j \rangle$, and corresponding slot-filler entities $\hat{e} \in s_i$ and $e' \in s_j$, the query is built by binding $\hat{e}$ to the head position *:h* and e' to the tail position *:t*. The query becomes undirected by the union of an expression with switched head/tail positions. When evaluating the query on the knowledge graph, relations between *:h* and *:r* are bound to the variable $?r$ resulting in a set of

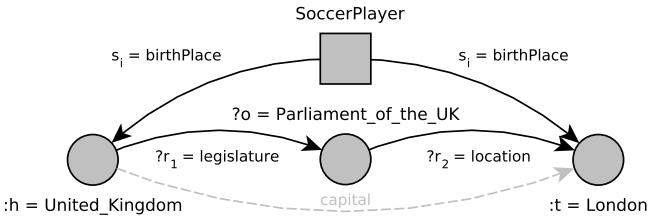

Figure 2. Example variable binding for query Q_2 assuming there is no direct relation between $United\_Kingdom$ and $London$ in the knowledge graph. The resulting tuple: $\langle{:}h = United\_Kingdom, {:}r_1 = legislature, {:}o = Parliament\_of\_the\_UK, {:}r_2 = location, {:}t = London\rangle$ is used by indicator functions $f^{\emptyset}$ (cf. Equation (6)) and $f^{\in}$ (cf. Equation (9)) to compute features.

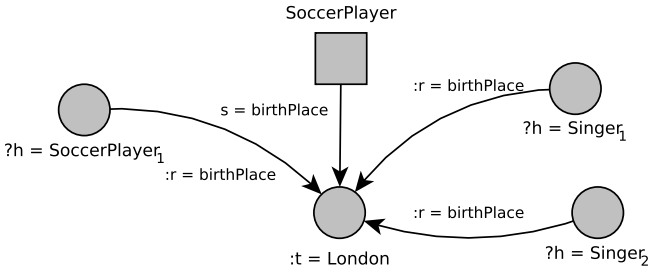

Figure 3. Example variable binding for query Q_3. The resulting set of head bindings $\{?h = SoccerPlayer_1, ?h = Singer_1, ?h = Singer_2\}$ is used by the indicator function $f^{\emptyset}$ (cf.Equation (11).

existing 1-hop-relations between $\hat{e}$ and e'. This query is built for entities coming from either the same slot (that is $i = j$, in case of multi value slots), or from different slots (that is $i \neq j$). Based on the result set of this query, we use the following two indicator functions to compute features:

$$f^{\emptyset}_{s_i,s_j}(\hat{e}, e') = \begin{cases} 1 & \text{iff } eval(Q(\hat{e}, e')) \neq \emptyset \\ 0 & \text{else} \end{cases} \tag{6}$$

and

$$f^{\textit{1-hop}}_{s_i,s_j}(\hat{e}, e', p) = \begin{cases} 1 & \text{iff } p \in eval(Q(\hat{e}, e')) \\ 0 & \text{else} \end{cases}. \tag{7}$$

Figure 1 shows an example of two entities coming from the same slot $s_i = s_j = birthPlace$. In the built query, *United_Kingdom* is bound to *:h* and *London* is bound to *:t*. The indicator function in Equation (6) returns true if the query result is not empty, that is any relation exists between $\hat{e}$ and e'. The indicator function in Equation (7) is instantiated for each relation p that is bound to $?r$. Our general intuition behind this query is that entities of a correct slot-filling show further relations in the knowledge graph, while wrongly assigned entities do not. For example, a person can be born in the United Kingdom and in London, but not in the United Kingdom and in Berlin. Thus, relations that connect *United_Kingdom* with *London* e.g. *capital* or *location* can be seen as indicators of a correct pair of slot filler entities.

2. Undirected-2-hop-relation:

$$Q_2(\hat{e}, e') = \exists o\{{:}h\ ?r_1\ o \wedge o\ ?r_2\ {:}t\} \cup \{{:}t\ ?r_1\ o \wedge o\ ?r_2\ {:}h\} \tag{8}$$

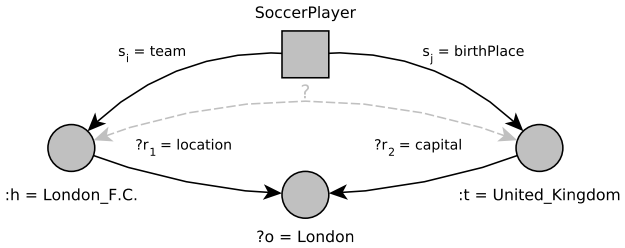

Figure 4. Example variable binding for query Q_4. The resulting tuple $\langle ?r_1 = London\_F.C., ?r_2 = United\_Kingdom\rangle$ is used by the indicator functions in Equation (6), (9), and (13).

The second query type is an extension of the first allowing an extra hop connecting two entities through an intermediate entity. Two entities $\hat{e} \in s_i$ and $e' \in s_j$ (cf. Q_1), are bound to the head $:h$ and tail $:t$ positions, respectively. Evaluating the query on the knowledge graph results in a threesome of bindings. The intermediate entity is bound to $?o$ while the two-hop-relations are bound to $?r_1$ and $?r_2$, respectively, thus connecting $:h$ with $:t$ through $?o$.
This query is designed to overcome the lack of completeness in knowledge graphs. For instance, under the assumption that there is no direct relation between *London* and *United_Kingdom* in the knowledge graph, two entities $\hat{e}$ and e' can be related via a two-hop-relation.
Figure 2 depicts such a two-hop-relation: *United_Kingdom* $\rightarrow$ *legislature* $\rightarrow$ *Parliament_of_the_United_Kingdom* $\rightarrow$ *location* $\rightarrow$ *London*. To compute features, we rely on the indicator functions from Equation (6) and the following Equation (9):

$$f^{\textit{2-hop}}_{s_i,s_j}(\hat{e}, e', p_1, p_2) = \begin{cases} 1 & \text{iff } \langle p_1, p_2\rangle \in eval(Q(\hat{e}, e')) \\ 0 & \text{else} \end{cases}. \tag{9}$$

3. Tail Plausibility:

$$Q_3(e, s) = \exists h\{h\ {:}r\ {:}t\} \tag{10}$$

The third type of query is parameterized with a single entity $e \in s$ and its slot s. In this query, we bind e to the tail position $:t$ and s to the relation $:r$. The evaluation of Q_3 results in a set of alternative head entities bound to $?h$ that have the same slot-entity (relation-tail) pair.
The intuition behind this feature is to measure the plausibility of the entity e in question as being in the tail position of a specific relation. The query returns alternative heads that have the same :tail; if the query returns a binding at all, this means that the tail e appears at least once more in the tail position of r in the knowledge base, thus being a plausible candidate.
An example is depicted in Figure 3, here the slot *birthPlace* is bound to $:r$ and the slot-filling entity *London* is bound to $:t$. The evaluation of this query our example knowledge graph results in the set of alternative head entities e.g. $SoccerPlayer_1$, $Singer_1$, and $Singer_2$. We compute features using the indicator function from Equation (11), simply checking whether other head-entities for that specific slot–slot-filler pair exist or not.

$$f^{\emptyset}_{s}(e, p) = \begin{cases} 1 & \text{iff } eval(Q(e, p)) \neq \emptyset \\ 0 & \text{else} \end{cases}. \tag{11}$$

4. V-relation:

$$Q_4(\hat{e}, e') = \exists o\{:h\ ?r_1\ o \wedge :t\ ?r_2\ o\} \quad (12)$$

The *V-relation* query is parameterized with two entities $\hat{e} \in s_i$ and $e' \in s_j$ (cf. Q_1). Different to Q_1 and Q_2, in Q_4 both entities $\hat{e}$ and e' are bound to head positions which are connected by two relations $?r_1$ and $?r_2$ through an intermediate variable entity $?o$ defining a v-structure. The evaluation of the query binds relations in the knowledge graph to $?r_1$ and $?r_2$, respectively, if both relations are connected to the same intermediate entity.
An example of a variable binding for this query is depicted in Figure 4. Here, the two entities coming from different slots $s_i = team$ and $s_j = birthPlace$ are $\hat{e} = London\_F.C. \in team$ and $e' = United\_Kingdom \in birthPlace$. The evaluation of this query on the knowledge graph binds the relation $location$ to $?r_1$ and $capital$ to $?r_2$, with an intermediate entity *London*.
Features for this type of query are computed by the indicator functions from Equation (6), (9) and the following indicator function (13):

$$f^{=}_{s_i,s_j}(\hat{e}, e', p_1, p_2) = \begin{cases} 1 & \text{iff } f^{2\text{-}hop}_{s_i,s_j} \wedge p_1 = p_2 \\ 0 & \text{else} \end{cases} \quad (13)$$

With these indicator functions, three types of features are computed: i) the first feature is true if the query result is not empty (cf. Equation (6)), ii) the second feature is true for a particular choice of values bound to $?r_1$ and $?r_2$ (cf. Equation (9)), and iii) the third feature is true when both relations returned are equal (cf. Equation (13)).

4 Experiments

We evaluate our approach to incorporating semantic dependencies extracted from knowledge graphs into template-based IE models on five domain specific datasets whose generation is described in the following.

Table 2. Performance of an approach that randomly fills slots based on the set of annotations that is generated by the symbolic entity annotation method. The last row (**max**) shows the maximum recall that can be reached by a system that relies on the set of annotations. *Architecture for ARCHITECTURALSTRUCTURE

	Random Slot Filling			
Dataset	**prec.**	**rec.**	**F_1**	**max**
SOCCERPLAYER	0.19	0.14	0.16	0.93
FILM	0.02	0.02	0.02	0.99
SINGLE	0.02	0.02	0.02	0.98
ARCHITECTURE*	0.09	0.10	0.09	0.98
DAM	0.13	0.12	0.13	0.99

4.1 Datasets and IE-template Definition

In the following, we briefly describe the extraction procedure for the datasets that we use throughout our experiments. We have used Wikipedia together with the DBpedia knowledge graph as they are strongly aligned with DBpedia being populated from the info boxes from Wikipedia. The info boxes themselves have not been used during training in our experiments but only to create the ground truth. Showing the impact of using knowledge graph derived features in information extraction requires a minimum alignment between the templates the IE system is supposed to extract and the entities mentioned in the text on the one hand and the relations and entities defined in the knowledge graph on the other hand. If the coverage of the knowledge graph and the text collection is low, the semantic dependencies extracted from the knowledge graph will hardly have any impact and in the extreme case no impact at all, with the pure intra-textual model as fallback.[2] To be able to show the impact and relevance of a well aligned knowledge base, we rely on the distant supervision paradigm to generate a number of datasets by aligning Wikipedia articles with DBpedia infoboxes following the methodology proposed by Reschke et al. [25]. The alignment searches for entities (anchors) and text mentions (literal values) in the article that are also annotated in the corresponding info box. Each dataset contains only resources from the same entity type e.g. *SoccerPlayer* and a set of annotatable slots based on an existing alignment. The coverage of an entity type with respect to a given set of properties denotes the number of resources for which all the properties could be successfully aligned. To find the datasets with the largest coverage, we performed an exhaustive search over all entity types that are listed in the DBpedia ontology.

Following this strategy, we selected five entity types that provide a high coverage of slots being filled throughout all resources[3]: i) *dbr:SoccerPlayer* (2338, 1176), ii) *dbr:Film* (1000, 6596), iii) *dbr:Single* (1000, 2128), iv) *dbr:ArchitecturalStructure* (527, 2701), and v) *dbr:Dam* (491, 2433).[4] Each training data point consists of exactly one distantly annotated IE-template and a corresponding full text Wikipedia article (without anchors). Detailed statistics for each dataset are given in Table 4, including the relevant slots, their types, and the number of potential slot-filler candidates for each slot.

Example SOCCERPLAYER Dataset: For the purpose of a running example, we present a detailed description of the SOCCERPLAYER-dataset. The slot-filling task is defined over six slots that need to be predicted which are either multi-value (mv) or single-value (sv) slots, and of either entity type (et) or literal type (lt): *birthPlace* (mv, et), *birthYear* (sv, lt), *deathYear* (sv, lt), *position* (sv, et), *team* (mv, et), and *resource* (sv, et). Slot-filler candidates for individual slots i.e. instances of different types such as *PopulatedPlace* for the *birthPlace* slot are specified in a corresponding ontology which has been automatically extracted from the DBpedia ontology. Slot filler candidates are extracted from the ranges of corresponding object properties, yielding 4905 soccer players[5], 4359 soccer clubs, 2314 places, and 14 playing positions.

4.2 Experimental Settings

We rely on a number of heuristics that are based on lexical matches in order to project entity annotations into the documents. Literal values are extracted via a set of regular expressions for each type of value. Further, all triples from the knowledge graph that are used to create the distantly supervised data are removed from the knowledge graph to prevent our system from using this knowledge.

[2] We have not used other datasets such as MUC or ACE as they are not strongly aligned with a particular knowledge graph.
[3] The first number in brackets indicates the count of resources (documents) in the dataset and the second number shows the average character based document length.
[4] *dbr:* `http://dbpedia.org/resource/`
[5] Note that there are 4905 soccer player resources but only 2338 have a high coverage of slots being filled by entities mentioned in the text.

Table 3. Comparison of the evaluation results between the baseline model, the intra-textual model and the intra-textual + semantic dependencies model. Results are averaged over three runs with a random 80/20 split. ARCHITECTURE short for ARCHITECTURALSTRUCTURE.

	Baseline			Intra-textual Model			Intra-textual + SD Model		
Dataset	**prec.**	**rec.**	**F_1**	**prec.**	**rec.**	**F_1**	**prec.**	**rec.**	**F_1**
SOCCERPLAYER	0.35	0.31	0.33	0.82	0.77	0.79	**0.91**	**0.88**	**0.89**
FILM	0.09	0.09	0.09	*0.63*	0.65	*0.64*	*0.63*	**0.66**	*0.64*
SINGLE	0.11	0.12	0.11	0.62	0.64	0.61	**0.70**	**0.79**	**0.74**
ARCHITECTURE*	0.25	0.28	0.26	0.76	0.68	0.71	**0.77**	**0.69**	**0.73**
DAM	0.39	0.42	0.40	*0.80*	0.66	0.72	0.80	**0.68**	**0.73**
Average	0.24	0.24	0.24	0.72	0.67	0.70	**0.76**	**0.74**	**0.75**

Table 4. Overview of the datasets. The table shows for each dataset the list of slots, their cardinality type, their filler type, and the number of possible values for each slot. Note that slots of type literal do not have a pre defined number of fillers as their values are theoretically arbitrary.

Dataset	Cardinality	Type	Values
SOCCERPLAYER			
resource	single	entity	4,905
birthPlace	multi	entity	2,314
birthYear	single	literal	n/a
deathYear	single	literal	n/a
position	single	entity	14
team	multi	entity	4,359
FILM			
resource	single	entity	5,405
producer	multi	entity	3,347
director	multi	entity	5,452
writer	multi	entity	5,452
starring	multi	entity	9,452
musicComposer	multi	entity	731
distributor	multi	entity	1,936
SINGLE			
resource	single	entity	17,881
writer	multi	entity	7,001
musicalBand	multi	entity	4,791
musicalArtist	multi	entity	4,791
album	multi	entity	10,090
producer	multi	entity	4,185
genre	multi	entity	548
ARCHITECTURALSTRUCTURE			
resource	single	entity	530
architect	multi	entity	356
architecturalStyle	multi	entity	70
location	multi	entity	408
openingYear	single	literal	n/a
yearOfConstruction	multi	literal	n/a
DAM			
resource	single	entity	502
river	multi	entity	370
country	multi	entity	74
location	multi	entity	601
status	single	literal	n/a
openingYear	multi	literal	n/a
buildingStartYear	multi	literal	n/a

Evaluation Protocol: In our experiments, the datasets were randomly split into 80% for training and 20% for testing. We ran each experiment three times, reporting the averaged harmonic macro F_1, defined as $F_1 = \frac{2tp}{2tp+fp+fn}$. A true positive (tp) is defined as a slot-value pair that was correctly predicted by our system, a false positive (fp) is defined as a slot-value pair that was predicted by our system but does not exist in the gold template, a false negative (fn) is defined as a slot-value pair that is in the gold template but was not predicted by our system. Note that the goal is always to find the correct entity type(s) (not a specific annotation) for each slot if they exist, otherwise assign *Null*. Further, correctly predicting a *Null*-value does not count as true positive, while wrongly predicting a *Null*-value indeed counts as false negative.

4.3 Experimental Results

In the main experiments, we compare two models i.e. the intra-textual model (IT) and the intra-textual + semantic dependencies model (IT+SD) against each other as well as against a baseline approach that selects slot-fillers based on the highest frequency of entity occurrence in the document. The IT model relies solely on linguistic and textual features, while the IT+SD model additionally incorporates the features based on the results of knowledge graph queries as described in Section 3.2. Results for these models are presented in Table 3.

Baseline Performance: The results of the baseline approach show moderate performances of 0.33 F_1, 0.26 F_1, and 0.40 F_1 for the dataset SOCCERPLAYER, ARCHITECTURALSTRUCTURE, and DAM, respectively. The performances for the datasets FILM, and SINGLE are comparatively low with 0.09 F_1, 0.11 F_1, respectively. The low results are likely due to the fact that full text articles describing films and singles generally have a higher variety of slot-filling candidates, which is apparent from the statistics depicted in Table 2.

Intra-textual Model Performance: When comparing the baseline with the intra-textual model, we notice a huge increase in precision, recall and F_1. The increase ranges from 47 point to 51 points in precision for the datasets SOCCERPLAYER and SINGLE, respectively. Considering the recall, the increase ranges from 22 points to 52 points for the datasets DAM and SINGLE. The general observation here is that datasets that already have a moderate performance in the baseline benefit less from textual features compared to datasets with low baseline performances. The highest performance is achieved on the SOCCERPLAYER-dataset with 0.79 in F_1. One can argue that when considering the average document length, which is lowest for

soccer players, a low variety in linguistic expressions favors the intra-textual model. However, this schema can not be necessarily observed on other datasets. All in all, the intra-textual model performs well overall, which shows that capturing linguistic structures is a useful (baseline) approach for the slot-filling problem in all tested domains, independent of their average document length.

Intra-textual + Semantic Dependencies Model Performance: Adding background knowledge on top of the intra-textual model further improves the performance in almost all datasets. The highest improvement is observed for the SINGLE and SOCCERPLAYER datasets with 8 and 9 points in precision, and 11 and 15 points in recall. Considering the already high performance of the IT model, moderate improvement is yielded for the datatsets ARCHITECTURALSTRUCTRUE and DAM with 1 and 2 points in F_1, while only a small improvement is observed for the FILM dataset with 1 point in recall. Overall, the experiments show promising results for integrating background knowledge on almost all datasets. Despite our initial intuition, we observed that semantic dependency features have a higher impact on recall rather than on precision. Considering the little improvement on the FILM dataset, we can think of three reasons: i) background knowledge was only insufficiently available for capturing dependencies between the required entities, ii) dependencies between specific entity types are more complex and could not be captured by our proposed queries, or iii) dependencies between some entities of a specific type might not be generalizable. For instance, dependencies between entities that are constant such as places, locations and countries are more reliable than those that change over time, e.g. properties between persons. We leave a more detailed investigation of this for future work. A positive outcome is that incorporating semantic dependencies in no case harms the performance of the model.

4.4 Query Ablation Study

We examine the impact of the proposed query types in a query ablation study. We chose a leave-out study design i.e. removing a single query type from the set of queries. Similar to the first experiments, we report the mean of three runs with a random 80/20 split, each averaged over all datasets. Results are reported in Table 5 as average precision, recall and F_1 over all datasets. The ablation study shows

Table 5. Evaluation results of the query ablation study averaged over all datasets.

Model	**precision**	**recall**	**F_1**
Lin+BK	0.76	0.74	0.75
Lin+BK - Q_1	0.75	0.73	0.74
Lin+BK - Q_2	**0.74**	**0.70**	**0.72**
Lin+BK - Q_3	0.76	0.73	0.74
Lin+BK - Q_4	0.77	0.73	0.75

that the biggest loss of 3 points in F_1-score is yielded when removing the second query (Q_2), which searches the knowledge base for two-hop connections between two slot-filler values. The loss is composed of 2 points in precision and 4 points in recall. When removing Q_1, wee see a slight drop in precision and recall by 1 point. Similar to Q_2, this query searches for direct connections of two slot-filler values. Q_3 creates very general features that capture whether an assigned entity is usually used as slot-filler value for the given slot in the knowledge graph. Removing Q_3 results in a loss by 1 point in recall while the precision remains. Q_4 captures the semantics of two slot-types given their slot-filler values. When removing this query, we observe a slight drop in recall of 1 point, while the precision increases of 1 point. The results show that the highest impact is achieved when integrating one- or two-hop relations between slot-filler values. Nonetheless, the rather little impact of individual queries further shows that a diverse range of queries is beneficial to model semantic relations properly. Future work should consider the impact of using complex queries e.g. increasing the number of possible hops.

5 Conclusion

We investigated the impact of incorporating background knowledge from knowledge graphs into template-based information extraction. We tackled the slot-filling task as a joint inference problem that enables a holistic view on the slot-filling variables within each template during training. Our proposed method is based on querying a knowledge graph with specifically designed triple queries. The results of such queries are used by indicator functions to compute features, capturing semantic relations between slot-filler entities. This approach is motivated by the fact that such semantic relations are often not explicitly expressed in text, but exist e.g. in knowledge graphs.

In our experiments, we showed that integrating background knowledge outperforms a strong baseline model that relies solely on linguistic features by up to 10 points in F_1. This is a remarkable improvement.

In a query ablation study, we investigated the impact of the proposed queries and showed that the most impact is achieved with queries that capture two-hop dependencies between slot-filler values. Generally, it seems beneficial to include a rather diverse set of queries which highlights the need to emphasize query design.

It is possible that the very strong results obtained depend on the fact that corpus and knowledge graph are strongly aligned. Weaker results might be obtained in case that there is a looser connection between corpus and knowledge graph. However, our work clearly shows the potential improvements in information extraction tasks provided we have high quality and wide coverage knowledge graphs that cover and are well aligned with the content and entities in the text collection. One might argue that the need for such an alignment is an assumption that limits the applicability of our method. Our take on this is different. Currently, we have knowledge graphs that are domain-independent and cover prominent or frequent entities only. In fact, recent work has shown that current knowledge graphs are biased to frequent entities and thus do not account for entities on the long tail of the frequency distribution very well [11]. As a community, we thus need to invest in the creation of domain-specific knowledge graphs that cover the terminology and the entities in a specific domain, e.g. medicine, finance, tourism, etc. Only if we have sufficient coverage in specific domains can the approach described here unfold its impact. In future work, we will investigate how to encode the knowledge in a knowledge graph such that deep learning based IE models can also profit from this knowledge.

ACKNOWLEDGEMENTS

This work has been funded by the Federal Ministry of Education and Research (BMBF, Germany) in the PSINK project (project number 031L0028A). To be included.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives, 'Dbpedia: A nucleus for a web of open data', in *The semantic web*, 722–735, (2007).

[2] Razvan Bunescu and Raymond J Mooney, 'Collective information extraction with relational markov networks', in *Proc. of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 438, (2004).

[3] Razvan C Bunescu and Raymond J Mooney, 'A shortest path dependency kernel for relation extraction', in *Proc. of the conference on human language technology and empirical methods in natural language processing*, pp. 724–731, (2005).

[4] George Casella and Edward I George, 'Explaining the gibbs sampler', *The American Statistician*, **46**(3), 167–174, (1992).

[5] Aron Culotta and Jeffrey Sorensen, 'Dependency tree kernels for relation extraction', in *Proc. of the 42nd annual meeting on association for computational linguistics*, p. 423, (2004).

[6] Jeffrey Dalton, Laura Dietz, and James Allan, 'Entity query feature expansion using knowledge base links', in *Proc. of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pp. 365–374, (2014).

[7] George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie M Strassel, and Ralph M Weischedel, 'The automatic content extraction (ace) program-tasks, data, and evaluation.', in *LREC*, volume 2, p. 1, (2004).

[8] Ralph Grishman and Beth Sundheim, 'Message understanding conference-6: A brief history', in *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, volume 1, (1996).

[9] Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min, 'Exploring various knowledge in relation extraction', in *Proc. of the 43rd annual meeting on association for computational linguistics*, pp. 427–434, (2005).

[10] Raphael Hoffmann, Congle Zhang, and Daniel S Weld, 'Learning 5000 relational extractors', in *Proc. of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 286–295. Association for Computational Linguistics, (2010).

[11] Filip Ilievski, Piek Vossen, and Stefan Schlobach, 'Systematic study of long tail phenomena in entity linking', in *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, pp. 664–674, (2018).

[12] Daphne Koller and Nir Friedman, *Probabilistic Graphical Models. Principles and Techniques*, 2009.

[13] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger, 'Factor Graphs and Sum Product Algorithm', *IEEE Transactions on Information Theory*, **47**(2), 498–519, (2001).

[14] John Lafferty, Andrew McCallum, and Fernando CN Pereira, 'Conditional random fields: Probabilistic models for segmenting and labeling sequence data', (2001).

[15] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu, 'Learning entity and relation embeddings for knowledge graph completion', in *Twenty-ninth AAAI conference on artificial intelligence*, (2015).

[16] Yuanfei Luo, Quan Wang, Bin Wang, and Li Guo, 'Context-dependent knowledge graph embedding', in *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1656–1661, (2015).

[17] Andrew McCallum, Karl Schultz, and Sameer Singh, 'Factorie: Probabilistic programming via imperatively defined factor graphs', in *Advances in Neural Information Processing Systems*, pp. 1249–1257, (2009).

[18] Paul McNamee and Hoa Trang Dang, 'Overview of the tac 2009 knowledge base population track', in *Text Analysis Conference (TAC)*, volume 17, pp. 111–113, (2009).

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, 'Distributed representations of words and phrases and their compositionality', in *Advances in neural information processing systems*, pp. 3111–3119, (2013).

[20] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky, 'Distant supervision for relation extraction without labeled data', in *Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pp. 1003–1011. Association for Computational Linguistics, (2009).

[21] Raymond J Mooney and Razvan C Bunescu, 'Subsequence kernels for relation extraction', in *Advances in neural information processing systems*, pp. 171–178, (2006).

[22] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih, 'Cross-sentence n-ary relation extraction with graph lstms', *Transactions of the Association for Computational Linguistics*, **5**, 101–115, (2017).

[23] Hoifung Poon and Pedro Domingos, 'Joint inference in information extraction', in *AAAI*, volume 7, pp. 913–918, (2007).

[24] Chris Quirk and Hoifung Poon, 'Distant supervision for relation extraction beyond the sentence boundary', *arXiv preprint arXiv:1609.04873*, (2016).

[25] Kevin Reschke, Martin Jankowiak, Mihai Surdeanu, Christopher D Manning, and Daniel Jurafsky, 'Event extraction using distant supervision', in *Proc. of the Language Resources and Evaluation Conference (LREC)*, (2014).

[26] Matthew Richardson and Pedro Domingos, 'Markov logic networks', *Machine learning*, **62**(1-2), 107–136, (2006).

[27] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin, 'Relation extraction with matrix factorization and universal schemas', in *Proceedings of NAACL/HLT*, pp. 74–84, (2013).

[28] Sameer Singh, Limin Yao, David Belanger, Ari Kobren, Sam Anzaroot, Mike Wick, Alexandre Passos, Harshal Pandya, Jinho D Choi, Brian Martin, et al., 'Universal schema for slot filling and cold start: Umass iesl at tackbp 2013', (2018).

[29] Noah A. Smith, *Linguistic Structure Prediction*, 2011.

[30] Daniil Sorokin and Iryna Gurevych, 'Context-aware representations for knowledge base relation extraction', in *Proc. of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1784–1789, (2017).

[31] Chang Wang, James Fan, Aditya Kalyanpur, and David Gondek, 'Relation extraction with relation topics', in *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pp. 1426–1436, (2011).

[32] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen, 'Knowledge graph and text jointly embedding', in *Proc. of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1591–1601, (2014).

[33] Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier, 'Connecting language and knowledge bases with embedding models for relation extraction', *arXiv preprint arXiv:1307.7973*, (2013).

[34] M. Wick, K. Rohanimanesh, A. Culotta, and A. McCallum, 'SampleRank. Learning Preferences from Atomic Gradients', in *Proc. of the NIPS Workshop on Advances in Ranking*, pp. 1–5, (2009).

[35] Benjamin Roth Tassilo Barth Michael Wiegand and Mittul Singh Dietrich Klakow, 'Effective slot filling based on shallow distant supervision methods', in *Proc. of NIST KBP workshop*, volume 1, (2013).

[36] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella, 'Kernel methods for relation extraction', *Journal of machine learning research*, **3**(Feb), 1083–1106, (2003).

[37] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning, 'Position-aware attention and supervised data improve slot filling', in *Proc. of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 35–45, (2017).

[38] Zhu Zhang, 'Weakly-supervised relation classification for information extraction', in *Proc. of the thirteenth ACM international conference on Information and knowledge management*, pp. 581–588. ACM, (2004).