Introductional Presentation

"Graph Kernels for RDF Data"*- Summary and implementation ideas

Dennis Oliver Kubitza

5. Dezember 2017

Introductional Presentation

"Graph Kernels for RDF Data"*- Summary and implementation ideas

Dennis Oliver Kubitza

5. Dezember 2017

Introductional Presentation

- **Graph kernels for RDF data.**
  Lösch, U., Bloehdorn, S., & Rettinger, A. (2012).
  The Semantic Web: Research and Applications, 134-148
  10.1007/978-3-642-30284-8_16
  2012

Content Overview

Content Overview

Combining RDF with Machine learing

Goal

We want to apply Machine learning algorithms on RDF.

Combining RDF with Machine learing

Goal

We want to apply Machine learning algorithms on RDF.

- Rescource Description Frameworks (RDF) impose a Graph structure on the Data

Combining RDF with Machine learing

Goal

We want to apply Machine learning algorithms on RDF.

- Rescource Description Frameworks (RDF) impose a Graph structure on the Data
- Machine learning algorithms are optimzed on certain Data Structures (Tables etc.)

Problem!

Combining RDF with Machine learing

Goal

We want to apply Machine learning algorithms on RDF.

- Rescource Description Frameworks (RDF) impose a Graph structure on the Data
- Machine learning algorithms are optimzed on certain Data Structures (Tables etc.)

Problem!

Solutions:

1. Rewrite the ML-Algorithms for RDF Data
2. Try to transform the RDF Data that it suits ML-Algorithms

Kernels mathematically

Definition

Let D be the Space of Data, $\psi : D \to F$ a mapping representing the Data as $F \subset \mathbb{R}^n$. A function K with a represenation

$$K : D \times D \to \mathbb{R}$$
$$K(d_1, d_2) = < \psi(d_1), \psi(d_2) >$$

is called Kernel.

Short version:

1. We take our Data d_1, d_2
2. We transfer them to real-valued vectors $\psi(d_1), \psi(d_2)$
3. We apply a scalar product.

Content Overview

Kernels for RDF

Let $G = (s, p, o)$ be our RDF.

Kernels for RDF

Let $G = (s, p, o)$ be our RDF.
What is D ?

Kernels for RDF

Let $G = (s, p, o)$ be our RDF.
What is D ?

- $D = \{G', G' \subset G\}$

What Kernels can we define for Graphs?

Kernels for RDF

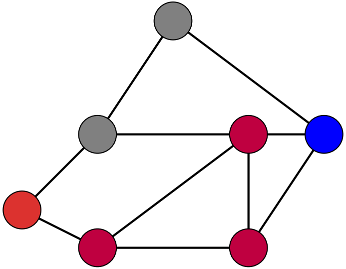
Let $G = (s, p, o)$ be our RDF.
What is D ?

- $D = \{G', G' \subset G\}$

What Kernels can we define for Graphs? Lösch et al. proposed for different suitable kernels.

1. Walk Kernel
2. Path Kernel
3. Full Subtree Kernel
4. Partial Subtree Kernel

Example: The path kernel



$$\kappa_{l,\lambda} = \sum_{i=1}^{l} \lambda^{i} |\{p | p \in paths_i(G_1 \cap G_2)\}|$$

Content Overview

1. RDF-Kernels
   - Motivation
   - Kernels for RDF
   - Relevance

Example

Why this approach ?

Example

Why this approach ? Lösch U., Bloehdorn, S., Rettinger, A. showed that this approach of applying Graph Kernels to RDF show almost the same peformance as retailoring ML algorithms to RDF structure.

Content Overview

Content Overview

Concept

- Build an Interface for the mangement of kernel information / values.

Concept

- Build an Interface for the mangement of kernel information / values.
  - Give the possibility to use User defined Update/Deletation algorithms and kernels.

Concept

- Build an Interface for the mangement of kernel information / values.
  - Give the possibility to use User defined Update/Deletation algorithms and kernels.
  - Provide the defined example Kernels with efficient implementation of Update/Deletation

Content Overview

2 Implementation
- General Concept
- Ideas for Implementation

Computation of Predefined Kernels

- Use Sansa's Graph Querying for Calculation of Trees, Paths.

Computation of Predefined Kernels

- Use Sansa's Graph Querying for Calculation of Trees, Paths.

+ Faster than Sparks RDD querying.

Computation of Predefined Kernels

- Use Sansa's Graph Querying for Calculation of Trees, Paths.

+ Faster than Sparks RDD querying.
+ Computational Performance will scale up with SANSA's performance.

Value Storage

Two ideas:

Value Storage

Two ideas:

1. Store Kernel-Values directly in original RDF
   - + All information is provied in Consistent Format.
   - + Induces a Distance Graph on Original RDF.
   - + Can be used for further analysis outside of Sansa

Value Storage

Two ideas:

1. Store Kernel-Values directly in original RDF
   - \+ All information is provied in Consistent Format.
   - \+ Induces a Distance Graph on Original RDF.
   - \+ Can be used for further analysis outside of Sansa
   - \- Do we really want to store everything permanently in the Dataset ?
   - \- ML-Algorithms need a lot of quering for extraction of Kernel Values.

Value Storage

Two ideas:

1. Store Kernel-Values directly in original RDF
   - \+ All information is provied in Consistent Format.
   - \+ Induces a Distance Graph on Original RDF.
   - \+ Can be used for further analysis outside of Sansa
   - \- Do we really want to store everything permanently in the Dataset ?
   - \- ML-Algorithms need a lot of quering for extraction of Kernel Values.
2. Store Kernel-Values in seperate Structure.
   - \+ No Changes on original RDF.
   - \+ Only relevant Information for ML/DM.

Value Storage

Two ideas:

1. Store Kernel-Values directly in original RDF
   - \+ All information is provied in Consistent Format.
   - \+ Induces a Distance Graph on Original RDF.
   - \+ Can be used for further analysis outside of Sansa
   - \- Do we really want to store everything permanently in the Dataset ?
   - \- ML-Algorithms need a lot of quering for extraction of Kernel Values.
2. Store Kernel-Values in seperate Structure.
   - \+ No Changes on original RDF.
   - \+ Only relevant Information for ML/DM.
   - \- What size ? (Has to be Scalable, e.g updates)
   - \- What structure ? (f.e Distance Matrix: Inefficient if Sparse)
   - \- Has to be a know format ! Otherwise not user friendly.

new Idea

- Use an RDF for storing the Kernel values.

new Idea

- Use an RDF for storing the Kernel values.
- Offer both possiblities:
  1. Store Kernels in original RDF
  2. Store Kernels in seperate reduced RDF (Original RDF without any Predicates and Attributes, except Kernel Oriented information)

new Idea

- Use an RDF for storing the Kernel values.
- Offer both possiblities:
  1. Store Kernels in original RDF
  2. Store Kernels in seperate reduced RDF (Original RDF without any Predicates and Attributes, except Kernel Oriented information)

+ Same implementation for both variants. Only different target for saving the values
+ User can pick which one he favours.
+ In each way the user is familiar with the storage format.
+ A posteriori the RDFs can still be merged.