Corpus-driven Annotation Enrichment

Felix Kuhr
University of Lübeck
Institute of Information Systems
Ratzeburgerallee 160, 23562 Lübeck
kuhr@ifis.uni-luebeck.de

Bjarne Witten
University of Lübeck
Institute of Information Systems
Ratzeburgerallee 160, 23562 Luebeck
witten@ifis.uni-luebeck.de

Ralf Möller
University of Lübeck
Institute of Information Systems
Ratzeburgerallee 160, 23562 Luebeck
moeller@ifis.uni-luebeck.de

***Abstract*—A reference library can be described as a corpus of an individual composition of documents containing related work of research, documents of favorite authors, or proceedings of a conference. The documents in the corpus may change over time; new documents extend the corpus while other documents are sorted out. A subset of documents may contain meaningful annotations describing their content while other documents contain only weakly annotations. Enriching documents with meaningful annotations is beneficial for the performance of applications like semantic search, content aggregation, automated relationship discovery, query answering and information retrieval. However, enriching a document with meaningful annotations is non-trivial. Available (semi-) automatic annotation tools ignore the individual composition of documents in corpora by annotating documents with generic named-entity related data. In this paper, we present and unsupervised corpus-driven annotation enrichment approach considering the composition of documents and use an EM-like algorithm to enrich weakly annotated documents with meaningful annotations of related documents from the same corpus.**

I. Introduction

In linguistics annotations add additional data to documents, supporting humans, and machines to understand the semantic meaning of words in the document. The degree to which *added value* is brought to a document by enriching the document with annotations depends on the benefit for applications like semantic search, aggregation of content, automated relationships discovery, Query-Answering (QA), Information Retrieval (IR), document retrieval (DR), and Knowledge Management (KM).

In recent years, systems have emerged using methods of Information Extraction (IE) [2] and statistical relational learning (SRL) [15] to extract data from the text of million of randomly selected unstructured documents and derive large graph databases (DBs), representing a symbolic content description using entities and relations. Some of the most known systems are DeepDive [27], NELL [12], YAGO [16], FRED [10], and KnowledgeVault [5]. Annotating documents with data from available graph DBs relates to the entity-linking problem that is a well studied field [4], where entities from documents are linked to entities of graphs. However, matching words in the text of documents to entities that are in a graph DB is difficult having no named-entities in the documents. Even if the documents contain named-entities and it is possible to match them to entities in graph DBs, simply annotating documents with entity-related data from graph DBs leads to annotations weakly describing the document's content and ignore the composition of documents.

Obviously, collecting documents is not an end in itself and the documents in a corpus might represent related work of research, documents of favorite authors, or selective proceedings of conferences. A subset of annotations of a document's annotation database (ADB) may add value to another document's ADB within the same corpus e.g., by increasing the performance in document retrieval. Let us assume that a person is searching for documents about *iterative algorithms* within a personal reference library using some keywords like 'iterative algorithm' or 'EM-like algorithm'. Generally, documents are in the set of relevant documents, if they contain the keywords. However, if a document contain a specific iterative algorithm and the document does not contain the words *iterative algorithm* or *EM-like algorithm*, then the document is not in the set of relevant documents. However, if the keywords are in the document's ADB because of the annotation enrichment process, then the document is part of the relevant documents and possibly useful for the person searching for iterative algorithms in an individual collection of documents. Thus, we are interested in enriching document-specific annotation databases with annotations from the ADBs of similar documents within the same corpus instead of using data from external graph DBs.

In this paper, we present an approach to enrich sparse and weakly annotated documents with annotations of documents in the same corpus taking advantage of the higher purpose in mind of people individually selecting the documents in a corpus. We introduce two holistic similarity measures identifying related documents within a corpus and present an unsupervised EM-like algorithm to identify symbolic content descriptions for document. The algorithm has the following properties: (i) Identifying for each document a set of related documents using both, D- and G-similarity. The D-similarity estimates the similarity using the text of documents and the G-similarity works at the annotation-level. (ii) Iteratively enriching ADBs of documents with annotations of relating documents' ADBs, representing a symbolic content description of the documents. (iii) Annotating new, unseen documents using related documents' annotations, and vice versa.

The remainder of this paper is structured as follows. Section II and III present related work and background information, respectively. In Section IV we introduce corpus-driven annotation enrichment of documents' ADBs. In Section V we provide empirical results presenting the potential of the new approach.

In Section VI we conclude and present feature work.

II. Related Work

Over the recent years, a considerable number of automatic annotation systems have been introduced in the natural language processing (NLP) community. Automatic annotation systems use human language to directly extract data from the text of documents. A well-established technique is named-entity recognition (NER), which is a subtask of IE taking an unannotated block of text and producing an annotated block of text that highlights the names of entities and classify them into predefined categories such as persons, organizations, locations, etc. Some annotation systems extract named-entities from the text and use available DBs to identify more entities having a relationship to the extracted entities by using link prediction [11], which is the discipline of estimating the likelihood of the existence of a link between nodes, using the given links and attributes of nodes within a graph [18]. The granularity of annotations depends on the application and a single annotation may cover a word, a sentence, a paragraph, a document, or an entire corpus [9].

MINTE [1] is an approach for semantically integrating RDF graphs. This requires the management of data to determine the relatedness of different RDF representations of the same entity. Tipalo [8] is an automatic system identifying types from the text of Wikipedia documents for DBpedia entities. OpenCalais [14] is a knowledge extraction tool by Reuters, which automatically tags data in unstructured text using a large ontology. SemTag is a module of Seeker, both introduced by Dill et al. in [20] to generate semantic annotations. SemTag uses a structural analysis of text and the ontology *TAP* to automatically annotate documents with data from the ontology. Analogous to SemTag, KIM is a semantic annotation, indexing, and retrieval platform, developed by Ontotext [21] that identifies entities in the text of documents and links the entities to semantic descriptions which are provided by the KIMBO ontology (pre-populated ontology with many instances). The platform allows KIM-based applications to use it for automatic semantic annotation. KIM has been applied in different domains like anti-corruption and asset recovery, analysis of bio-medical content, or scientific papers. BOEMIE [13] is another approach, focusing on text block locations that correspond to specific types of named-entities, and additionally performs annotations of text that refers to the same topic to automatically creating annotations. For further annotation systems please refer to the survey of from Oliveira et al. [7].

Generally, all available annotation systems ignore the composition of documents and simply add data from external sources to documents to describe entities occurring in the documents. However, we believe that identifying DBs that are useful for annotating documents is as difficult as identifying a human expert adding high added value annotations to documents. Even if external DBs are available, we take the view that separately annotating documents simply adds domain-specific data to documents instead of describing the content of documents with respect to the document composition.

Compared with existing automatic annotation systems, the contributions of this paper are: 1. a novel corpus-driven annotation enrichment approach that considers the composition of documents in the annotation process instead of simply adding data from knowledge bases (KBs) or formulas from ontologies; 2. a flexible annotation approach that allows the annotation of documents both with additional external data and without external data using well annotated documents to enrich the annotations of other documents within the same corpus.

III. Preliminaries

Topic modeling techniques estimate topics from a collection of documents and calculate for each of the documents a topic probability distribution θ. Topics represent co-occurring words of the documents. The statistical technique called latent Dirichlet allocation (LDA)[19] generates a topic model from a set of documents to identify latent structures such as the topic distribution of documents and word topic distribution. LDA uses a bag of words approach simplifying documents. For document d, LDA learns a discrete probability distribution θ_d that contains for each topic $k \in \{1, \ldots, K\}$ a value between 0 and 1. The sum over all K topics for d is 1. To find topically similar documents we use the Hellinger distance [26] measuring the distance between two probability distributions. Given two topic distributions θ_{d_i} and θ_{d_j} for documents d_i and d_j, the Hellinger distance $H(\theta_{d_i}, \theta_{d_j})$ is given by $\frac{1}{\sqrt{2}}\sqrt{\sum_{k=1}^{K}\left(\sqrt{\theta_{d_i,k}} - \sqrt{\theta_{d_j,k}}\right)^2}$ where k refers to the topics in the documents. Topic modelling techniques reduce the dimensionality of each document to the number of topics k. Having the topic model for the documents within corpus $\mathcal{D}$, it is feasible to calculate the Hellinger distance between documents d_i and d_j. The result is a value between 0 and 1 and $H(\theta_{d_i}, \theta_{d_i}) = 0$. LDA has input documents d_i, $i \in \{1, ..., D\}$, where each document d_i contains words w_n; $n \in \{1, ..., N\}$. The per-word topic assignment $z_{d,n}$ is drawn from a per-document topic distribution vector θ_d. Each topic $k \in \{1, ..., K\}$ is a multinomial distribution of words w. LDA contains two hyperparameters α and β, where α conditions the per-document topic distributions θ_d and β conditions the per-corpus topic distributions ϕ_k, $k \in \{1, ..., K\}$.

Information extraction is a subdomain of NLP referring to methods that extract entities and their relations from text documents. Two main tasks of IE systems are NER and relation extraction. A possible result of an IE system is a set of Resource Description Framework (RDF) triples containing the extractable relations between entities. Identifying entities and relations within arbitrary long sentences containing subordinate clauses and other grammatical structures make IE difficult. Systems are OpenIE [2], Textrunner [17], Gate [6], and the framework document spanners [3]. We use OpenIE which learns a classifier to split sentences of text documents into shorter utterances and apply natural logic [23] to further shorten the utterances in a way such that the shortened utterances can be mapped to OpenIE triples representing subject, predicate, and object.

IV. Corpus-driven Annotation Enrichment

In this section, we present the annotation enrichment process enriching ADBs of documents with annotations of related documents in the same corpus. Simply considering all annotations from d-related documents may include many annotations not describing the content of d. One approach to avoid enriching an ADB with annotations that do not describe the content of the corresponding document is to enrich the ADB only with annotations sharing named-entities occurring in the text of the document. Then, new annotations add relations to already known named-entities. However, focusing only on annotations including named-entities not necessarily leads to annotations describing the content of documents. We assume that the annotations of one document add value to another document in the same corpus, if the content of both documents is somehow related. Thus, enriching ADB g_e with annotations from d_e-related documents requires the identification of d_e-related documents and those annotations that are semantically related to the annotations of document d_e. We introduce the D- and G-similarity identifying the d_e-related documents in corpus $\mathcal{D}$ and present an iterative algorithm using the two similarity measures identifying d_e-related documents to assign each annotation with an Expected Relevance Value (ERV) to identify the annotations describing the content of d_e without focusing on named-entities.

A. D-Similarity

D-similarity is based on the idea of topic models and compares the relatedness between two documents using the similarity of the documents' topics. The document-specific topic vector is known as the topic distribution of a document. D-Similarity is defined by:

$$Sim_D(d_e, d_k) = 1 - H(\theta_{d_e}, \theta_{d_k}), \tag{1}$$

where $H(\theta_{d_e}, \theta_{d_k})$ estimates the Hellinger distance between the topic distributions of d_e and d_k and $Sim_D(d_e, d_k) \in [0, 1]$. The interval follows directly from the definition of the Hellinger distance. The higher the D-similarity the more similar the documents' topic distribution. The text of documents d_e and d_k having a high D-similarity contain similar content such that annotations for d_k might be added value for d_e.

Comparing two documents using the D-similarity requires both documents having a topic distribution. But, how to compare the D-similarity between a document $d \in \mathcal{D}$ and a new document $d' \notin \mathcal{D}$? Using the parameters of the topic model generated from all documents in $\mathcal{D}$ it is possible to infer the topic distribution for a new document $d' \notin \mathcal{D}$ by applying the *folding in* Gibbs sampling technique [22], which is the same as Gibbs sampling [24], except the sampling bases on the topic-word distribution ϕ and per document-topic distributions θ of the topic model of documents in $\mathcal{D}$. First, for each word w in d' the most probable topic is initialized using ϕ. If d' contains a new word w not part of any document $d \in \mathcal{D}$, we randomly assign the topic. Second, Gibbs sampling estimates the topic distribution of d'. This means that it is only required to perform Gibbs sampling for the words in the new document to infer the topic distribution of document d'. After extending $\mathcal{D}$ with some documents, it is useful to create a new topic model from all documents in $\mathcal{D}$ such that the topic models' parameters depend on all documents.

B. G-Similarity

G-similarity identifies d_e-related documents in $\mathcal{D}$ comparing annotations of g_e with annotations of other documents' ADB. Each document corresponds to a specific graph ADB which contains the annotations of the corresponding document. Technically, we assume an annotation to be a triple containing a subject (s), predicate (p), and object (o). Comparing the annotations in g_e with those in g_k is the same as identifying subgraph matches between g_e and g_k using labeled vertices and edges in both graphs. Thus, we introduce the G-similarity which identifies subgraph matches between the annotations of two ADBs and bases on the assumption that semantically related documents have at least parts of a subset of annotations in common. We define a similarity function $s(g_e^i, g_k^j)$ calculating a similarity score between two annotations, comparing the i-th annotation in g_e with the j-th annotation in g_k using the entities and relations to estimate a similarity score in $[0, 1]$. The more similar two annotations $g_e^i \in g_e$ and $g_k^j \in g_k$ the higher $s(g_e^i, g_k^j) \in [0, 1]$ and define $s(g_e^i, g_k^j)$ by:

$s(g_e^i, g_k^j)$ is 0, if $(s^i \neq s^j \wedge p^i \neq p^j \wedge o^i \neq o^j)$, $\frac{1}{3}$, if $(s^i = s^j \wedge p^i \neq p^j \wedge o^i \neq o^j)$ or $s^i \neq s^j \wedge p^i = p^j \wedge o^i \neq o^j)$ or $(s^i \neq s^j \wedge p^i \neq p^j \wedge o^i = o^j)$, $\frac{2}{3}$, if $(s^i = s^j \wedge p^i = p^j \wedge o^i \neq o^j)$ or $(s^i \neq s^j \wedge p^i = p^j \wedge o^i = o^j)$ or $(s^i = s^j \wedge p^i \neq p^j \wedge o^i = o^j)$, and 1, if $(s^i = s^j \wedge p^i = p^j \wedge o^i = p^j)$.

Calculating the G-similarity between g_e and g_k requires annotation-wise comparison of each annotation in g_e with all annotations in g_k using the similarity score $s(g_e^i, g_k^j)$ for all i and j. Matrix $\mathcal{M}$ is an $m \times n$ matrix, where m is the number of rows and n is the number of columns. $\mathcal{M}$ represents all possible similarity scores between annotations in g_e and g_k where $m = |g_e|$ and $n = |g_k|$, such that $a_{i,j}$ represents the similarity score for $s(g_e^i, g_k^j)$. It is possible that two annotations within two ADB have nothing in common. Hence, we use $\mathcal{M}$ to identify the best match for each annotation in g_e and all annotations in g_k, and vice versa. $v^c \in \mathbb{R}^n$, with $v_j^c = max_i\ a_{i,j}$ represents the similarity vector containing for each annotation in g_e the highest possible similarity score and $v^r \in \mathbb{R}^m$, with $v_i^r = max_j\ a_{i,j}$ represents the similarity vector containing for each annotation in g_k the highest possible similarity score. The G-similarity is defined as $Sim_G(g_e, g_k) = \frac{1}{2} \cdot (\overline{v^c} + \overline{v^r})$, where $\overline{v^c}$ and $\overline{v^r}$ represents the average value of the similarity vectors taking the ratio between high and low similarity scores into account such that two ADBs g_e and g_k sharing only a small number of high similarity scores and a high number of low similarity scores have a small G-similarity. We normalize $Sim_G(g_e, g_k)$ to the interval $[0, 1]$.

C. Iterative Annotation Enrichment Algorithm

In this section we present the iterative annotation enrichment algorithm in Algorithm 1, which bases on Dempster et al.

[25]. Their EM-algorithm estimates the maximum likelihood of parameters handling unobserved variables alternating between the expectation and maximization step. The expectation step creates a function for the expectation of log-likelihood using the present values for the parameters. The maximization step calculates the parameters maximizing the expected log-likelihood in the expectation step. The expectation step of Algorithm 1 identifies d_e-related documents, represented as $\mathcal{D}^{d_e}$, using D- and G-similarity and calculates for all annotations $\mathcal{G}^{d_e}$ the ERV value. The maximization step calculates the new average G-similarity optimizing the ERVs in the next expectation step. We define ERV to estimate only the annotations in $\mathcal{G}^{d_e}$ describing the semantic meaning of the content from document d_e as $ERV_t^{d_e} = \overline{Sim_{D_t}} \cdot \overline{Sim_{G_t}} \cdot f(t)$, where $\overline{Sim_{D_t}}$ is the average D-similarity of documents $d \in \mathcal{D}^{d_e}$ such that g contains annotation t, $\overline{Sim_{G_t}}$ is the average G-similarity of all ADBs $g \in \mathcal{G}^{d_e}$ containing annotation t and $f(t)$ is the frequency of $g \in \mathcal{G}^{d_e}$ containing annotation t. Obviously, the definition for the ERV depends on the annotations we are interested in. We include the frequency to increase the rank of recurrent annotations. The average D-similarity of documents where the corresponding ADBs contain annotation t is given by $\overline{Sim}_{D_t^{d_e}}$. $\overline{Sim}_{G_t^{d_e}}$ represents the average G-similarity containing annotation t and $\overline{ERV^{d_e}}$ is the average ERV of all annotations in d_e. There are two ways leading to a high ERV. First, D- and G-similarity between d_e and $\mathcal{D}^{d_e}$ is high which means the text of each $d \in \mathcal{D}^{d_e}$ and d_e is semantically related. Second, the number of documents in $\mathcal{D}^{d_e}$ containing annotation t is high. Thus, enriching ADB of d_e with annotation t occurring in many other ADB may add value to the ADB of d, because it seems to be generic or very specific for those documents. The input parameters of Algorithm 1 are document d_e, g_e, $\mathcal{D} \setminus \{d_e\}$, and D-similarity selection threshold τ. The output is the optimal ADB g'_e. In the E-Step, the algorithm updates variable erv_t for each annotation t in $\mathcal{G}_e^d$. The algorithm adds annotations with high ERV to ADBs and ignores annotations with low ERV. In the M-Step, the algorithm updates the average G-similarity $\overline{Sim_{G^{d_e}}}$ which is part of the termination condition in line 5.

Theorem 1: Algorithm 1 is correct and terminates for each $\epsilon \in \mathbb{R}^+$ and a finite set of documents $\mathcal{D}$.

Proof: Let ϵ be greater than zero and $\mathcal{D}$ be a finite set. Then, Algorithm 1 terminates if the condition in line 5 is not fulfilled anymore. In the M-step of Algorithm 1 we update the average G-similarity $\overline{Sim_{G^{d_e}}}$. This approach prunes the solution space for the next iteration because the new average G-similarity leads to a more stringent condition in line 11 compared to the previous iteration. In each iteration $\mathcal{D}^{d_e}$ represents the set of documents such that $\mathcal{G}^{d_e}$ contains only annotations having a G-similarity greater than the average G-similarity. In each iteration we optimize the annotations in $\mathcal{G}^{d_e}$ calculating for each of the annotations a new ERV value. Hence, the algorithm extends g'_e only with annotations best describing the semantic meaning of d_e. ∎

The complexity of Algorithm 1 depends only on the number of documents in the corpus $\mathcal{D}$ (n) and the number of sets of annotations in $\mathcal{G}$ (m). For each document $d \in \mathcal{D}$ we are interested in enriching its ADB with annotations from related documents' ADB. Algorithm 1 estimates for each document the set of related documents using the D- and G-similarity. Algorithm 1 extends g_e only with the most similar annotations in $\mathcal{G}^{d_e}$ instead of enriching g_e with all annotations in d_e-related documents' ADBs. This is an important step to reduce the number of non-added value annotations in g_e. The worst case complexity for calculating for each document $d \in \mathcal{D}$ both similarity measures is in $\mathcal{O}(m^2 n)$ *(line 9)*. Enriching database g for each document d in $\mathcal{D}$ *(line 11 to 12)* has the worst case complexity $\mathcal{O}(m^2 n^2)$.

Algorithm 1 Iterative Annotation Enrichment

1: **Input**: d_e, g_e, $\mathcal{D} \setminus \{d_e\}$, τ
2: **Output**: g'_e
3: **Define:** $\epsilon = 0.1$, $\mathcal{D}^{d_e}$, $\mathcal{D}'^{d_e}$, $\mathcal{G}^{d_e}$, g'_e
4: **Initialize:** $\overline{Sim_{G^{d_e}}} = \epsilon$, $\overline{Sim'_G} = \overline{Sim_{G^{d_e}}} - \epsilon$, $\mathcal{D}^{d_e} = \emptyset$, $\mathcal{G}^{d_e} = \emptyset$, $erv_t^{d_e} = 0$, $g'_e = \emptyset$
5: **while** $|\overline{Sim_{G^{d_e}}} - \overline{Sim'_{G^{d_e}}}| \geq \epsilon$ **and** $\overline{Sim_{G^{d_e}}} > \overline{Sim'_G}$ **do**
6: $g'_e \leftarrow g_e$
7: $\mathcal{D}^{d_e} \leftarrow \emptyset$ ▷ E-Step
8: **for each** $d_k \in \mathcal{D}$ **do**
9: **if** $Sim_D(d_e, d_k) > \tau$ **and** $Sim_G(g'_e, g_k) > \overline{Sim_{G^{d_e}}}$ **then**
10: $\mathcal{D}^{d_e} \leftarrow \mathcal{D}^{d_e} \cup \{d_k\}$
11: **for each** $t \in \mathcal{G}^{d_e}$ **do**
12: $erv_t^{d_e} \leftarrow erv_t^{d_e} + ERV_t^{d_e}$
13: **for each** $t \in \mathcal{G}^{d_e}$ **do**
14: **if** $ERV_t^{d_e} > \overline{ERV}^{d_e}$ **then**
15: $g'_e \leftarrow g'_e \cup \{t\}$
▷ M-Step
16: $\overline{Sim'_{G^{d_e}}} = \overline{Sim_{G^{d_e}}}$
17: $\overline{Sim_{G^{d_e}}} = \frac{\sum_{k=1}^{|\mathcal{D}^{d_e}|} Sim_{G^{d_e}}(g_e, g_k)}{|\mathcal{D}^{d_e}|}$
18: **return** g'_e

Algorithm 1 calculates for $n-1$ documents the D-similarity $Sim_D(d_e, d_k)$ and G-similarity $Sim_G(g_e, g_k)$, where D-similarity has a complexity $\mathcal{O}(1)$ given the topic distribution for documents in $\mathcal{D}$. The constant complexity is given by the number of K multiplications of discrete topic distributions in the D-similarity. G-similarity has a worst case complexity of $\mathcal{O}(m^2)$ because it is theoretically possible that g_e and g_k might contain all possible annotations of $\mathcal{G}$. In practise, the document-specific ADBs contain only few annotations having a relation to the content of the related document. Hence each g_i is small and $|g_i| << n$. Additionally, for each t in $\mathcal{G}^{d_e}$ ERV is calculated in constant time $\mathcal{O}(1)$ *(line 12)*. Afterwards, Algorithm 1 filters all annotations t having an expected relevance value less than the average expected relevance value $\overline{ERV}^{d_e}$ *(line 14 to 15)*. For both steps, the algorithm iterates over $\mathcal{G}^{d_e}$ having a theoretically size of m annotations. This leads to a complexity of $\mathcal{O}(m)$. Hence, worst case complexity for calculating the annotations for each document is given by $\mathcal{O}(m^2 n)$. In total Algorithm 1 has a worst case complexity for estimating the annotations of one document of $\mathcal{O}(n^2 m^2)$. Applying Algorithm 1 to each document in $\mathcal{D}$ leads to the complexity $\mathcal{O}(n^3 m^2)$.

However, in practise the number of documents $d_k \in \mathcal{D}$ (n'), being similar to document d_e, is small ($n' \ll n$) and the rank of the similarity matrix $\mathcal{M}$ (m') is small, too ($m' \ll m$). Additionally, the number of iterations for each document is only a fraction of n (see Section V).

V. Empirical Results

In this section, we present empirical results of the corpus-driven annotation enrichment approach of documents' ADB using annotations of related documents in the same corpus. We present the performance of the iterative annotation database enrichment process where Algorithm 1 identifies for each document $d \in \mathcal{D}$ valuable annotations from relating documents extending ADB g such that the annotations in g describe the content of document d. We analyse the performance of Algorithm 1 using the following three datasets relating to car manufacturers and US universities/organizations.

- Dataset 1: Documents relating to cars from BMW
- Dataset 2: Documents related to cars from Mercedes
- Dataset 3: Documents from US universities and institutions.

$\mathcal{D}$ contains for each dataset a set of documents from Wikipedia and each document represents an article. For each $d \in \mathcal{D}$ the corresponding ADB } contains data from *DBpedia* KB. This means if d_e contains text from the article *car* then g_e contains the corresponding data of entity *car* from the DBpedia knowledge base. The data in DBpedia's KB is extracted in a crowd-sourced community effort and mostly about structured content like infoboxes and tables within the Wikpedia articles.

We assume, that enriching documents with annotations is even more difficult if human experts directly generate the annotations instead of starting from a graph DB annotation generated by a graph system. Although, the goal of corpus-driven enrichment of documents is identifying annotations from unstructured content of related documents which describe the meaning of the documents. We assume that the data available in DBpedia is a good starting point to the describe the content of Wikipedia articles, because DBpedia contains data from the articles. Obviously, we can use any aforementioned information extraction system as a starting point to extract annotations directly extractable from the Wikipedia articles. However, the annotations from DBpedia have a high quality and we are able to use them directly from the DBpedia KB instead integrating third party software. Next, we give a brief overview of preprocessing documents including details about the hyperparameters for D- and G-similarity.

A. Data Preprocessing

We implement Algorithm 1 as a Java program and use the `MALLET` [22] library for topic modeling with the following parameters: (i) $\alpha = 0.01$, (ii) $\beta = 0.01$, and (iii) 1000 iterations for the model in library `MALLET`. We analyse the performance of Algorithm 1 using different settings for topics within the topic model ($k = 5, 10, 20, 30$) to identify the impact of parameter k. Generally, the number of topics within the topic model is highly dependent on the documents in $\mathcal{D}$. We use `MALLET` to preprocess all text documents in corpus $\mathcal{D}$ by: i) lowercasing all characters, ii) stemming words, iii) tokenizing the result, and iv) eliminating tokens part of a stop-word list which contains 524 words. Each dataset requires additional preprocessing using the following three steps: 1) Analysing the annotating data in each ADB $g \in \mathcal{G}$ and marking annotations occurring in at least two different ADBs such that the algorithm can annotate the annotation database g_e of document d_e even if we remove some annotations from g_e, because the removed annotations are part of ADB of other documents. 2) The second step contains two parts: i) Choose one document d_e from corpus $\mathcal{D}$ and remove (r) 90% of the annotations in the corresponding ADB g_e. We remove 90% of annotations which are in at least one ADB of another document in $\mathcal{D}$. Furthermore, we remove 90% of annotations which are only in g_e. ii) Generating topic model, from $\mathcal{D} \setminus \{d_e\}$ for each document d_e in $\mathcal{D}$ which results in $|\mathcal{D}|$ different corpora $\mathcal{D}_1$ to $\mathcal{D}_{|\mathcal{D}|}$, each containing $|\mathcal{D}| - 1$ documents. 3) Infer topic distribution for document d_e using parameters of the corresponding topic model from documents $\mathcal{D} \setminus \{d_e\}$ generated in the second step. Inference bases on the *folding in* Gibbs sampling technique [22]. Topic distribution θ_{d_e} enables comparison of two documents using the D-similarity.

Next, Algorithm 1 estimates for each document d_e the annotations in $\mathcal{G}^{d_e}$. Algorithm 1 iteratively enriches each annotation database g_e with annotations from all sets in $\mathcal{G}^{d_e} \setminus \{g_e\}$. We perform Algorithm 1 $|\mathcal{D}|$ times to use each of the corpora. As explained before, D-similarity and G-similarity are holistic measures and bases on all documents in $\mathcal{D}$. Thus, the document-specific annotations depend on the documents in $\mathcal{D}$.

B. Database Enriching

We perform three steps enriching the ADB g_e of document d_e with data from related documents: (i) Estimate d_e's topic distribution θ_{d_e}, (ii) identify d_e-related documents and append $\mathcal{D}^{d_e}$ with all d_e-related documents. (iii) Calculate the expected relevance value for each annotation $t \in \mathcal{G}^{d_e}$. Algorithm 1 optimizes ADB g for each d by enriching documents' ADBs with new data from ADBs of related documents.

Next, we evaluate the performance and necessary number of iterations in Algorithm 1 using the true positive rate (tpr), positive predictive value (ppv), and F-measure. In classification tasks, tpr is the number of true positives divided by the number of all items labeled as belonging to the positive class. ppv is the fraction of true positives and the total number of items that belong to positive class. As we have assumed, the performance of Algorithm 1 highly depends on the number of topics and the optimal number of topics depends on the documents in the corpus. For the first two datasets the algorithm has best results using 30 topics. Interestingly, the annotation performance using only 5 topics vary slightly from the results using 30 topics for both datasets. For the third dataset $k = 30$ leads to best results using a D-similarity selection threshold between

doc_17484	**doc_7071**
M41,subject,Category:BMW_vehi.	Z3,hypernym,Car
M41,hypernym,Car	Z3,hypernym,Engine
	Z3,configuration,Diesel_engine
doc_3407	**doc_45969**
1_Series,subject,Cat.:Roadsters	M_Roadster,bodyStyle,Coupe
1_Series,conf.,Diesel_engine	M_Roadster,hypernym,Car
1_Series,hypernym,Engine	M_Roadster,hypernym,Engine
	M_Roadster,subject,cat:1960s_auto.

TABLE I: Example of associative annotations in dataset 1.

0.01 and 0.2. For higher D-similarity selection thresholds 5 topics results in a better performance. Identifying relating documents using the G-similarity compares the documents on the annotation level using annotations in the ADBs. Figure 1 presents the influence of both, the D- and G-similarity within the ADB enrichment process for each of the three datasets. Algorithm 1 identifies for all three datasets the most possible annotations using a G-similarity selection threshold between 0.0 and 0.6. Figure 3 confirms this result. F-measure for dataset 1 and 2 are similar for G-similarities between 0.01 and 0.1.

Next, we analyse the performance of Algorithm 1 for varying D-similarity selection thresholds. Figure 4 presents the performance for all three datasets using the following parameters: k = 30, r = 0.90, ϵ = 0.01, $\overline{Sim_{G^{d_e}}}$ = 0.1. The tpr increases for all datasets up to a selection threshold of 0.7 while ppv is slightly decreasing at the same time. $F - Base$ is the baseline representing F-measure of random guesses.

Figure 3 presents the F-measure for k = 10,20,30, varying $\overline{Sim_{G^{d_e}}}$, and an average D-similarity between 0.1 to 0.9. 30 topics leads to the best results for all datasets independent from $\overline{Sim_{G^{d_e}}}$. Figure 2 presents the number of iterations for all documents in the datasets having a fixed D-similarity (0.25) and fixed number of topics (30), considering only the documents fulfilling the condition in line 5 of Algorithm 1 at least once. The median for all three datasets is less or equal five iterations.

The ppv in Figure 4 increases with increasing D-similarity threshold for all three datasets. The tpr seems to be low. The reason for the positive predictive value is the ground truth. Algorithm 1 enriches documents ADBs with annotations from related documents which are not in the documents corresponding DBpedia KB which leads to a high number of false-negative annotations within the evaluation. However, we are interested in enriching documents with annotations adding value the documents ADB and we are not interested in generating exactly the same annotations available in DBpedia. There is no single correct ADB for a document, even if two different human experts would annotate the same document. Thus, only a subset of false-negative annotations bring no value to the documents' ADBs and might be useless in describing the content of the corresponding documents.

C. Associative Annotations

Algorithm 1 enriches the ADBs of documents with annotations from ADBs of related documents. After the iterative annotation process, we identify many annotations in ADBs describing the meaning of the corresponding documents which are not in DBpedia's KB. We call those annotations associative annotations describing the content of the document, but are not linked to the corresponding entity in DBpedia's KB acting as ground truth in the case study. Obviously, associative annotations have a relation to the content of documents and enrich the annotation database with based on the composition of documents within the corpus. Table I presents some annotations identified by Algorithm 1 which are not extractable by DBpedia's KB. However, some of the associative annotations are suitable content descriptions for the documents. In the evaluation of Algorithm 1, we mark associative annotations as false positive, because they are not in the data acting as ground truth. Thus, the results might be even better if human experts rate the added value of each annotation. The content of document d_e (*doc_7071*) describes the two-seat convertible and coupe *Z3* from BMW. Algorithm 1 enriches the ADB g_e with annotations (BMW_Z3, hypernym, Car), (BMW_Z3, hypernym, Engine), and (BMW_Z3, configuration, Diesel_engine). Obviously, Z3 is a car, but the authors of the article do not explicitly write that BMW Z3 is a car even though the word *car* occurs 8 times in d_e. Enriching g_e with annotation (BMW_Z3, hypernym, Car) is beneficial for people searching for all car models manufactured by BMW. The other two associative annotations add value to d_e, too. The engine of all BMW Z3 models requires fuel and there exists no original BMW Z3 model with an diesel engine. Associative annotation (BMW_Z3, configuration, Diesel_engine) is wrong if we think about original Z3 configurations. However, enriching g_e with annotation (BMW_Z3,configuration, Diesel_engine) adds value to the document retrieval and query answering, because people searching for the Z3 model containing a diesel engine might receive document *doc_7071* because of the new annotation and can directly extract from the document's content that there exist no original Z3 model with a diesel engine. The content of a second document d'_e (*doc_3407*) describes the Series 1 from BMW. Some models of this Series have a diesel engine. The authors of document d'_e mention three times the word *diesel* in the text but the corresponding DBpedia KB does not contain annotation (BMW_1_Series, configuration, Diesel_engine). However, Algorithm 1 enriches g'_e with the associative annotation that is again beneficial for applications like document retrieval and query answering. Algorithm 1 enriches (BMW_1_Series, subject, Category: Roadsters) to the ADB of *doc_3407*. This annotation only imprecisely describes the BMW 1 series. All cars from this Series has four seats and some of the cars are convertible. A short definition of a roadster is the following: open two-seat cars of sporting appearance or character. We assume that Algorithm 1 enriches g'_e with this annotation, because the text in d_e is similar to some other documents being from category roadsters. There exist models in this series which are convertibles and the associative annotation (BMW_1_Series, subject, Category: Roadsters) might be useful for people searching for convertibles from BMW, because many people use roadster and convertible interchangeable.

Fig. 1: Left plot: Influence of D-Similarity identifying annotations depending on the number of topics for Dataset 1 (lines), Dataset 2 (dashed), and Dataset 3 (dotted). Right plot: Influence of G-Similarity in identifying annotations.

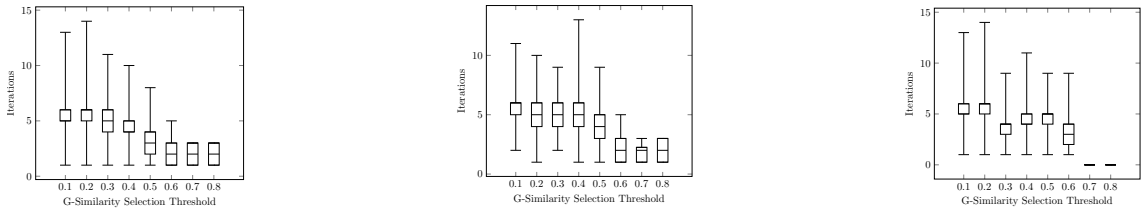

Fig. 2: Number of necessary iterations for dataset 1 (left), dataset 2 (center), and dataset 3 (right) until Algorithm 1 reaches a fixed point, depending on Sim_G.

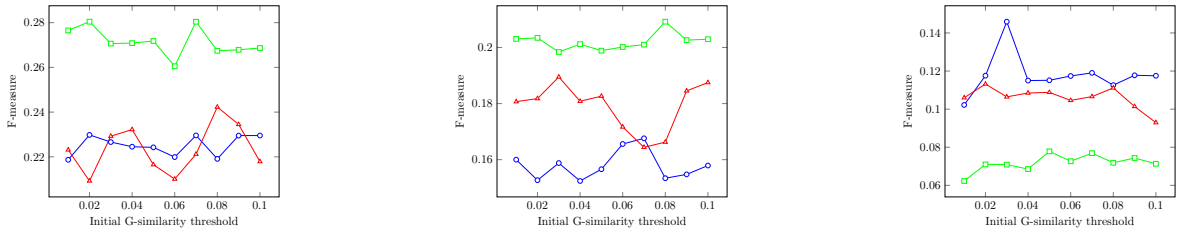

Fig. 3: Performance of Algorithm 1 for all three datasets using 10 (blue lines), 20 (red lines), and 30 (green lines) topics and varying the initial G-similarity threshold from 0.01 to 0.1

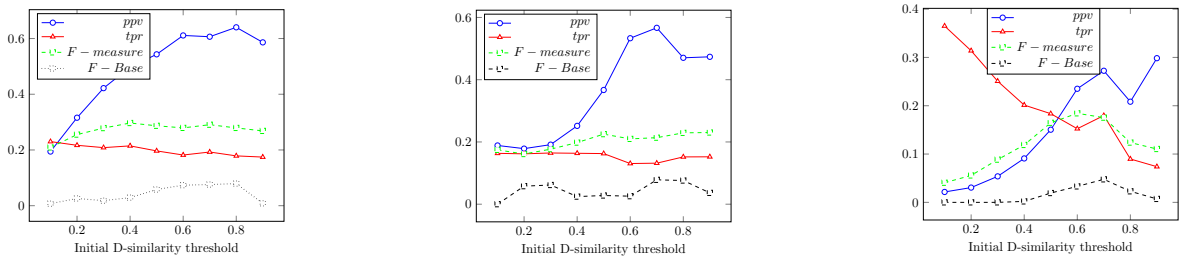

Fig. 4: Performance depending on D-similarity; $k = 30$, $r = 0.90$, $\epsilon = 0.01$, $\overline{Sim_{G^{de}}} = 0.1$. F-Base represents the F_1-score for randomly guessing possible annotations for ADBs.

VI. Conclusion

In this paper, we have introduced an unsupervised corpus-driven annotation enrichment approach considering the composition of documents and have used an EM-like algorithm to enrich weakly annotated documents with meaningful annotations of related documents from the same corpus. To the best of our knowledge this is the first corpus-driven annotation enrichment model rest upon a combination of two holistic measures to enrich documents with annotations of related documents. In the context of our case study, we conclude that the approach enriches ADBs with annotations representing a valuable content description. Algorithm 1 has a positive predictive value of up to 0.72 for different documents in dataset 1 and 0.96 for some documents in the second dataset. In future work we will extend the algorithm to independently handle D- and G-similarity. Actually, the algorithm cannot identify annotations describing the semantics of documents when only D- or G-similarity is high. Another idea is to learn the thresholds for D- and G-Similarity for each corpus individually.

References

[1] Diego Collarana et al. “MINTE: semantically integrating RDF graphs”. In: *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics*. ACM. 2017, p. 22.

[2] Gabor Angeli, Melvin Johnson Premkumar, and Christopher D Manning. “Leveraging linguistic structure for open domain information extraction”. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015)*. 2015.

[3] Ronald Fagin et al. “Document spanners: A formal approach to information extraction”. In: *Journal of the ACM (JACM)* 62.2 (2015), p. 12.

[4] Wei Shen, Jianyong Wang, and Jiawei Han. “Entity linking with a knowledge base: Issues, techniques, and solutions”. In: *IEEE Transactions on Knowledge and Data Engineering* 27.2 (2015), pp. 443–460.

[5] Dong, Xin Luna and Gabrilovich, Evgeniy and Heitz, Geremy and Horn, Wilko and Murphy, Kevin and Sun, Shaohua and Zhang, Wei. “From data fusion to knowledge fusion”. In: *Proceedings of the VLDB Endowment* 7.10 (2014), pp. 881–892.

[6] Hamish Cunningham et al. “Getting more out of biomedical documents with GATE’s full lifecycle open source text analytics”. In: *PLoS computational biology* 9.2 (2013), e1002854.

[7] Pedro Oliveira and João Rocha. “Semantic annotation tools survey”. In: *Computational Intelligence and Data Mining (CIDM), 2013 IEEE Symposium on*. IEEE. 2013, pp. 301–307.

[8] Aldo Gangemi et al. “Automatic typing of DBpedia entities”. In: *International Semantic Web Conference*. Springer. 2012, pp. 65–81.

[9] ISO ISO. “24612: 2012 Language resource managementLinguistic annotation framework (LAF)”. In: *Project leader: Nancy Ide* (2012).

[10] Valentina Presutti, Francesco Draicchio, and Aldo Gangemi. “Knowledge extraction based on discourse representation theory and linguistic frames”. In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2012, pp. 114–129.

[11] Linyuan Lü and Tao Zhou. “Link prediction in complex networks: A survey”. In: *Physica A: statistical mechanics and its applications* 390.6 (2011).

[12] Carlson, Andrew and Betteridge, Justin and Kisiel, Bryan and Settles, Burr and Hruschka Jr, Estevam R and Mitchell, Tom M. “Toward an Architecture for Never-Ending Language Learning.” In: *AAAI*. Vol. 5. 2010.

[13] Pavlina Fragkou et al. “BOEMIE Ontology-Based Text Annotation Tool.” In: *LREC*. Citeseer. 2008.

[14] Thomson Reuters. “OpenCalais”. In: *Retrieved June* 16 (2008).

[15] Lise Getoor and Ben Taskar. *Introduction to statistical relational learning*. MIT press, 2007.

[16] Suchanek, Fabian M and Kasneci, Gjergji and Weikum, Gerhard. “Yago: a core of semantic knowledge”. In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 697–706.

[17] Alexander Yates et al. “Textrunner: open information extraction on the web”. In: *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. Association for Computational Linguistics. 2007, pp. 25–26.

[18] Lise Getoor and Christopher P Diehl. “Link mining: a survey”. In: *Acm Sigkdd Explorations Newsletter* 7.2 (2005), pp. 3–12.

[19] David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent dirichlet allocation”. In: *Journal of machine Learning research* 3.Jan (2003).

[20] Stephen Dill et al. “SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation”. In: *Proceedings of the 12th international conference on World Wide Web*. ACM. 2003, pp. 178–186.

[21] Borislav Popov et al. “KIM–semantic annotation platform”. In: *International Semantic Web Conference*. Springer. 2003, pp. 834–849.

[22] Andrew Kachites McCallum. “MALLET: A Machine Learning for Language Toolkit”. http://mallet.cs.umass.edu. 2002.

[23] Víctor Manuel Sánchez Valencia. *Studies on natural logic and categorial grammar*. Universiteit van Amsterdam, 1991.

[24] Stuart Geman and Donald Geman. “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images”. In: *IEEE Transactions on pattern analysis and machine intelligence* 6 (1984), pp. 721–741.

[25] Arthur P Dempster, Nan M Laird, and Donald B Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[26] Ernst Hellinger. “Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen.” In: *Journal für die reine und angewandte Mathematik* 136 (1909), pp. 210–271.

[27] Ce Zhang. “DeepDive: a data management system for automatic knowledge base construction”. PhD thesis. The University of Wisconsin-Madison.