HyperTwitter: Collaborative Knowledge Engineering via Twitter Messages

Martin Hepp

E-Business & Web Science Research , Group, Universität der Bundeswehr
Werner-Heisenberg-Weg 39 , D-85579 Neubiberg, Germany

mhepp@computer.org

E-Business and Web Science Research Group
Chair of General Management and E-Business

Univ.-Prof. Dr. Martin Hepp
Bundeswehr University Munich
Werner-Heisenberg-Weg 39
85577 Neubiberg
Germany

http://www.unibw.de/ebusiness/
ebusiness@unibw.de

Technical Report

TR-2010-01

Version: February 22, 2010

HyperTwitter: Collaborative Knowledge Engineering via Twitter Messages

Martin Hepp

Universität der Bundeswehr München, E-Business & Web Science Research Group
Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany
mhepp@computer.org

Abstract. A recent trend in the evolution of the Web is the massive contribution of small chunks of content by regular users, typically in combination with mechanisms for fostering social interaction. Such is often referred to as microblogging, with Twitter[1], Identi.ca[2], and Google Buzz[3] being the most widely known microblogging services. In this paper, we propose to use the underlying interaction pattern, and existing respective services, for the collaborative construction and maintenance of structured knowledge representations. We define (1) a syntax for embedding triple-like statements in Twitter messages, (2) develop a transformation into RDF, (3) suggest mechanisms for controlling the inclusion of such statements made by other users, (4) exploit the resulting graph for query expansion, and thereby provide a direct incentive for users to adopt our syntax, and (5) demonstrate the approach by means of a prototype implementation. The resulting RDF graphs can be combined easily with other Semantic Web data.

Keywords: Microblogging, Twitter, Extreme Tagging, RDF, Folksonomies, Tagging, Knowledge Acquisition

1 Overview

Microblogging services, in particular Twitter, Identi.ca, and Google Buzz have gained wide popularity. A 2009 Nielsen study [1], for example, reports an increase in Twitter user numbers from 475 k to over 7 Million between February 2008 and February 2009. In such services, contributions are limited to very short, plain text messages (140 characters in the case of Twitter), which typically forces users to take some cognitive effort for verbalizing thoughts or at least shortening existing pieces of content prior to publication. Mechanic copy-and-paste will typically not work due to the limited message length, or will at least require a very well-thought selection of the source fragment for the copy-and-paste operation. Thus, the majority of Twitter posts

[1] https://twitter.com/
[2] http://identi.ca/
[3] http://www.google.com/buzz

(called "tweets" or "status updates" in jargon) represent some human effort in processing information. Even the relaying of another user's message (called "retweeting") is based on human judgment and reflects a cognitive effort.

Twitter and most other microblogging services support users in filtering relevant content by a simple yet effective syntactical convention for *user identifiers* (@username) and *keywords* (#keyword, called "hashtag" in jargon). This allows spotting messages directed to a particular user or containing a particular keyword effectively. For instance, Twitter users can easily introduce multiple users to each other or point users who are monitoring a particular hashtag to a new Web resource:

```
@paulsmith : You should talk to @petermiller
#html5 developers: look at http://foo.com/
```

Based on simple string comparison techniques for such significant tokens, the service can link the millions of isolated short messages and build a densely meshed graph, representing social proximity and shared interests.

Unfortunately, Twitter hashtags and, to a lesser degree, Twitter user identifiers suffer from tag ambiguity (the same tag may stand for multiple meanings), tag heterogeneity (multiple tags are in use for the same meaning), and the lack of relationships between tags (e.g. super/subtag relations). The same problems are known from traditional social tagging systems; for an overview of those problems, see e.g. [7, pp. 74-76]. In Twitter, for example, it is very common that participants of an academic conference cannot immediately agree upon one authoritative hashtag for that event, which leads to disconnected messages about the same conference, because some posts contain the hashtag `#icwe10` and others contain `#icwe2010`.

The user community has only weak social instruments or techniques at hand for dealing with such synonymous tags. Very frequently, Twitter users spotting the use of synonymous tags will post messages like:

```
Please use #icwe instead of #icwe10 or #icwe2010
```

Such messages will be visible for any user watching any of the three variants and hopefully foster convergence. Also, we can often see that organizers of events try to stimulate consensus ex ante by publishing a hashtag recommendation.

In this paper, we will describe how a minimal extension of the existing Twitter syntax will allow Twitter users to

1. consolidate multiple synonymous hashtags for their future queries,
2. express hierarchical or other types of relationships between multiple tags,
3. introduce tags for types of properties betweet arbitrary resources, and
4. use popular Web vocabularies like FOAF, SIOC, Dublin Core, GoodRelations, and others inside Twitter messages.

The guiding principle is to provide a mechanism that is (1) immediately useful for the user contributing the additional content but is at the same time (2) suitable for sharing contributions along social networks, so that many people can benefit from it.

From such augmented tweets, we can easily construct an RDF graph that can be used to improve the recall of search operations on Twitter and that can be exported and combined with any other RDF data on the Web of Linked Data. Since all augmented statements remain regular Twitter messages, they can be shared with others via *Twitter lists* (grouping posts by a selected set of people) or *retweeting*, i.e. confirming and relaying a message to all individuals reading your own posts. The main idea is to provide a direct incentive for users to contribute useful statements in the extended syntax, which can be shared and used for weaving a Web of Linked Data.

The remainder of this paper is structured as follows: In section 2, we define a simple yet flexible syntax for embedding triple-like statements into Twitter messages, specify a mapping to RDF, and show how Twitter user identifiers and Twitter lists can be used to manage the inclusion of trustworthy statements. In section 3, we describe our implementation, which has been deployed to the Google Appengine cloud and is available at http://semantictwitter.appspot.com. In section 4, we evaluate our contribution. Section 5 summarizes related work, while section 6 discusses our contribution and highlights future extensions.

2 Collaborative Knowledge Engineering via Twitter Messages

In this section, we describe how a lightweight syntactical convention can support users of the Twitter microblogging service to (1) consolidate synonymous hashtags relevant to them and (2) author rich contributions for the Web of Linked Data.

2.1 Motivating Example

Very often, Twitter users cannot immediately agree upon a single authoritative hashtag for a topic, which makes it hard to spot all tweets related to that topic. Also, individuals and organizations often use multiple Twitter user IDs, which makes it hard to monitor all tweets from these accounts in one turn. Imagine that the hashtags `#munich` and `#muenchen` are in use for the German city of Munich, and the users `@mfhepp` and `@hypertw` relate to the same individual. While we could manually expand a query

```
#munich @mfhepp
```

to

```
#munich OR #muenchen @mfhepp OR @hypertw
```

we cannot model and thus reuse and share the underlying equivalency relationship.

Also, we cannot express more subtle relationships between tags, like the fact that one tag is more specific than another tag, nor model useful relationships between other resources. For example, we may want to state that two users know each other.

2.2 TripleTweets Syntax

With a lightweight syntactical convention based on the established Twitter syntax for tags ("#paris") and users ("@mfhepp"), we can empower Twitter users to embed machine-accessible statements into their tweets, which can then be used for query expansion and that can be combined with other RDF data sources. Basically, we (1) suggest to use "=" or "sameas" for expressing equivalence between tags or between user IDs, (2) "subtag" for expressing that one tag is more specific than a second one, (3) allow introducing arbitrary new properties between elements by means of a preceding greater sign, and (4) support popular CURIEs [2] (e.g. `foaf:knows`).

Our proposed syntax for triple-like statements inside Twitter messages ("trippletweets") is as follows:

```
tripletweet := { subject predicate object [ . tripletweet]}
subject := { @userid | #hashtag | http_uri }
predicate := { = | sameas | subtag | a | >property |
prefix:suffix }
object := { @userid | #hashtag | http_uri | "value" |
prefix:suffix }
userid := [-_a-zA-Z0-9\.]+
hashtag := [-_a-zA-Z0-9\.]+
http_uri := http://[-_a-zA-Z0-9\./?&%#]+
property := [-_a-zA-Z0-9]+
prefix := { foaf: | tag: | gr: | sioc: | rdfs: | rdf: | skos:
| owl: | dc: | dcterms: | rev: }
suffix := [-_a-zA-Z0-9]+
value := "[^"]+"
```

The elements subject, predicate, and object, as well as multiple tripletweets must be separated by one or more valid whitespace characters in the given encoding. The combination of prefix:suffix is a subset of all CURIEs [2].

2.3 Usage

In the following, we illustrate the use of our proposed syntax.

Simple examples

```
#iswc09 = #iswc2010
#newyork sameas #nyc
```

The hashtag #iswc09 is equivalent to #iswc2009. (tag:equivalentTag)

```
#iswc09 subtag #iswc
#tennis subtag #sports
```

The hashtag #iswc09 is a specialization of #iswc and the hashtag #tennis is a specialization of #sports.

```
@mfhepp = @martinhepp
```

The user @mfhepp is the same individual as the user @martinhepp. Note that the formal semantics of "=" and "sameas" depends on the type of the subject and object of the statement. For pairs of tags, it is `tag:equivalentTag`, for individuals it is `owl:sameAs`, and for http_uris is also `owl:sameAs`. For other pairs of entities, the statement is unsupported.

Using predefined vocabularies

```
@mfhepp foaf:knows @kidehen
@mfhepp foaf:name "Martin Hepp"
@mfhepp foaf:birthday "07-11"
```

The user @mfhepp knows the user @kidehen. The name of user @mfhepp is "Martin Hepp. His birthday is July 11.

```
#iswc09 skos:broader #iswc
```

The tag #iswc09 is related to the tag #iswc via the `skos:broader` property. This is equivalent to "#iswc09 subtag #iswc".

```
@microsoft a gr:BusinessEntity
@microsoft rdf:type gr:BusinessEntity
```

The user @microsoft is an instance of the class BusinessEntity in the GoodRelations ontology.

Introducing new tags for types of relationships

```
#munich >translation #muenchen
@mfhepp >dob "1971-07-11"
@mfhepp >hasname "Martin Hepp"
```

The hashtag #munich is related to the hashtag #muenchen via a property labeled with "translation". The user @mfhepp has a property labeled "dob" (date of birth) with the value "1971-07-11".

Using http URIs

```
@mfhepp >attends http://www.iswc2010.org/
http://iswc2010.org/ >successor http://iswc2009.org/
```

The user @mfhepp is related via a relationship labeled "attend" to something for which the Web page is http://www.iswc2010.org/. Note that the Web of Linked Data requires distinct URIs for events and for Web pages about events, so we cannot use the HTTP URI directly as the object of the statement but have to mint a new URI and link back to the original URI via `foaf:topic`. See section 3 for details. Also note that abbreviated URIs (bit.ly etc.) should be expanded prior to that, but are not at the moment.

Multiple statements in a single tweet

```
@mfhepp foaf:knows @kidehen . @mfhepp foaf:name "Martin
Hepp"
```

The user @mfhepp knows the user @kidehen and the real name of @mfhepp is “Martin Hepp”. Note that the whitespace is significant in here.

2.4 Representation in RDF

In the following, we describe how rich statements matching our syntax can be represented in the Resource Description Framework (RDF). This allows both the flexible implementation of query expansion (e.g. whether you just want to expand hashtags and user IDs by *equivalent* ones, or expand a query from a single user to *everybody from his social network.*)

As a basic pattern, we suggest to use a global namespace for the resulting conceptual element, i.e. each hashtag from Twitter will be assigned to the same URI in our RDF representation. Alternatively, one could assign individual namespaces for each individual user of a particular tag. The motivation for this decision is that in Twitter, hashtags have a stronger global reach than in other social tagging systems, because one does not invent tags for personal retrieval but for being visible by others.

Note that this does not mean than one could not control the subset of statements to be used for personal query expansion or other purposes, because the HyperTwitter approach defines a mechanism for defining the single user or group of users whose messages should be parsed and included in the RDF representation. See subsection 2.5 for more details.

The prefixes for users, tags, properties, and data elements are as follows:

```
props: <http://semantictwitter.appspot.com/id/properties/>
users: <http://semantictwitter.appspot.com/id/users/>
tags: <http://semantictwitter.appspot.com/id/tags/>
data: <http://semantictwitter.appspot.com/id/data/>
```

In addition, the following three Web ontologies will be used for the core RDF representation:

```
foaf: <http://xmlns.com/foaf/0.1/>
tag: <http://www.holygoat.co.uk/owl/redwood/tag/>
skos: <http://www.w3.org/2004/02/skos/core#>
```

Users

```
users:<userid> a foaf:Agent;
```

The preceding hash is not included in the URI.

Tags

```
tags:<hashtag> a tag:Tag;
               tag:name "<hashtag>".
```

HTTP URIs

```
data:<http_uri> foaf:topic <http_uri>.
```

All non-standard characters of string <http_uri> will be encoded before compiling the subject URI. For example, the URI http://purl.org/ will be represented as

```
data:http%3A%2F%2Fpurl.org%2F
```

It may be a good idea to first try to dereference the URI in order to expand abbreviated URIs (e.g. http://bit.ly/…), but this is currently not implemented.

Property tags

```
props:<property> a rdf:Property .
```

New properties do not immediately have any formal semantics beyond being `rdf:Property`, but can be found via SPARQL queries. Then, heuristics can be applied based on the frequency of usage, the types of values attached, and the lexical analysis of the property name.

CURIEs for Properties

For the supported prefixes, i.e. foaf, tag, gr, sioc, rdfs, rdf, owl, skos, dc, dcterms, and rev, the standard expansion to full URIs as per [2] is being used.

Predefined properties "=" and "sameas"

```
tags:<hashtag1> tag:equivalentTag tags:<hashtag2>
users:<user1> owl:sameAs users:<user2>
data:<http_uri1> owl:sameAs data:<http_uri2>
```

The formal semantics of "=" / "sameas" depends on the type of the subject and object of the statement. For pairs of tags, it is `tag:equivalentTag`, for individuals it is `owl:sameAs`, and for http_uris it is also `owl:sameAs`. For pairs of other types, such statements are ignored in the transformation.

Predefined property "subtag"

```
tags:<hashtag1> skos:broader tags:<hashtag2>
```

We use the SKOS [3] properties for hierarchy relations. Note that this is supported for pairs of tags only.

Predefined property "a"

```
users:<user1> rdf:type prefix:suffix.
data:<http_uri1> rdf:type prefix:suffix
```

The predefined property "a" is equivalent to `rdf:type` and can be meaningfully applied only to users and newly minted http URIs as the subject, and classes in the supported Web vocabularies, given as CURIEs.

In general, triples will be generated only if all constraints for that pattern match (e.g. `@user1 subtag @user2` or `#hashtag1 = @user1` will be ignored).

2.5 Trust and Filtering

It is important to be able to control the source of tweets to be included. For example, different users or group of users may disagree on whether particular hashtags are equivalent for them. We propose to use existing Twitter techniques for selecting subsets of tweets for a particular purpose: In the simplest form, a user will trust only his / her own tweets containing statements of equivalence or hierarchical relationships for query expansion. Alternatively, one can manage a specific list on Twitter that defines a set of users whose tweets should be considered for query expansion or other purposes. Such lists can be private or public.

Even if one decides to trust one's own statements only, it is possible to find and use other users' statements by simply retweeting them. So if a friend of yours makes the statement

```
<hashtag1> = <hashtag2>
```

and you find that useful, you can add it to your own query expansion and relay it to people following you in one turn by retweeting.

If you want to try a public account or Twitter list, take the following:

```
http://twitter.com/hypertw/
http://twitter.com/hypertw/trust
```

3 Implementation

In this section, we describe the HyperTwitter prototype, which is available on-line at http://semantictwitter.appspot.com/. The overall goal of the prototype is to provide a service that is immediately useful for each individual user, thus creating an incentive for adopting the proposed syntax. At the same time, the RDF content of all public Twitter messages is made accessible for further research and novel applications.
The prototype supports three types of functionality:
RDF/XML Export: First, the prototype can extract the Twitter messages for an arbitrary Twitter user or user-defined list and create an RDF/XML representation of the entailed statements following our extended syntax. The RDF/XML syntax uses very popular Web vocabularies like FOAF and SKOS whenever possible. If a particular user-defined list is set private or if the given user has protected his/her

tweets from the public, one can specify the password associated with that user ID to access the protected content. The RDF/XML method is also used by all other services.
Web Frontend: For testing the functionality and explaining the HyperTwitter syntax, there is a form-based Web frontend. Users can specify the user ID or user-defined list from which tweets should be considered for query expansion and enter a standard Twitter search. The prototype will analyze the query and expand all user IDs and hashtags to include equivalent hashtags or users and more specific hashtags.
Augmented REST-style API: The prototype supports extended variants of the three Twitter API services for searching twitter, which return an HTML, JSON, or Atom representation of the results for a given search. The extended API supports and carries along all native Twitter API parameters plus expects a user ID and optionally a list ID that determine the origin of "semantic" tweets to be considered for query expansion. If the list or user account is protected from the public, the respective Twitter password must be provided.

The application was written entirely in Python and deployed within the Google Appengine cloud. Internally, it uses only standard Python modules plus the rdflib library for handling RDF. The parsing of tweets is done mainly by applying regular expressions.

3.1 Web Interface

The Web interface takes an arbitrary query plus a user ID and optionally the ID of a Twitter list maintained by that user, and a query string. It then fetches the last 200 tweets from that user or user list, parses all HyperTwitter statements found, and constructs an RDF graph. Then, it analyzes the original query and expands every hashtag in the original query by the logical disjunction of all of its equivalent and more specific hashtags, and all user IDs by the logical disjunction of all equivalent user IDs. For example, the query for

```
"#munich @hypertw"
```

will be expanded to

```
"#munich OR #muenchen @hypertw OR @mfhepp"
```

for the user ID "@hypertw", because this contains respective statements. Note that there is no "OR" keyword in between the two sequences of keywords in order to preserve the original semantics of the query.

Quite clearly, the HyperTwitter application could quickly generate a large amount of traffic if it had to process and relay all *content* for all queries. To avoid that, HyperTwitter only computes the *expanded query* locally and then uses HTTP redirects (status codes 302) to point the requesting client to the original Twitter API or services, but with the query parameter replaced by the expanded variant. The Web interface is available at http://semantictwitter.appspot.com/.

3.2 RDF/XML Export

HTTP clients can request an RDF/XML serialization of the entire graph of HyperTwitter statements contained in the tweets from a given user ID or user-defined list. The service is available at http://semantictwitter.appspot.com/rdf and supports the following parameters:

u = ID of the Twitter user or list owner
list (optional) = ID of the list that contains the HyperTwitter statements to trust
p (optional) = password for user u, if list or user tweets are protected

Example: http://semantictwitter.appspot.com/rdf?u=hypertw&list=trust

The handler returns the RDF graph extracted from the given user ID or trust list. The media type is “application/rdf+xml”. Other serializations could be added quite easily.

3.3 Augmented REST API: Twitter Search With Query Expansion

There are three services that basically take the initial query plus the user ID and list name of the "trust" list and expand all hashtags and user IDs from the original query to all additional usernames and hashtags that are stated to be equivalent (`tag:equivalentTag` / `owl:sameAs`) or more specific, i.e. `skos:broader`. The three handlers differ only in the format of the returned results, which can be HTML, JSON, or Atom:

- http://semantictwitter.appspot.com/search
  This handler expands the twitter search API for HTML output.
- http://semantictwitter.appspot.com/search.atom
  This handler expands the twitter search API for Atom output.
- http://semantictwitter.appspot.com/search.json
  This handler expands the twitter search API for JSON output.

All three services require at least the following two parameters:
q = Twitter query string
u = Twitter user ID of the user or list owner
They also accept the following two additional parameters:
list = ID of the list that contains the HyperTwitter statements to trust
p = password for user u, if list or user tweets are protected
In addition, the handlers will pass along all regular Twitter search API parameters.

Examples:
http://semantictwitter.appspot.com/search?q=%23munich&u=hypertw&list=trust
http://semantictwitter.appspot.com/search.atom?q=%23munich&u=hypertw&list=trust
http://semantictwitter.appspot.com/search.json?q=%23munich&u=hypertw&list=trust

All three handlers carry out the query expansion and then return an HTTP 302 redirect response pointing the client to the original Twitter API with the new query

parameter. By this trick, we avoid relaying all search results via the Google Appengine. For example, the request
http://semantictwitter.appspot.com/search?q=%23munich&u=hypertw&list=trust
will return the following HTTP response, once the query expansion is completed:

```
HTTP/1.0 302 Moved Temporarily
Server: Development/1.0
Date: Tue, 16 Feb 2010 14:53:26 GMT
Content-Type: text/html; charset=utf-8
Cache-Control: no-cache
Location:
http://search.twitter.com/search?q=%23munich+OR+%23frankfurt
Expires: Fri, 01 Jan 1990 00:00:00 GMT
Content-Length: 0
Public: http://twitter.com/hypertw/trust
```

4 Evaluation

In the following, we present a preliminary evaluation of the technical feasibility of our approach. For testing purposes, we created the Twitter account "hypertw"

```
http://twitter.com/hypertw
```

and two lists managed by that user, named "trust"

```
http://twitter.com/hypertw/trust
```

and a protected one that requires authentication:

```
http://twitter.com/hypertw/trust-protected.
```

Using that account, we posted the following tweets:

```
@hypertw sameas @mfhepp
@hypertw >workson "Linked Data"
@mfhepp >dob "1971-01-01"
@mfhepp foaf:knows @hypertw
@hypertw foaf:name "Martin Hepp"
#eswc09 skos:broader #eswc
#iswc09 subtag #iswc
@mfhepp foaf:name "Martin Hepp"
#munich = #muenchen
```

Both twitter lists currently include just this one member but can be expanded easily. Theoretically, the Twitter API should return the very same set of tweets for the individual user and both lists, but we found that the lists are sometimes incomplete, likely due to limitations in Twitter's ability to update their complex indices in real-time.

4.1 RDF/XML Graph

Querying the HyperTwitter application for the RDF/XML graph for this set of tweets returns the expected result. Without any caching techniques and other optimizations, the RDF/XML serialization of the graph is being returned in between 500 and 1700 ms, which seems a fair latency. We assume that certain popular Twitter lists or user IDs with useful alignments will evolve, which would greatly benefit from HTTP caching approaches. In the following, we show the RDF representation gained from the tweets listed above. For reasons of readability, we show a Turtle syntax representation, derived mechanically from the actual result in RDF/XML.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix users: <http://semantictwitter.appspot.com/id/users/> .
@prefix tags: <http://semantictwitter.appspot.com/id/tags/> .
@prefix props:
<http://semantictwitter.appspot.com/id/properties/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix tag: <http://www.holygoat.co.uk/owl/redwood/tag/> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

users:hypertw  foaf:name "Martin Hepp" ;
               props:workson "Linked Data" ;
               owl:sameAs users:mfhepp ;
               rdf:type foaf:Agent .
users:mfhepp   props:dob "1971-01-01" ;
               foaf:knows users:hypertw ;
               foaf:name "Martin Hepp" ;
               rdf:type foaf:Agent .
tags:munich    tag:name "munich" ;
               tag:equivalentTag tags:muenchen ;
               rdf:type tag:Tag .
tags:iswc      tag:name "iswc" ;
               rdf:type tag:Tag .
tags:eswc09    tag:name "eswc09" ;
               skos:broader tags:eswc ;
               rdf:type tag:Tag .
tags:eswc      tag:name "eswc" ;
               rdf:type tag:Tag .
tags:iswc09    tag:name "iswc09" ;
               skos:broader tags:iswc ;
               rdf:type tag:Tag .
tags:muenchen  tag:name "muenchen" ;
               rdf:type tag:Tag .
props:dob      rdf:type rdf:Property .
props:workson  rdf:type rdf:Property .
```

We can see how the Twitter statements are properly translated into tag and user definitions and relationships between those, and that two new RDF properties "dob" and "workson" have been defined.

4.2 Query Expansion

Now, this graph can be used for straightforward query expansion. The current prototype expands just statements of tag and user equivalence and `skos:broader` relationships between tags, but any Web application is free to search for and exploit other useful types of relations between tags. For example, one could easily use the RDF/XML output to expand a query for a particular user to this user or anybody to whom he or she is linked via `foaf:knows`.

If we try the following URI,
http://semantictwitter.appspot.com/search?q=%23munich&u=hypertw
we will be properly redirected to
http://search.twitter.com/search?q=%23munich+OR+%23muenchen
We can see how the encoded query string

```
q=%23munich
```

has been substituted by

```
q=%23munich+OR+%23muenchen
```

based on the tweet

```
#munich = #muenchen.
```

4.3 SPARQL Access

The RDF/XML graph can of course be loaded into a repository of choice and then combined with other sources for SPARQL queries.

For example, we can use the public endpoint http://uriburner.com/sparql and force it to load our graph using the FROM keyword in SPARQL:

```
PREFIX tag:
<http://www.holygoat.co.uk/owl/redwood/tag/>
SELECT ?t
FROM <http://semantictwitter.appspot.com/rdf?u=hypertw>
WHERE {?t a tag:Tag}
```

This returns a list of all `tag:Tags` defined in the relevant set of tweets (only such that are part of HyperTwitter tweets are considered by our parser for the moment). We can easily combine that with other RDF representations of tag or tagging data.

Towards deriving ontologies from such HyperTwitter contributions, we can search for all novel properties introduced by users and rank them by frequency of usage. We could then define a threshold of popularity (e.g. say that we are interested in properties used only at least 100 times) and analyze the property names lexically or semantically. The following SPARQL query[4] will return all properties introduced in the tweets from the given user or list of users, and compute the frequency of usage for each property:

```
SELECT ?prop, count(?instance) as ?freq
FROM <http://semantictwitter.appspot.com/rdf?u=hypertw>
WHERE
    {?instance ?prop ?object.
    {SELECT ?prop WHERE
     { ?prop a rdf:Property. }
    }}
```

5 Related Work

HyperTwitter is related to and partly inspired by the following branches of work.

5.1 Meta-models of Tagging and Extensions

While tagging was introduced as an informal technique for attaching descriptors to resources in a collaborative setting, mainly to support the performance of retrieval, the huge amount of respective tagging data soon triggered interest in research to exploit this data for deriving more formal knowledge representations; for an overview of related questions, see *Gruber* [4]. A first step was the development of ontologies for sharing tags and tagging data, with *Newman's* **Tag Ontology** [5] being an early approach that has gained wide popularity. Other researchers proposed extended tag ontologies, mainly for facilitating access to the underlying social structures of tagging and for supporting the semantic enrichment of tags, e.g. by disambiguating homonymous tags. Two major efforts in this direction are **SCOT** [6] and **MOAT** [7]. For a comprehensive review of tag ontologies, see [8] and [9]. Our approach uses Newman's tag ontology for representing the tag entities found in HyperTwitter statements. This could be mapped easily to equivalent classes in additional tag ontologies.

The two most relevant works (and a direct inspiration) for creating the HyperTwitter prototype are **Extreme Tagging** [10] and **Tag4Tags** [11]. Both basically suggest to expand the domain of tagging activities from tagging a resource to tagging tags (in the case of Extreme Tagging), and other types of resources. Such can help to use tagging for consolidating personal or public tag usage and for authoring knowledge representations, since triples of tags can be understood as triples in the RDF model.

HyperTwitter applies the idea of Tags4Tags and Extreme Tagging to the significant tokens, e.g. user IDs and hashtags, in free-text microblogging.

[4] The nesting of queries is not supported by all RDF repositories on the market.

5.2 Syntactical Conventions for Embedding Semantics into Microblogging

Almost all microblogging services rely on simple syntactical conventions for marking up content in messages, e.g. using the hash sign as a prefix for tags/keywords and the "at" sign for user IDs. In the past three years, several proposals have been made to define additional syntactic conventions for representing richer structures.

On the high end of granularity is **MicroTurtle (µttl)** [12] by *Inkster*, a specification for embedding small RDF graphs into microblogging messages using the Turtle syntax. MicroTurtle is very similar to our approach. The main differences are that (1) we use a simpler, linear syntax that is closer to tagging than to RDF and has convenient shortcuts for tag consolidation, which is likely a key motivation for users to use rich structures in messages; (2) we did develop and deploy a reference implementation, and (3) we introduced a simple yet effective message for managing the inclusion of messages to the semantic representation by "trust" lists or user IDs.

TwitterData [13] is a proposal by *Fast* and *Kopsa* for encoding property-value pairs in Twitter messages, e.g.

```
San Francisco Airport $lat 37.612804 $long -122.381687
```

Other than HyperTwitter, it focuses on property-value pairs instead of triples. Also, a mapping to Semantic Web standards has been announced but is not yet available.

MicroSyntax [14] is a community effort to identify and document lightweight syntactical conventions for encoding information in Twitter messages and other short user contributions. The initiative also aims at supporting convergence among competing syntaxes and at creating reference implementations.

Picoformats [15] is an initiative led by *Messina* to define syntactical conventions for communications and for executing simple commands via short text messages, originally intended for command-line interfaces, SMS, and other devices. It also refers to other conventions, e.g. MicroTurtle or TwitterData.

Twitter Nanoformats [16] is another specification for embedding lightweight semantics into short messages. For example, it suggests the prefixes "L:" for locations, "event:" for events, and "time:" for temporal data.

Triple tags, also called "machine tags" on Flickr and on other services, are a convention for representing property-value pairs with explicit namespacing in short messages [17]. A popular usage is geo data, e.g. "geo:long=50.123456".

5.3 Semantic Microblogging

Very recently, there have been several proposals of lifting Twitter content to the Semantic Web technology stack in order to make it accessible for SPARQL queries that combine Twitter data with other RDF data on the Web. For example, **SemanticTweet** by *Flinter* is a straightforward service that automatically constructs a FOAF graph from a Twitter user's social network [18].

More sophisticated contributions are on one hand the work by *Nowack* [19] and on the other hand the **SMOB** (Semantic-MicrOBlogging) framework [20]. Nowack's work lifts Twitter content to RDF and makes it accessible to SPARQL. While the usage of machine tags (see above) is being discussed, it does not support the authoring of explicit triple statements in tweets, limiting the accessible content to user IDs, SIOC relations, URIs, and property-value pairs. We think it would be a very worthwhile extension to implement our HyperTwitter syntax in that prototype. SMOB allows exposing Twitter content in RDF in a similar way but also supports aligning tags or modeling relationships using existing vocabularies. The focus in SMOB is on exposing the obvious meta-data using standard vocabularies. A main difference of our approach is that we additionally foster the introduction of new tags for relationships so that the convenience of free tagging can be used for predicates as well.

5.4 Other works

Approaches of maintaining and consolidating tags have been discussed by several authors, e.g. by *Golov, Weller*, and *Peters* [21]. Their **TagCare** system allows users to collate their tags from multiple tagging systems and express semantic relations for future query expansion and other purposes.

Another stream of research aims at mining ontologies from tagging data, which can also be used for query expansion. A prominent example is the work by *Specia* and *Motta* [22]. They propose an automatic approach for deriving formal relations between tags from the combination of tagging data, existing Web ontologies, and other Web resources like Wikipedia. A major difference to our work is that the focus is on a fully automated extraction of formal representation, while we provide a syntax and application for the contribution of human judgment.

As far as Twitter query expansion is concerned, there are already first approaches, e.g. **TipTop** [23], a semantic Twitter-based search engine which seems to use mining and NLP techniques to extract relevant content for a given search from Twitter. However, it does not provide any RDF export of the data and can thus not be integrated with other Semantic Web resources or technology.

6 Discussion and Conclusion

At the time of writing, the amount of Twitter messages posted reaches 50 million tweets per day, which is an average of 600 tweets per second [24]. That means that users contribute an unprecedented amount of content, time, and intelligence, that may be very rewarding to tap for weaving a dense and current Web of Linked Data. Both for maintaining ontologies and facts in knowledge bases, the delayed inclusion of user feedback has kept on being a major bottleneck towards powerful intelligent knowledge-based systems; for a discussion, see e.g. [25].

In this paper, we described five main contributions: (1) A syntax for embedding triple-like statements in Twitter messages, (2) a mapping into RDF, (3) mechanisms for controlling the inclusion of statements made by other users, (4) a technique for exploiting the resulting graph for query expansion, which creates a direct incentive for users to adopt our syntax, and (5) a prototype that demonstrates our approach. We have shown how the RDF graphs resulting from typical messages in the expanded syntax can be used for query expansion and how they can be combined with other Semantic Web data.

Our approach reuses the ideas of Extreme Tagging [10] and Tags4Tags [11], i.e. using free tagging for modeling new types of relationships, for the challenge of knowledge authoring by means of microblogging.

While the implementation already supports the full Twitter search API, it is subject to a few limitations, which we plan to overcome shortly. First, the application fetches the latest 200 tweets for a given user or list only. This could be expanded easily but bears the risk of consuming too much CPU time and bandwidth on the Google Appengine. It may turn out that this limitation is indeed a feature, because older tweets could be less relevant, e.g. by consolidating tags that have become out of interest to us in the meantime.

Second, the application supports only HTTP Basic Authentication, which is subject to vulnerabilities, in particular if used with weak passwords or via insecure channels. Third, we do not yet use caching techniques (e.g. Google MEMCACHE), which could turn out very valuable if a few popular Twitter lists with popular tag alignments evolve. If added, such could be combined with an internal archive of older messages.

Fourth, by intention, the prototype uses one global namespace for all tags in Twitter. This increases access to tags but excludes tag disambiguation. Our motivation for that design choice is that in Twitter, other than in typical tagging systems, tags are first and most used as tokens to receive the attention of others, i.e., they are designed to be global. Of course, multiple users or communities may produce "tag collisions" by that, but there is a strong incentive to use globally valid hashtags.

Fifths, the potentially very valid mechanism of approving HyperTwitter statements by retweeting is currently supported for Twitter lists only, not for individual users. This will be added shortly, for we think that this will be very powerful for speeding up the dissemination of useful statements.

Acknowledgments: HyperTwitter is inspired by the Tags4Tags approach as described by Leyla Jael García-Castro et al. [11].

7 References

1. NielsenWire: Twitter's Tweet Smell Of Success, available at http://blog.nielsen.com/nielsenwire/online_mobile/twitters-tweet-smell-of-success/.
2. CURIE Syntax 1.0. A syntax for expressing Compact URIs. W3C Candidate Recommendation 16 January 2009, available at http://www.w3.org/TR/curie.
3. SKOS Simple Knowledge Organisation System Reference. W3C Recommendation 18 August 2009, available at http://www.w3.org/TR/skos-reference.
4. Gruber, T.R.: Ontology of Folksonomy: A Mash-Up of Apples and Oranges. International Journal on Semantic Web and Information Systems (IJSWIS) **3** (2007) 1-11
5. Tag Ontology: An Ontology for Tags, available at http://www.holygoat.co.uk/owl/redwood/0.1/tags/.
6. SCOT: Let's Share Tags!, available at http://scot-project.org/.

7. Passant, A., Laublet, P., Breslin, J.G., Decker, S.: A URI is Worth a Thousand Tags: From Tagging to Linked Data with MOAT. International Journal on Semantic Web and Information Systems (IJSWIS) **5** (2009) 72-94
8. Kim, H.L., Passant, A., Breslin, J.G., Scerri, S., Decker, S.: Review and Alignment of Tag Ontologies for Semantically-Linked Data in Collaborative Tagging Spaces. 2008 IEEE International Conference on Semantic Computing (ICSC2008). IEEE Computer Society Washington, DC, USA, Santa Clara, CA, USA (2008) 315-322
9. Kim, H.L., Scerri, S., Breslin, J.G., Decker, S., Kim, H.-G.: The State of the Art in Tag Ontologies: A Semantic Model for Tagging and Folksonomies. International Conference on Dublin Core and Metadata Applications, Berlin, Germany (2008) 128-137
10. Tanasescu, V., Streibel, O.: Extreme Tagging: Emergent Semantics through the Tagging of Tags. First International Workshop on Emergent Semantics and Ontology Evolution (ESOE 2007), Vol. 292. CEUR-WS.org, Busan, Korea (2007) 84-94
11. García-Castro, L., Hepp, M., García, A.: Tags4Tags: Using Tagging to Consolidate Tags. 20th International Conference on Database and Expert Systems Applications (DEXA 2009), Vol. LNCS 5690. Springer, Linz, Austria (2009) 619-628
12. MicroTurtle (µttl), available at http://buzzword.org.uk/2009/microturtle/spec.
13. Twitter Data. A Simple, Open Proposal for Embedding Data in Twitter Messages, available at http://twitterdata.org/.
14. Microsyntax.org: Deep Structure of the Real-time Stream, available at http://www.microsyntax.org/about-microsyntax/.
15. Picoformats, available at http://microformats.org/wiki/picoformats.
16. Microblogging Nanoformats: Twitter (or Jaiku) Nanoformats Proposal, available at http://microformats.org/wiki/microblogging-nanoformats.
17. Tag (metadata), section: Triple Tags, available at http://en.wikipedia.org/wiki/Triple_tag#Triple_tags.
18. SemanticTweet. Twitter Meets the Semantic Web, available at http://semantictweet.com/.
19. Turn Twitter Into Your Personal Assistant, available at http://www.devx.com/semantic/Article/40869.
20. SMOB: Semantic-MicrOBlogging Framework, available at http://smob.me/.
21. Golov , E., Weller, K., Peters, I.: TagCare: A Personal Portable Tag Repository.: Proceedings of the Poster and Demonstration Session at the 7th International Semantic Web Conference (ISWC2008), Vol. 401. CEUR-WS.org, Karlsruhe (2008)
22. Specia, L., Motta, E.: Integrating Folksonomies with the Semantic Web. In: Franconi, E., Kifer, M., May, W. (eds.): 4th European Semantic Web Conference (ESWC 2007), Vol. LNCS 4519. Springer, Innsbruck, Austria (2007) 624-639
23. TipTop Search Engine, available at http://feeltiptop.com/.
24. Measuring Tweets. Twitter Blog Post, February 22, 2010., available at http://blog.twitter.com/2010/02/measuring-tweets.html.
25. Hepp, M.: Possible Ontologies: How Reality Constrains the Development of Relevant Ontologies. IEEE Internet Computing **11** (2007) 90-96