GATECloud.net: Web-Scale Semantic Annotation and Search Made Easy

Valentin Tablan and Kalina Bontcheva and Ian Roberts and Hamish Cunningham

Department of Computer Science, University of Sheffield
S1 4DP, Sheffield, UK

Abstract. The growth of social media and other unstructured content on one hand, and Linked Open Data on the other, now pose a significant challenge for Semantic Web researchers wishing to design, implement, and evaluate semantic annotation algorithms on web scale datasets. Running experiments on standard servers is often too time consuming, whereas implementing these efficiently with MapReduce/Hadoop requires significant engineering skills.

This paper presents a cloud-based semantic annotation and search infrastructure, which enables researchers to configure and run easily semantic annotation pipelines, while the infrastructure itself takes care of the distribution problem. The infrastructure also supports shared/crowdsourced gold-standard creation, performance evaluation, efficient text and annotation indexing, and semantic search. The paper also evaluates the performance gains achieved by using the cloud-based infrastructure and presents a cost/benefit analysis against existing similar services.

1 Introduction

The semantic annotation of web and intranet content is a research problem which has been receiving significant interest over the past ten years. With the growth of textual content on the web have come the challenges of scalability and running efficient experiments for large-volume semantic annotation. There are still many outstanding technological challenges, especially the problem of delivering semantic annotation using the Software-as-a-Service model. Unfortunately content analysis services are currently problematic for both suppliers and customers, for two reasons. Firstly, service creation can have high initial and ongoing infrastructural costs, which are only affordable to very few, large companies as a result. Secondly, existing semantic annotation services mostly focus on English and are not easily customisable by their users. All processed content is also made accessible to the provider, which is not always appropriate due to confidentiality. While there are other proven semantic annotation tools, which are open and easily customisable, they are not yet available as a scalable cloud-based infrastructure.

From an infrastructural perspective, semantic annotation computation is well suited to the cloud, since it is very often bursty, and therefore ill-suited to fixed
In addition to lowering the barrier to entry, GATECloud.net also reduces the time required to carry out large-scale semantic annotation experiments by allowing researchers to harness on-demand compute power on the cloud. As argued by Barga et al [1], cloud-based research infrastructures are beneficial in general, since they offer the potential to accelerate the pace of engineering and scientific discovery. In other words, the easy access to data and computation at a large scale will make it easier for researchers from smaller organisations to carry out novel research on large-scale data and to participate in collaborative research communities. In particular, Foster [2] argues that cloud-based infrastructures can contribute not just through data sharing and on-demand compute power, but also by lowering the complexity of managing the end-to-end research process.

This paper presents a cloud-based semantic annotation and search infrastructure (GATECloud.net), which enables researchers to configure and run easily semantic annotation pipelines, while the infrastructure itself takes care of the distribution problem. The infrastructure also supports shared/crowd-sourced gold-standard creation, performance evaluation, efficient text and annotation indexing, and semantic search. The paper also evaluates the performance gains achieved by using the cloud-based infrastructure and presents a cost/benefit analysis against existing similar services.

2 Related Work

Services such as OpenCalais1, Extractiv2, Alchemy API3, and Lupedia4 all provide semantic annotation of text, which is accessible programmatically. However, the semantic annotation application is pre-packaged, i.e., it is not easy for the user to extend the functionality. In addition, many of these come with a daily limit on the number of API calls that can be made or documents that can be processed, even when monthly subscription is paid for. Many also have limits on document size. In practical terms, if one needs to process tweets (millions of very small documents), patent documents (ranging between 3kB and 6MB), or millions of news articles, processing them one after another, using a remote API, is extremely inefficient. It can also be very expensive in the case of tweets, since each tweet is very short, but will count as an individual API transaction.

GATECloud.net is innovative and different in that it is currently the only open-source cloud-based semantic annotation platform. Similar to the services above, it offers a growing number of pre-packaged semantic annotation services. However, uniquely GATECloud.net also supports a bring-your-own-annotator option, which can be built easily by reusing pre-existing components and adding some bespoke ones. Even more crucially, GATECloud.net is the only end-to-end full lifecycle solution. It doesn’t just do entity extraction, but also data

---

1 http://www.opencalais.com
2 http://extractiv.com
3 http://www.alchemyapi.com
4 http://lupedia.ontotext.com
preparation (e.g., HTML content extraction), manual corpus annotation, interannotator agreement, quality assurance and control, data visualisation, indexing and search of full text, annotations, and ontological knowledge. GATECloud.net also offers a web-based collaborative annotation tool to correct mistakes and create training and evaluation data (see Section 6).

Some other differences are that GATECloud.net supports bulk upload of compressed documents to process and there is no need to use programming APIs (everything is accessible via a web interface). Crucially for researchers who only need large-scale semantic annotation infrequently, there are no recurring monthly costs: instead, GATECloud.net is pay-per-use, billed per hour (since Amazon bills cloud machine usage on a per hour basis). There is also no daily limit on number of documents to process and neither there is one on document size. Consequently, costs of processing dependent on overall data size, not the number of documents to be processed. This is particularly advantageous for Twitter processing. The only downside of our per-hour billing approach is that it makes it somewhat harder for users to estimate costs upfront. We return to this issue in Section 8, where we discuss how users can overcome this problem. Next, we introduce the GATECloud.net platform in more detail.

![Fig. 1. Reserving a Mimir or Teamware Server Workflow](image)

3 GateCloud.net: An Overview of the Infrastructure

GATECloud.net is a new cloud-based infrastructure that deploys GATE semantic annotation and search tools on cloud computing infrastructure. The intention behind it is to bring the advantages typically associated with cloud computing within the reach of semantic web researchers and companies wishing to experiment with and develop large-scale semantic annotation projects.

At present, there are two types of functionality offered:

- on-demand servers running as Amazon EC2 instances, based on 64 bits Ubuntu Linux, with pre-installed with specialised software: Mimir semantic search (see Section 5) and Teamware for collaborative corpus annotation (see Section 6)

- execution of automatic semantic annotation pipelines over large document collections, using the massive parallel, on-demand compute infrastructure offered by the cloud.
The process of reserving a new server is illustrated in Figure 1 and comprises four simple steps: the user requests a new reservation through the GATECloud.net web site (1), a persistent data volume is created and associated with the user’s account (2), a persistent server name is reserved for the user (3), and the user is notified of success or otherwise (4).

Once reserved, a server can be started using the workflow in Figure 2, where following the user’s start request (1), a new instance is started (2), the previously created data volume is attached to the new instance (3), the reserved server name is associated with the IP address of the new instance (4), and, as soon as the start-up process is confirmed to have completed successfully (5), the user is notified by email (6). While the server is running it can be accessed through its web interface, which includes support administrative actions, such as adding user accounts and setting up security options.

The other half of the GATECloud.net infrastructure is the support for on-demand processing of large document collections using cloud computing. Running large-scale semantic annotation jobs is a task that is far from trivial. The most obvious barrier is the amount of compute power required, or, alternatively, a very large amount of time. Less obvious, but equally challenging, are tasks such as controlling the workflow that distributes the workload across several computers, dealing with errors, recovering from exceptional conditions, making the most of the available CPU power, and dealing with various input and output formats. Another difficulty is supplying sufficient storage for the input document collection, the output files, and all the temporary files – which are short-lived but can be orders of magnitude larger.

The support for Annotation Jobs on GATECloud.net aims to address most of these engineering issues, leaving researchers free to concentrate on their experiments. The steps for semantic processing on the cloud are shown in Figure 3, where three parallel workflows are present. Activities that result from interaction with the user are on the left side of the picture, using a yellow background. The...
whole process starts with the user uploading the annotation pipeline and the input document collection to the storage server (1), followed by them starting the annotation job (2). This causes the GATECloud.net system to create a workflow for the new job (3), consisting of tasks that get queued for execution. As tasks are consumed, new ones are created as defined by the workflow, until all the tasks for the given job are completed. When that happens, the user is notified (4) and they can download the execution reports and annotated documents from the storage server (5).

In parallel, the GATECloud system manages groups of server nodes (referred to as swarms) whose job is to execute tasks from the task list. The system makes sure that the required swarm is active whenever there are tasks queued for it.

On the side of the swarms, the workflow (represented by the steps on the right side of the image, using a red background) consists of picking up tasks from the appropriate queue, executing them, saving the results produced (if any), and updating the task state in the list with the corresponding exit code.

**Security and privacy** GATECloud.net does not run a separate machine for each annotation job. Instead it splits each annotation job up into manageable pieces (referred to as tasks), puts these tasks into a queue, and runs a collection of processing machines (referred to as "nodes") that simply take the next task from the queue whenever they have finished processing their previous task. While a task is running it has exclusive use of that particular node - we never run more than one task on the same node at the same time - but once the task is complete the same node will then run another task (which may or may not be part of the same annotation job).

To ensure the security and privacy of code and data, the node takes the following precautions. Firstly, all processes are run as an unprivileged user account which only has write permission in a restricted area of the filesystem. Secondly,
at the end of every task, all processes running under that user ID are forcibly terminated (so there is no risk of a stray or malicious background process started by a previous task being able to read new data). And lastly, the application and data directories are completely deleted at the end of every task (whether the task completed successfully or failed).

4 Deploying Semantic Annotation Pipelines

GATECloud.net can run any semantic annotation pipeline packaged by the GATE developer environment [3] using the application export option “Export for GATECloud.net”. This builds a self-contained zip file, including all semantic annotation modules and the linguistic data and ontologies required by them. In other words, there is no additional development effort required, in order to deploy a GATE-based semantic annotation pipeline on the cloud and execute it on a large-scale dataset.

The development lifecycle consists of the following steps:

1. Create a semantic annotation pipeline on a small document collection, within the GATE development environment, running on a standard desktop or local server machine;
2. Create a gold-standard corpus for evaluation and/or training via Teamware on the cloud;
3. Evaluate the performance of the automatic pipeline on the gold standard using GATE Developer locally or Teamware on the cloud. Return to step 1 for further development and evaluation cycles, as required.
4. Upload the large datasets and the semantic annotator to GATECloud.net;
5. Execute them, injecting the output into Mimir for semantic search and/or download the result files in XML, RDFa, or schema.org formats.
6. Analyse any errors, optionally annotate more gold-standard data with Teamware and evaluate again on unseen, randomly chosen documents. If required, iterate again over system development steps on local machines.

The GATE framework [3] underpinning the semantic annotation pipelines for GATECloud.net, provides reusable implementations of semantic annotation components and a set of prefabricated software building blocks that researchers can use, extend and customise for their specific needs. Figure 4 shows the main semantic annotation components which can be reused and extended when building new applications.

Firstly, GATE supports importing, accessing and visualising RDF and OWL ontologies, as well as using those as lexical and knowledge resources within semantic annotation systems. Since the emphasis is on document annotation, rather than ontology authoring, only basic ontology editing capabilities are provided and the assumption is that pre-existing ontologies will be used, e.g., Linked Open Data ontologies. Nevertheless, application-specific extensions with new classes and instances are possible from within GATE.
The most reused and extended components for semantic annotation are the automatic ones, especially the ontology-based gazetteers, the JAPE pattern matching engine, and the ontology-aware evaluation facilities.

A gazetteer typically contains names of entities/instances such as cities, organisations, days of the week, etc. The word gazetteer is often used interchangeably for both the set of resources that contain the names and for the algorithm that makes use of those lists to find occurrences of these names in documents. There are two ontology-based gazetteers, which are useful for building large-scale annotation pipelines.

The OntoRoot gazetteer analyses the ontology, i.e., all classes, instances, and properties, to derive a list of lexicalisations (e.g., IBM, Big Blue) and their corresponding URIs (i.e., the URI of the IBM instance). In addition, OntoRoot captures morphological variations, e.g., the string "language resources" in the document would be matched against a class with label "language resource". A potential limitation of OntoRoot is that it builds the lexical resources from the ontology only once on initialisation, which means that any run-time updates to the ontology are not taken into account as soon as they appear.

A much more scalable gazetteer is the Large Knowledge Base (LKB) Gazetteer, which we have used successfully with many Linked Open Data sets. Any RDF-based ontology can be loaded and then the gazetteer produces lookup annotations that have both instance and class URIs. The LKB gazetteer builds a
dictionary from the ontology, based on a SPARQL or SERQL query. Any query is supported, as long as its projection contains at least two columns in the following order: label and instance. As an option, the query may have a third column for the ontology class of the RDF entity.

The Semantic Enrichment module allows adding new data to semantic annotations by querying external RDF (Linked Data) repositories. It complements the ontology-based gazetteers and showcases the usefulness of using Linked Data URI as identifiers. This component runs a SPARQL query against a given repository and puts a comma-separated list of the values mentioned in the query output in a connections feature of the semantic annotations.

The JAPE pattern matching engine uses rules which describe patterns to be matched (left-hand side) and annotations to be created (right-hand side). It provides access to ontologies on the right-hand side of JAPE rules, which allows rules to add new information to the ontology (e.g., add an instance or a newly discovered property value) or to use reasoning (e.g., to obtain semantic distance between concepts). The ontology and most notably the subsumption relation is also taken into account when matching on the left-hand side. So for example, a rule might look for an organization followed by a location, in order to create a locatedAt relationship between them. By using subsumption, the rule automatically matches not just organisations, but also all of its sub-classes in the ontology, e.g., Company, GovernmentOrg.

Another key part of the development of semantic annotation systems is quantitative evaluation. Since semantic annotation identifies mentions of instances from a given ontology, there are cases when a system would identify an instance successfully but does not assign it the correct class. Since the assigned class does not match the correct class according to the manually annotated data, traditional precision would regard it as wrong. However, due to the closeness of the two classes in the ontology, the system should be given some credit. For such cases, GATE offers BDM (a Balanced Distance Metric), which measures the closeness of two concepts in an ontology or taxonomy [4].

5 Mímir: Scalable Semantic Search on the Cloud

Hybrid search tools effectively combine keywords and semantic search (e.g. [5, 6]), but do not have the functionality required for annotation-based search and do not enable the kind of complex queries which motivated us to develop Mímir. For example, they cannot reliably return sentences in a particular section of a document from a particular domain where a cabinet minster talks about a London hospital spending between 1 and 10 million pounds at a date between July 2009 and January 2010. Furthermore, these tools have a slightly different goal in terms of information finding: they are designed to return a ranked list of documents matching the search terms, whereas what is required in annotation-based search is to return every instance of a query match in the document collection.
Mímir\(^5\) is a multi-paradigm information management index and repository which can be used to index and search over text, annotations, semantic schemas (ontologies), and semantic meta-data (instance data). It allows queries that arbitrarily mix full-text, structural, linguistic and semantic queries and that can scale to terabytes of text.

Support for full-text search represents the most basic indexing functionality and it is required in most (if not all) cases. Even when semantic annotation is used to abstract away from the actual textual data, the original content still needs to be accessible so that it can be used to provide textual query fragments in the case of more complex conceptual queries. Mímir uses inverted indexes for indexing the document content (including additional linguistic information, such as part-of-speech or morphological roots), and for associating instance of annotations with the position in the input text where they occur. The inverted index implementation used by Mímir is based on sequence operators in MG4J\(^7\) which proved to be sufficiently efficient for our purposes.

Semantic annotation tools produce some form of meta-data associated to text snippets in the documents, which is typically graph structured, due to the pointers from one annotation to another (e.g., event annotations typically point to the entity annotations, involved in the event).

Mímir’s view of annotations is based on that of GATE\(^3\), with each annotation described by: (i) the document it belongs to; (ii) the start and end offset of the referred text snippet; (iii) the annotation type; (iv) an arbitrary set of \(<\text{feature},\text{value}>\) pairs, encoding further semantic information.

An annotation index supports a more generic search paradigm. Depending on the type of annotations available, the user can search across different dimensions. If, for example, the documents are annotated with occurrences of \texttt{Person}, \texttt{Location}, \texttt{Organization} entities, then searches like \{\texttt{Person}\}, CEO of \{\texttt{Organization}\}, based in \{\texttt{Location}\} become possible.

Efficient indexing and search over annotation graphs, is a challenging problem, which cannot be addressed with existing off-the-shelf software. XML-based search techniques tend to focus on tree-like data, which makes them difficult to adapt to the graph structures in which we store annotation data, whereas DBMS techniques are similarly oriented towards relational models and need to be adapted. The efficient indexing of annotation data in Mímir is handled via two plugins. The first one stores annotation data in a specifically tailored relational database. The other one models semantic annotations as ontological objects in order to support richer semantic querying. OWLIM\(^6\) is used as the underlying knowledge repository, due to its scalability (currently up to 1 trillion unique entities can be stored) and linear parallel query performance. The links between annotations, the textual data, and the knowledge base information are created by the inclusion into the text indexes of a set specially-created URIs that are associated with annotation data. Furthermore, URIs of entities from the Knowledge Base can be stored as annotation features. We have successfully

\(^5\) Old Norse “The rememberer, the wise one”.
\(^6\) http://www.ontotext.com/owlim/whats-new
experimented both with using custom-built ontologies (e.g., for patent processing (see Section 7) as well as reusing popular Linked Open Data (e.g., DBPedia). Additional scalability in Mímir is available through support for federated indexes, which present a group of indexes as a single virtual index.

The main benefit of deploying Mímir on the GATECloud.net infrastructure is that very large semantic annotation indexes can be built extremely quickly and without any knowledge of the underlying infrastructure, simply by specifying the URL of a cloud-based Mímir instance as an output target of a GateCloud.net annotation job. In addition, the typical software-as-a-service advantages apply, i.e., no installation and configuration overheads, as well as making it easy for users who do not have suitably large-scale server hardware to carry out semantic search experiments on web-scale data, at affordable prices.

Mímir also supports the option of building the semantic index on the cloud and then downloading and importing it into a local Mímir server, which minimises the costs of carrying out searches against the cloud-based server, while at the same time, making the indexing step highly efficient and straightforward.

Since all searches are local to a document, we can use clusters of semantic repositories, which has several advantages:

- Faster indexing: less data in each repository;
- Faster search: search space is broken into slices that are searched in parallel.
- Joining results is trivial: union of result sets.
- Simple scalability: through adding more nodes.
- Search speed stays almost constant while the data increases: each individual repository has the same amount of data; there are just more repositories.

From a user’s point of view, Mímir is an interface for searching over semantically annotated documents. It supports not only standard full-text search, but also search by part-of-speech and/or morphological roots. Beside searching the document text, Mímir also supports searches over the documents’ semantic annotations, where queries are based on annotation types and restrictions over the values of annotation features. These different search paradigms can be combined freely into complex queries, with support for sequences, repetitions, and Boolean operators. Date restrictions are also supported, e.g., restricting the search to tweets only from the past 3 months. For query examples, see Section 7.

6 Teamware: Collaborative Creation of Training and Evaluation Data

Manual semantic annotation editors (e.g., OntoMat Annotatizer [8], Annotea [9], GATE [10]) are typically used in semantic annotation projects, in order to create gold-standard human annotated datasets. However, apart from editing tools, much more is required in order to produce high quality annotated corpora: a stringent methodology, annotation guidelines, inter-annotator agreement measures, and in some cases, annotation adjudication (or data curation) to reconcile differences between annotators.
In order to address these issues in a comprehensive framework, we have developed GATE Teamware [11]: a web-based software suite and a methodology for the implementation and support of complex semantic annotation projects. In addition to its research uses, it has also been tested as a framework for cost-effective commercial annotation services, supplied either as in-house units or as outsourced specialist activities.

From an implementational perspective, GATE Teamware uses a 3-tier service-oriented architecture for distributed collaborative annotation and is driven by a centralised workflow engine. The architecture is appropriate to the task and performs effectively, but deployment and administration of the tool is complex and error-prone. In order to address this problem, we have developed a cloud-based deployment where a standard virtual machine image is supplied. Instead of needing skilled administrator time, the process is fully automatic and results in a running GATE Teamware instance that can be turned on and off as required.

In addition, the GATECloud.net integration makes it straightforward to select a sample from all automatically annotated documents on the cloud and channel these into a cloud-based Teamware instance, where human annotators correct them in order to create a gold standard corpus.

Since typically, manual corpus annotation is an activity undertaken for relatively short periods of time (days, weeks and rarely one or two months) by teams of people, our SaaS approach cuts costs and offers flexibility with respect to intermittent usage.

7 Three Case Studies: Patent Processing, News and Twitter Analysis

In order to quantify the performance gains from running semantic annotation experiments on large datasets on GATECloud.net, we carried out experiments on three kinds of document collections: 50 million tweets (many, very short text), 20,000\textsuperscript{7} news articles (medium-sized texts, targeted by existing semantic annotation services), and 100,000 patents (some large documents of up to 6MB).

The news and Twitter datasets were annotated for named entities with the standard GATE components, whereas for the patents dataset we re-used a pre-existing semantic annotation pipeline [12] which recognises patent-specific types from an ontology: references to other patents or scientific publications, measurements, patent sections, claims, etc. In addition to semantic annotation, the application uses an ontology to carry out data normalisation. Taking an example from references to figures and claims, expressions such as “Figures 1-3” and “Claims 5-10” imply references not just to the explicitly mentioned figure/claim numbers but also to all those in between. Measurements are an even more com-

\textsuperscript{7} We are preparing experiment results on a much larger news collection, which will be reported instead should the final paper be accepted. The current dataset however is nevertheless sufficient to validate the conclusions made here, since performance is very close to linear.
plex case, where the ontology and reasoning are used to match values such as 2 inches or 135mm to a query for measurements of between 2.5 and 15cm.

The semantic annotation output was automatically indexed in Mimir and here we exemplify some semantic queries over the results. Figure 5 shows two example queries and some of the matching annotations in the patents dataset. The first query is annotations based and matches references to patents or publications, which occur either in the Prior Art or in the Background Art sections:

\[ \{\text{Reference type="Literature"} \} \text{ OR } \{\text{Reference type="Patent"} \} \text{ IN } \{\text{Section type="PriorArt"} \} \text{ OR } \{\text{Section type="BackgroundArt"} \} \]

The second query is for mentions of speeds in the patents, which are between 1 and 100 m/s. In this case, underlying knowledge from the ontology is used in order to match speeds such as 235 feet/sec or 8 mph. In this case, the external knowledge is used implicitly by Mimir, during search.

Another example, this time on the news dataset is of a Mimir query which contains embedded SPARQL. In this way, additional knowledge from the ontology is used during matching, even though it is not present in the text. In this case, we are using DBPedia [13] as the underlying knowledge base.

8 Cost-Benefit Evaluation

In order to carry out a cost-benefit evaluation of GATECloud.net, we ran three sets of experiments, processing the above mentioned patents, news, and Twitter datasets. Each dataset was processed on the following machines:
Fig. 6. Example Mimir semantic query

- a desktop: Intel Xeon E5502 1.86GHz, 2 CPU cores, 4GB RAM, 320 GB HDD Serial ATA II, cost 1300. This is a fairly standard desktop configuration for researchers in our group, but PhD students tend to have less powerful machines, which would result in further increases in processing time.

- a modern server: HP ProLiant DL385 G7, AMD Opteron 2.3 GHz, 12 CPU cores, 32 GB RAM, 2 TB disk space, approximate cost 4800. For these experiments, we restricted the processing to using 6 cores only.

- GATECloud.net running on 10 Extra Large Instances on the Amazon Cloud. 15 GB of memory, 8 EC2 Compute Units (4 virtual cores with 2 EC2 Compute Units each), 1690 GB of local instance storage, 64-bit platform. One EC2 Compute Unit (ECU) provides the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor. For running costs, see Figure 7.

The only exception is the patents dataset which was too large to be processed on the desktop computer in reasonable time and consequently results are not reported at present.

With respect to data sizes, the statistics for the three datasets are as follows:

- Patents: The corpus consisted of 100 000 patent documents, with overall size of 5.47 GB. The mean document size is around 58kB, with the smallest document WO-2004077936-A3 being under 3kB (abstract only) and the largest: EP-1777296-A2 at 5.94 MB. The majority of the patent documents are fairly well clustered around the middle, with lower quartile: 22kB, median of 43kB, and the upper quartile: 70kB.

- News: The news corpus currently consists of 20 000 HTML pages (1.31 GB), collected from the web sites of the BBC and CNN as news broadcaster and Independent and the Guardian as newspapers. The shortest document was just 9 characters (from the CNN web site), whereas the longest – 230 kB. Similarly to the patents dataset, most news articles are well clustered around the middle, with average size of 68.7 kB.

- Twitter data: We randomly selected 50 million tweets from a 1TB dataset. No filtering for language was carried out, since we wanted to test how our semantic annotation pipeline would cope with a dataset in multiple languages.
The JSON of the 50 million tweets was converted into XML, prior to running the experiments. The resulting corpus was 6GB due to the fact that we only kept the tweet text, tweet ID and the ID of the author. This basic information was sufficient for our benchmarking purposes. The 50 million tweets were chosen to be a usefully sized dataset, 10 times larger than the one used by Abel et al [14] and 10 times smaller than the one used by Laniado and Mika [15]. Additional experiments not reported here showed that on all platforms processing time scales linearly with the number of tweets. Consequently, the performance and cost figures reported here can easily be adjusted for the smaller and the larger tweet datasets respectively.

**Fig. 7. Experiment Results**

<table>
<thead>
<tr>
<th>Experiment 1: Patents</th>
<th>CPU Time</th>
<th>Computer Time</th>
<th>Clock Time</th>
<th>Speed (Kils)</th>
<th>Cost (GBP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100,000 patent documents</td>
<td>Desktop</td>
<td>91:42:00</td>
<td>18:39:54</td>
<td>18:39:54</td>
<td>85.33</td>
</tr>
<tr>
<td></td>
<td>Server</td>
<td>162:38:00</td>
<td>16:56:37</td>
<td>02:03:32</td>
<td>773:6</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Experiment 2: News</td>
<td>Desktop</td>
<td>05:20:19</td>
<td>06:20:19</td>
<td>06:20:19</td>
<td>71.52</td>
</tr>
<tr>
<td>20,000 documents</td>
<td>Server</td>
<td>04:43:00</td>
<td>03:08:00</td>
<td>03:08:00</td>
<td>121.86</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>07:47:00</td>
<td>01:21:20</td>
<td>00:35:31</td>
<td>645:04</td>
</tr>
<tr>
<td>50,000,000 tweets</td>
<td>Server</td>
<td>22:16:15</td>
<td>03:19:12</td>
<td>03:19:12</td>
<td>516.53</td>
</tr>
<tr>
<td></td>
<td>Cloud</td>
<td>40:08:00</td>
<td>07:00:14</td>
<td>01:25:46</td>
<td>1199.69</td>
</tr>
</tbody>
</table>

Firstly, in terms of costs, the price per GB for named entity recognition is very low (between 1.35 and 1.50). The patents application is significantly more complex and therefore, its cost per GB processed is approximately double. Similarly, its performance figures should not be compared directly to those obtained on news and tweets. The more general point is that the cost per GB of text processed on GATECloud.net varies widely depending on the complexity of the underlying semantic annotation algorithms. In addition, specifically on the patents dataset, significant amount of time was spent on processing the fewer larger documents of over 1MB in size. Co-reference on such large documents, in particular, tends to be much more time consuming, due to the large number of candidate entities that need to be checked. The complexity of most other components tends to be close to linear with respect to document size.
With respect to CPU time taken, on the cloud this always tends to be higher than the CPU times for the desktop and server. This is due to the overhead in using distributed computing, and more specifically, due to the need to split the large datasets into batches that run on the virtual machines, as well as efficiency lost to virtualisation. In addition, the actual hardware specifications of the Amazon Extra Large virtual machines are not as good as those of our server, but are quite comparable to our desktop configuration.

In general, as can be seen from the computer time statistics, the major benefit of using GATECloud.net comes from the significant reduction in wall clock time, taken for each of the experiments. There tends to be between 10- and 20-fold improvement over the desktop. For example, the processing time for the news dataset is reduced down from over 5 hours to 35 minutes, which can help significantly not only for processing large-scale datasets, but also during development and testing of the semantic annotation algorithms. The improvements are even more staggering on the tweets dataset, where time goes down from 32 hrs to 1 hr and 25 mins. From the execution logs we have discovered that a third of that time (i.e., around 30 minutes) was taken by the data splitting and distribution operations, so we have already started work on optimising these further.

With respect to the gains made by using GATECloud.net instead of a local powerful server, these are also significant improvements (5-fold on the news set and 10-fold on the patents data). The benefits are less pronounced on tweets, due to their large number and small size. In general, GATECloud.net has been optimised for processing medium- to large-sized documents, where the benefits are most pronounced. In future work, we will be working on improving the infrastructure’s performance on large collections of smaller documents.

9 Conclusions

GATECloud.net empowers semantic web researchers to run experiments on large datasets, at an affordable cost, and without requiring expensive in-house compute infrastructure and system administration personnel. This cloud-based semantic annotation infrastructure makes it possible to obtain a fully-configured web-based annotation tool and/or scaleable semantic search index in a matter of minutes. The costs incurred only cover the actual number of compute hours used, thus saving researchers money by alleviating the need to purchase and maintain expensive servers, which may be utilised only occasionally.

The cost-benefit experiments reported here highlighted some performance issues on large collections of smaller documents. This will be one of the areas for future improvement. We also plan to carry out further experiments and make available public benchmark datasets, which will enable researchers to estimate upfront the likely costs of running their custom-made semantic annotation algorithms on GATECloud.net. Further minor improvements to the user interface, as well as the implementation of a public API will also be addressed.
References

2. Foster, I.: Globus online: Accelerating and democratizing science through cloud-based services. IEEE Internet Computing 15 (2011) 70–73