Extracting Information from Social Media with GATE

Kalina BONTCHEVA [1] and Leon DERCZYNSKI [2]

Abstract.
Information extraction from social media content has only recently become an active research topic, following early experiments which showed this genre to be extremely challenging for state-of-the-art algorithms. Unlike carefully authored news text and other longer content, social media content poses a number of new challenges, due to shortness, noise, strong contextual anchoring, and highly dynamic nature.

This chapter provides a thorough analysis of the problems and describes the most recent GATE algorithms, specifically developed for extracting information from social media content. Comparisons against other state-of-the-art research on this topic are also made. These new GATE components have now been bundled together, to form the new TwitIE information extraction pipeline, distributed as a GATE plugin.

Keywords. Information Extraction, GATE, social media

1. Introduction

In recent years, social media – and microblogging in particular – have established themselves as high-value, high-volume content, which organisations increasingly wish to analyse automatically. For instance, the widely popular Twitter microblogging platform has around 288 million active users, posting over 500 million tweets a day,[3] and has the fastest growing network in terms of active usage.[4]

Researchers have therefore started to study the problem of mining social media content automatically. The focus of this chapter is on information extraction, but other hotly researched topics include opinion mining [1], summarisation [2], and visual analytics and user and community modelling [3]. It is relevant in many application contexts, including knowledge management, competitor intelligence, customer relation management, eBusiness, eScience, eHealth, and eGovernment.

Information extraction from social media content has only recently become an active research topic, following early experiments which showed this genre to be extremely challenging for state-of-the-art algorithms. For instance, named entity recognition methods typically have 85-90% accuracy on longer texts, but 30-50% on tweets [4,5].

[1] Department of Computer Science, 211 Portobello, Sheffield S10 1SL, UK; Email: kalina@dcs.shef.ac.uk
[2] Department of Computer Science, 211 Portobello, Sheffield S10 1SL, UK; Email: leon@dcs.shef.ac.uk
[3] See http://news.cnet.com/8301-1023_3-57541566-93/report-twitter-hits-half-a-billion-tweets-a-day/.
[4] See http://globalwebindex.net/thinking/social-platforms-gwi-8-update-decline-of-local-social-media-platforms/.

The aim of this chapter is to provide a thorough analysis of the problems and to describe the most recent GATE algorithms, specifically developed for extracting information from social media content. Comparisons against other state-of-the-art research on this topic are also made. These new GATE components have now been bundled together, to form the new TwitIE IE pipeline, made available as a GATE plugin.

As the chapter progresses, the information extraction tasks and algorithms become progressively more complex and, consequently, more error-prone. For instance, on longer, well-formed content, tokenisation and part of speech (POS) tagging are typically performed with 95-98% accuracy, entity recognition between 90-95%, and even less for parsing. Genre adaptation is also a major issue, and again performance on tweets varies more widely as the tasks become more complex.

Consequently, there is a trade-off between the sophistication of the linguistic analysis and its accuracy, especially on unseen, noisier types of text. As a result, it is advisable, as an integral part of system development, to carry out rigorous quantitative evaluation against a gold standard dataset. It is through such experiments that it is possible to establish the usefulness of each of the text mining processing steps. GATE offers a wide range of evaluation tools described in detail in [6] and [7].

On large datasets, computational complexity and implementation efficiency would also need to be considered. To address this challenge, the GATE family has been extended recently with GATECloud (see Section 3.2 for details).

The rest of the chapter is structured as follows. Section 2 discusses the characteristics of social media content, which make it particularly challenging for state-of-the-art text mining algorithms. Next Section 3 introduces briefly the GATE family of text mining tools. Since the focus of this chapter is primarily on Information Extraction (IE) methods, Section 4 provides a brief overview of IE and differentiates it from information retrieval. Next Section 5 discusses in details IE from social media and introduces the TwitIE pipeline of reusable GATE components for extracting information from social media content. The chapter concludes with a brief discussion on outstanding challenges and future work.

2. Social Media Streams: Characteristics, Challenges and Opportunities

Social media sites allow users to connect with each other for the purpose of sharing content (e.g. web links, photos, videos), experiences, professional information, and online socialising with friends. Users create posts or status updates and social media sites circulate these to the user's social network. The key difference from traditional web pages is that users are not just passive information consumers, but many are also prolific content creators.

Social media can be categorised on a spectrum, based on the type of connection between users, how the information is shared, and how users interact with the media streams:

- Interest-graph media [8], such as Twitter, encourage users to form connections with others based on shared interests, regardless of whether they know the other person in real life. Connections do not always need to be reciprocated. Shared information comes in the form of a stream of messages in reverse chronological order.

- Social networking sites (SNS) encourage users to connect with people they have real-life relationships with. Facebook, for example, provides a way for people to share information, as well as comment on each other's posts. Typically, short contributions are shared, outlining current events in users' lives or linking to something on the internet that users think their friends might enjoy. These status updates are combined into a time-ordered stream for each user to read.
- Professional Networking Services (PNS), such as LinkedIn, aim to provide an introductions service in the context of work, where connecting to a person implies that you vouch for that person to a certain extent, and would recommend them as a work contact for others. Typically, professional information is shared and PNS tend to attract older professionals [9].
- Content sharing and discussion services, such as blogs, video sharing (e.g. YouTube, Vimeo), slide sharing (e.g. SlideShare), and user discussion/review forums (e.g. CNET). Blogs usually contain longer contributions. Readers might comment on these contributions, and some blog sites create a time stream of blog articles for followers to read. Many blog sites also advertise automatically new blog posts through their users' Facebook and Twitter accounts.

These different kinds of social media, coupled with their complex characteristics, make text mining extremely challenging. State-of-the-art natural language processing algorithms have been developed primarily on news articles and other carefully written, long web content [10]. In contrast, most social media streams (e.g. tweets, Facebook messages) are strongly inter-connected, temporal, noisy, short, and full of slang, leading to severely degraded results[5].

These challenging social media characteristics are also opportunities for the development of new text mining approaches, which are better suited to media streams:

Short messages (microtexts): Twitter and most Facebook messages are very short (140 characters for tweets). Many semantic-based methods reviewed below supplement these with extra information and context coming from embedded URLs and hashtags[6]. For instance, Abel *et al* [13] augment tweets by linking them to contemporaneous news articles, whereas Mendes *et al* exploit online hashtag glossaries to augment tweets [14].

Noisy content: social media content often has unusual spelling (e.g. 2moro), irregular capitalisation (e.g. all capital or all lowercase letters), emoticons (e.g. :-P), and idiosyncratic abbreviations (e.g. ROFL, ZOMG). Spelling and capitalisation normalisation methods have been developed [15], coupled with studies of location-based linguistic variations in shortening styles in microtexts [16]. Emoticons are used as strong sentiment indicators in opinion mining algorithms (see e.g. [17]).

Conversational : this content has very conversational linguistic style, including ubiquitous use of slang and swear words. Moreover, microblog services like Twitter hide conversations amongst the stream of tweets and to fully interpret the meaning of a reply tweet it is often necessary to process it together with the preceeding tweets

[5] For instance, named entity recognition methods typically have 85-90% accuracy on news but only 50% on tweets [11,4].

[6] A recently study of 1.1 million tweets has found that 26% of English tweets contain a URL, 16.6% – a hashtag, and 54.8% contain a user name mention [12].

in the conversation. In many cases, however, detecting conversational threads automatically is far from trivial, since threads can branch and more participants join.

Temporal: in addition to linguistic analysis, social media content lends itself to analysis along temporal lines, which is a relatively under-researched problem. Addressing the temporal dimension of social media is a pre-requisite for much-needed models of conflicting and consensual information, as well as for modelling change in user interests. Moreover, temporal modelling can be combined with opinion mining, to examine the volatility of attitudes towards topics over time.

Social context is crucial for the correct interpretation of social media content. Semantic-based methods need to make use of social context (e.g. who is the user connected to, how frequently they interact), in order to derive automatically semantic models of social networks, measure user authority, cluster similar users into groups, as well as model trust and strength of connection.

User-generated: since users produce, as well as consume social media content, there is a rich source of explicit and implicit information about the user, e.g. demographics (gender, location, age, etc.), interests, opinions. The challenge here is that in some cases, user-generated content is relatively small, so corpus-based statistical methods cannot be applied successfully.

Multilingual: Social media content is strongly multilingual. For instance, less than 50% of tweets are in English, with Japanese, Spanish, Portuguese, and German also featuring prominently [12]. Unfortunately, semantic technology methods have so far mostly focused on English, while low-overhead adaptation to new languages still remains an open issue. Automatic language identification [12,18] is an important first step, allowing applications to first separate social media in language clusters, which can then be processed using different algorithms.

3. The GATE Family of Text Mining Tools: An Overview

The GATE family of open-source text mining tools has grown over the years to include a desktop application for researchers developing new algorithms, a collaborative workflow-based web application, an annotation indexing and retrieval server, a Java library, an architecture and a process. To summarise, GATE comprises:

- **GATE Developer**: an integrated development environment (IDE) for language processing components, which is bundled with a widely used information extraction [6] system and a diverse set of several hundred other plugins [19];
- **GATE Cloud** [20], a cloud computing solution for hosted large-scale text processing;
- **GATE Teamware** [21]: a collaborative environment for large-scale manual semantic annotation projects built around a workflow engine and a heavily-optimised backend service infrastructure;
- a multi-paradigm index server, **GATE Mímir** [22], which can be used to index and search over text, annotations, semantic schemas (ontologies), and semantic meta-data (instances), allowing queries that arbitrarily mix full-text, structural, linguistic and semantic constraints and that can scale to terabytes of text;

- a framework, **GATE Embedded**: an object library optimised for inclusion in diverse applications giving access to all the services used by GATE Developer and others;
- an architecture: a high-level organisational picture of language processing software composition;
- a process for the creation of robust and maintainable NLP applications;

Note that GATE Developer and Embedded are bundled together, and in early distributions were referred to just as 'GATE'.

3.1. GATE Developer

GATE Developer is a specialist Integrated Development Environment (IDE) for language engineering R&D. It is analogous to systems like Eclipse or Netbeans for programmers, or Mathematica or SPSS for mathematics or statistics work. The system performs tasks such as:

- Visualisation and editing of domain-specific data structures associated with text: annotation graphs, ontologies, terminologies, syntax trees, etc.
- Constructing applications from sets of components (or plugins).
- Measurement, evaluation and benchmarking of automatic systems relative to *gold standard* data produced by human beings, or to previous runs of variants of experimental setups.

A sophisticated graphical user interface provides access to the models of the GATE architecture and particular instantiations of that architecture.

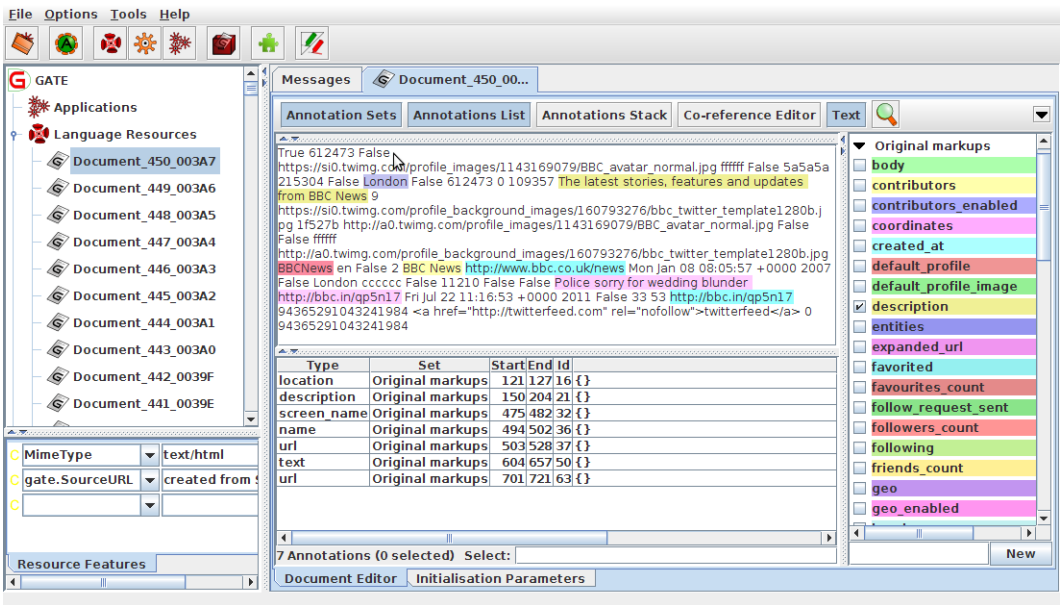

Figure 1. The GATE Developer Interface

Figure 1 displays a tweet, where the tweet text and JSON tweet metadata are imported as document content. The central pane shows a version of the source text from which formatting markup has been removed (and converted into arcs in an annotation graph associated with the document). The left pane details resources loaded in the system, including any application being used to annotate the text (e.g. TwitIE - see below) and the documents under analysis. The right pane lists the annotation types that exist in the document. Annotations are organised in annotation sets and here the "Original markups" set is shown, where the JSON fields are used to create different annotations. For example, a `description` annotation is created, which covers the text from the user profile ("The latest stories..."); a `text` annotation covers the tweet text ("Police sorry for..."), etc. The central pane highlights the selected annotation types and there is also an optional annotations list table underneath, which shows more details on each annotation, including its start and end character offset, any features contained, etc.

3.2. GATE Embedded

Underlying GATE Developer (and most of the rest of the GATE family tools) is an object-oriented Java framework called GATE Embedded. Some of the architectural principles which we adopted when developing the framework are as follows:

- Neutrality. The framework tries hard to be non-prescriptive and theory neutral. This is a strength because it means that no approach to language processing that users favour is excluded, but it is also a weakness because more restricted and specialised tools can capture more abstractions about their target domains, hence:
- Re-use. We minimise the impact of that weakness by emphasising re-use and interoperation with related systems, and avoiding reimplementation wherever possible. Thus GATE provides diverse XML support, integration with the OWLIM semantic repository [23], the Weka machine learning library [24], the Lingpipe[7] and OpenNLP[8] language analysis pipelines, the SVM Lite library [25], and many others (see [7]).
- Componentisation. Almost everything in GATE is modelled as a component, and the various component sets are all user-extendable. This means that all of the functions of the system can be swapped out, extended or replaced by users and developers with specific needs.
- Multiple usage modes. Almost all operations are available both from API (GATE Embedded) and UI (GATE Developer). A common process is to develop and test using the IDE and then embed in the target environment using the Java library. In both cases exactly the same underlying framework is in operation.

The set of plugins that are integrated with GATE is called CREOLE, a Collection of REusable Objects for Language Engineering. Components are defined as Java Beans bundled with XML configuration, and the overheads imposed by the model are very small (the minimal component comprises a few lines of Java code plus a few lines of XML). Components can be packaged in the same way as other Java libraries and can be loaded over the network via a URL.

[7] http://alias-i.com/lingpipe/
[8] http://opennlp.apache.org/

GATE Embedded encapsulates a number of modular APIs for text processing, which cover functions including:

- persistence, visualisation and editing
- a finite state transduction language (JAPE, a Java Annotation Patterns Engine [26])
- extraction of training instances for machine learning (ML – methods for automated abstraction of pattern recognition models from data, see e.g. [27])
- pluggable ML implementations (e.g. Weka, [28], support vector machines [25], etc.)
- components for language analysis, e.g. parsers, taggers and stemmers for various languages
- a very widely used information extraction system (ANNIE) which has been evaluated in comparative events including MUC, TREC, ACE, DUC, Pascal, NTCIR, etc. [29,30,31,32,33]
- indexing and search tools (including Lucene, Google and Yahoo plugins)
- a simple API for RDF, OWL and Linked Data

The modularity of the library and the low level of commitment imposed on its clients has proven flexible enough to prosper for more than a decade since the release of version 2 (the first Java version).

3.3. GATE Cloud

`GATECloud.net` [34] is a web-based platform which deploys GATE analysis pipelines and GATE server products on Amazon EC2 (Elastic Compute Cloud – a popular cloud computing platform). GATE annotation pipelines provide a PaaS (Platform as a Service [35]) arrangement: software produced using GATE Developer/Embedded can be trivially scaled up to large data volumes. In this way GATE Teamware and Mímir on the cloud provide a SaaS (Software as a Service) arrangement where responsibility for installation and administration are removed from the end user.

GATE Cloud is based on a **parallel** execution engine of automatic annotation processes (using pooling and model sharing to minimise the load on individual nodes) and **distributed** execution of the parallel engine [34]. Its characteristics include:

- **scalability**: auto-scaling of processor swarms dependent on loading;
- **flexibility**: user-visible parameters configure system behaviour, select the GATE application being executed, the input protocol used for reading documents, the output protocol used for exporting the resulting annotations, and so on;
- **robustness**: jobs run unattended over large data sets using a parallelisation system that has been extensively tested and profiled.

In general, making optimal use of virtual IaaS infrastructures for data-intensive NLP again comes with a significant overhead, e.g. having to learn the intricacies of Amazon's APIs for the elastic compute cloud and simple storage services. An added complication is that not all NLP algorithms are actually implementable in the MapReduce paradigm, e.g. those requiring a shared global state.

Therefore, the GATECloud platform was designed as an open cloud platform that can run existing NLP algorithms on large datasets, as well as algorithms already adapted

to Hadoop. Cloud infrastructural issues are dealt with by the GATECloud platform, completely transparently to the user, including load balancing, efficient data upload and storage, deployment on the virtual machines, security, fault tolerance, and resource usage.

4. Information Extraction: An Overview

Information Extraction (IE) is a technology based on analysing natural language in order to extract snippets of information. For example, when extracting information about companies, key information to be identified would be the company address, contact phone, fax numbers, and e-mail address, products and services, members of the board of directors and so on.

The process takes texts (and sometimes speech) as input and produces fixed-format data as output. This data may be used directly for display to users, or may be stored in a database or spreadsheet for later analysis, or may be used for indexing purposes in Information Retrieval (IR) applications such as Internet search engines like Google.

IE is quite different from IR:

- an IR system finds relevant texts and presents them to the user;
- an IE application analyses texts and presents only the specific information from them that the user is interested in.

The main tasks carried out during information extraction are:

- named entity recognition, which consists on the identification and classification of different types of names in text;
- coreference resolution, which is the task of deciding if two linguistic expressions refer to the same entity in the discourse;
- and relation extraction, which identifies relations between entities in text.

Information extraction usually employs the following natural language processing components: Part-Of-Speech (POS) taggers, morphological analyser, named entity recognisers, full (or shallow) parsing, and semantic interpretation. Generic versions of these linguistic processors are available in GATE [6]), although some require adaptation to social media.

There are two main classes of approaches to information extraction:

1. rule-based systems which are built by language engineers, who design lexicons and rules for extraction; and
2. machine learning systems which are trained to perform one or more of the IE tasks. Learning systems are given either an annotated training corpus (i.e., supervised machine learning) or unannotated corpus together with a small number of seed examples (i.e., unsupervised or lightly supervised methods).

The advantages of rule-based approaches are that they do not require training data to create (although a small gold standard is needed for evaluation) and harness human intuition and domain knowledge. Depending on the lexical and syntactic regularity of the target domain, rule creation ranges from extremely fast (when few, clear patterns exist) to rather time-consuming (if more ambiguities are present). Depending on the system design, some changes in requirements may be hard to accommodate. Since rule-based

systems tend to require at least basic language processing skills, they are sometimes perceived as more expensive to create.

In comparison, machine learning approaches typically require at least some human-annotated training data, in order to reach good accuracy. While the cost per individual annotator is lower than the cost of language engineers, given the size of data needed, often more than one or two annotators are required. This raises the problem of inter-annotator agreement (or consistency), since the accuracy of the learnt models can be affected significantly by noisy, contradictory training data. However, getting training annotators to agree on their labels is again dependent on the complexity of the target annotations and could in itself be rather time-consuming. Another potential problem could arise if the semantic annotation requirements change after the training data has been annotated, since this may require substantial re-annotation.

To summarise, both types of approaches have advantages and drawbacks and the choice of which one is more appropriate for a given application depends on the target domain, the complexity of the semantic annotations (including the size of the ontology), and the availability of trained human annotators and/or language engineers. Last but not least, there is no reason why one cannot have a hybrid approach, which uses both rules and machine learning.

5. IE from Social Media with GATE

GATE comes pre-packaged with the ANNIE general purpose IE pipeline. ANNIE consists of the following main processing resources: tokeniser, sentence splitter, POS tagger, gazetteer lists, finite state transducer (based on GATE's built-in regular expressions over annotations language [6]), orthomatcher and coreference resolver. The resources communicate via GATE's annotation API [6], which is a directed graph of arcs bearing arbitrary feature/value data, and nodes rooting this data into document content (in this case text).

The ANNIE components can be used individually or coupled together with new modules in order to create new applications. For example, we reuse the ANNIE sentence splitter and name gazetteers unmodified on social media content, but re-train and adapt other components to the specifics of this genre.

Adaptation to the specifics of the social media genre is required, in order to address the genre-specific challenges discussed in Section 2. General-purpose linguistic tools such as POS taggers and entity recognisers do particularly badly on such texts (see Sections 5.4 and 5.6 respectively).

Therefore, we have developed TwitIE – a customisation of ANNIE, specific to social media content, which we have currently tested most extensively on microblog messages. The latter content is both readily available as a large public stream and also is the most challenging to process with generic IE tools, due to the shortness, noisy nature, and prevalence of slang and Twitter-specific conventions.

Figure 2 shows the TwitIE pipeline and its components. TwitIE is distributed as a plugin in GATE, which needs to be loaded for these processing resources to appear in GATE Developer. Components reused from ANNIE without any modification are shown in blue, whereas the red ones are new and specific to social media.

The first step is language identification, which is discussed next (Section 5.1), followed by the TwitIE tokeniser (Section 5.2).

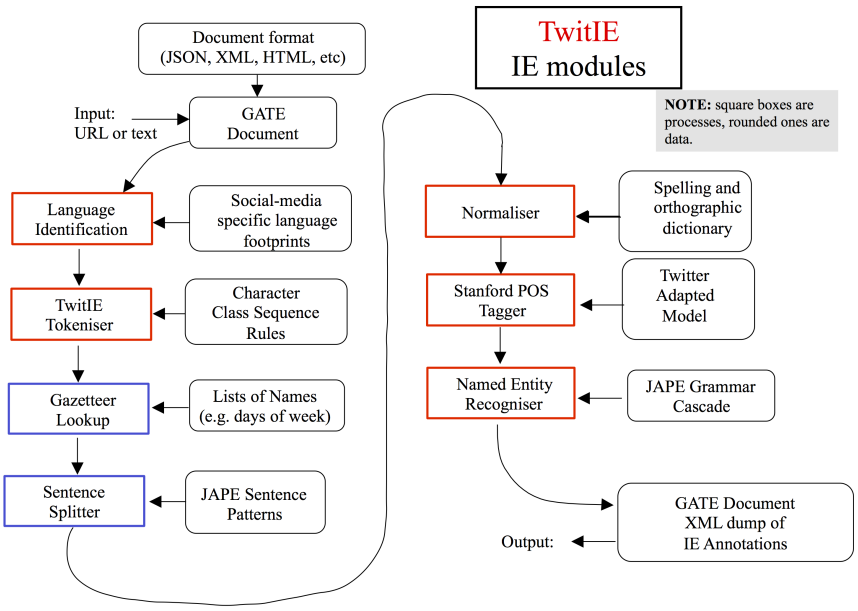

Figure 2. The TwitIE Information Extraction Pipeline

The **gazetteer** consists of lists such as cities, organisations, days of the week, etc. It not only consists of entities, but also of names of useful *indicators*, such as typical company designators (e.g. 'Ltd.'), titles, etc. The gazetteer lists are compiled into finite state machines, which can match text tokens. TwitIE reuses the ANNIE gazetteer lists, at present, without any modification.

The **sentence splitter** is a cascade of finite-state transducers which segments text into sentences. This module is required for the POS tagger. Again, at present, the ANNIE sentence splitter is reused without modification, although when processing tweets, it is also possible to just use the text of the tweet as one sentence, without further analysis.

The normaliser, the adapted POS tagger, and named entity recognition are discussed in detail in Sections 5.3, 5.4, and 5.6 respectively.

We have also included a section on stemming and morphological analysis in GATE (Section 5.5), since these can be useful in some applications. By default they are not included in the TwitIE pipeline, but can easily be added, if required.

5.1. Language Identification

It is critical to determine the language in which a document is written, in order to know which tools to apply. The *language identification* task is thus typically performed before other linguistic processing, having as its goal is to output a language suggestion given some unprocessed text. We consider two types of approach: n-gram frequency-based and n-gram information gain-based.

Both approaches include an implicit tokenisation step, though this is speculative and does not inform later processes in the GATE TwitIE pipeline. TextCat [36] relies on n-gram frequency models to discriminate between languages, relying on token sequences

System	Overall accuracy	English	Dutch	French	German	Spanish
TextCat	89.5%	88.4%	90.2%	86.2%	94.6%	88.0%
langid	89.5%	92.5%	89.1%	89.4%	94.3%	83.0%
TextCat (in-genre)	**97.4%**	**99.4%**	**97.6%**	**95.2%**	**98.6%**	**96.2%**
langid (in-genre)	87.7%	88.7%	88.8%	88.0%	92.5%	81.6%

Table 1. Language classification accuracy on the ILPS dataset for systems before and after adaptation to the microblog genre.

that are strong language differentiators. The information gain-based langid.py [37] uses n-grams to learn a multinomial event model, with a feature selection process designed to cope with variations of expression between text domains. They have both been adapted for microblog text, using human-annotated data from Twitter. The TextCat adaptation [12] works on a limited set of languages; the langid.py adaptation [38] – on 97.

We evaluated four system runs on the ILPS TextCat microblog evaluation dataset[9]. Results are given in Table 1, with the Twitter-specific versions marked "-twitter". It should be noted that the adapted version of TextCat has a slightly easier task than langid.py does, since it expects only five language choices, whereas the adapted langid.py is choosing labels from a set of 97 languages. The latter assigned a language outside the five available to 6.3% of tweets in the evaluation set. Why the adapted langid.py performed worse than the generic version is not clear; the results are quite close for some languages, and so if an approximate 6% improvement could be made in these cases, the Twitter-adapted version would be better.

The results below demonstrate that language identification is harder on tweets than longer texts. Nevertheless, it has reasonable accuracy, which can be used to inform choices of later tools, without introducing too many errors.

Given the above much higher results, only the adapted TextCat is distributed as part of the GATE TwitIE plugin. Due to the shortness of tweets, it makes the assumption that each tweet is written in only one language. The choice of languages used for categorization is specified through a cofiguration file, supplied as an initialisation parameter.

Figure 3 shows three tweets – one English, one German, and one French. TwitIE TextCat was used to assign automatically the lang feature to the tweet text (denoted by the Tweet annotation).

Given a collection of tweets in a new language, it is possible to train TwitIE TextCat to support that new language as well. This is done by using the Fingerprint Generation PR, included in the `Language_Identification` GATE Plugin [7]. It builds a new ïňĄngerprint from a corpus of documents.

Reliable tweet language identification allows us to only process those tweets written in English with the TwitIE English POS tagger and named entity recogniser. GATE also provides POS tagging and named entity recognition in French and German, so it is possible to process tweets differently, dependent upon the language they are written in. This is achieved by making the execution of these components conditional on the respective tweet being in English, German, or French, by using a Conditional Corpus Pipeline [7].

[9] http://ilps.science.uva.nl/resources/twitterlid

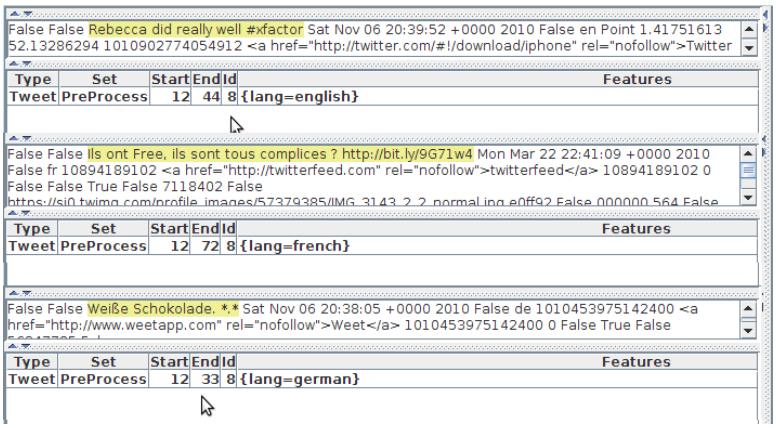

Figure 3. Example Tweets Annotated by TextCat

5.2. Tokenisation

Tokenisation is the task of splitting the input text into very simple units, called tokens. Tokenisation is a required step in any linguistic processing application, since more complex algorithms typically work on tokens as their input, rather than using the raw text. Consequently, it is important to use a high-quality tokeniser, as errors are likely to affect the results of all subsequent NLP algorithms.

Different languages require different tokenisers, with some easier than others [39]. Even punctuation use can differ between languages for the microblog genre, in which "smileys" (comprised of extended sequences of punctuation symbols) are prevalent.

Commonly distinguished types of tokens are numbers, symbols (e.g., $, %), punctuation and words of different kinds, e.g., uppercase, lowercase, mixed case. Tokenising well-written text is generally reliable and reusable, since it tends to be domain-independent. One widely used tokeniser for English is bundled in the open-source ANNIE system in GATE [6].

However, such general purpose tokenisers need to be adapted to work correctly on social media, in order to handle specific tokens like URLs, hashtags (e.g. #nlproc), user mentions in microblogs (e.g. @GateAcUk), special abbreviations (e.g. RT, ROFL), and emoticons. A study of 1.1 million tweets established that 26% of English tweets have a URL, 16.6% – a hashtag, and 54.8% – a user name mention [12]. Therefore, tokenising these accurately is very important.

To take one tweet as an example:

```
#WiredBizCon #nike vp said when @Apple saw what
http://nikeplus.com did, #SteveJobs was like wow I didn't
expect this at all.
```

One option is to tokenise on white space alone, but this does not work that well for hashtags and @mentions. In our example, if we have #nike and @Apple as one token each, this will make their recognition as company names harder, since the named entity recognition algorithm will need to look at sub-token level. Similarly, tokenising on white space

Approach	Precision	Recall	F1
ANNIE Tokeniser	90%	72%	80%
TwitIE Tokeniser	**98%**	**94%**	**96%**

Table 2. Tokeniser performance on sample microblog text

and punctuation characters does not work well, since in that case URLs get separated in more than one token (e.g. http, nikeplus), as are emoticons and email addresses.

Therefore, the TwitIE tokeniser in GATE is an adaptation of the ANNIE English tokeniser. It follows Ritter's tokenisation scheme [4]. More specifically, it treats abbreviations (e.g. RT, ROFL) and URLs as one token each. Hashtags and user mentions are two tokens (i.e., # and nike in the above example) plus a separate annotation HashTag covering both. Capitalisation is preserved, but an orthography feature is added: all caps, lowercase, mixCase. Lowercasing and emoticons are optionally done in separate modules, since they are not always needed. Consequently, tokenisation is faster and more generic, as well as more tailored to the needs of named entity recognition.

In terms of implementation, the TwitIE Tokeniser relies on a set of regular expression rules which are then compiled into a finite-state machine. This differs from most other tokenisers in that it maximises efficiency by doing only very light processing, and enabling greater flexibility by placing the burden of deeper processing on the grammar rules, which are more adaptable. For example, there is a specialised set of rules for recognising the hashtags and user mentions.

In order to evaluate the benefit from making adaptations to microblog and social media content, we compare the TwitIE tokeniser against the general purpose ANNIE tokeniser in GATE [6] on Ritter's tweet dataset [4]. Performance was measured in terms of precision and recall, as well as F1 measure, and is given in Table 2.

The original tokeniser performed poorly, reaching an F1 of only 80% (near 100% is typical), and with many errors around punctuation and twitter-specific entities. This is too weak to accurately inform later tasks. Smileys cause some trouble for tokenisers, many of which do not occur in the training data. Orthographic errors are also rife in this genre, an analysis of which can be found in [40]. Aside from smileys and typos, the low performance of a conventional tokeniser such as this is mostly due to differing tokenisation rules regarding usernames and Twitter-specific content.

5.3. Normalisation

Noisy environments such as microblog text pose challenges to existing tools, being rich in previously unseen tokens, elision of words, and unusual grammar. Normalisation is commonly proposed as a solution for overcoming or reducing linguistic noise [41]. The task is generally approached in two stages: first, the identification of orthographic errors in an input discourse, and second, the correction of these errors.

Example 5.3.1 shows an original microblog message, including a variety of errors, and the post-normalisation repaired version.

Example 5.3.1 *Source text:* @DORSEY33 lol aw . i thought u was talkin bout another time . nd i dnt see u either !

Normalised text: @DORSEY33 lol aww . I thought you was talking about another time . And I didn't see you either !

As can be seen, not all the errors can be corrected (*was* ought to be *were*, for example) and some genre-specific slang remains – thought not in a particularly ambiguous sense or grammatically crucial place. Note the frequent shortening of words in messages entered by users, possibly attributable to both their effort to minimise the energy cost of communication and also out of habit of fitting within the tight message length limits typical of the genre.

Normalisation approaches are typically based on a correction list, edit-distance based, cluster-based, or a mixture of these, with hybrid approaches common. They often include a dictionary of known correctly-spelled terms, and refer to in-vocabulary (IV) and out-of-vocabulary (OOV) terms with respect to this dictionary.

One common source of errors on social media content is variation of spelling of proper names, which in turn impacts the accuracy of named-entity recognition. Therefore, resources have been developed to capture the many variations on spellings seen for given entities. For example, JRC-Names [42] is a list-based collection of name variations for many entities, coupled with an algorithm for matching target words to a given entity. An integration of JRC-Names into GATE is currently ongoing.

Dictionary-based approaches can be used to repair errors on all kinds of words, not just named entities; for example, Han et al. [43] construct a general-purpose spelling correction dictionary for microblogs. This achieves state-of-the-art performance on both the detection of mis-spelled words and also applying the right correction.

The TwitIE Normaliser is currently a combination of a generic spelling-correction dictionary and a spelling correction dictionary, specific to social media. The latter contains entries such as "2moro" and "brb", similar to Han et al. [43]. Figure 4 shows an example output from normalisation, where the abbreviation "Govt" has been normalised to government.

Instead of a fixed list of variations, it is also possible to use a heuristic to suggest correct spellings. Both text edit distance and phonetic distance can be used to find candidate matches for words identified as mis-spelled. Han and Baldwin [15] achieved good corrections in many cases by using a combination of Levenshtein distance and double-metaphone distance between known words and words identified as incorrectly entered. We experimented also with this normalisation approach in TwitIE, with mixed success. Namely, this method has higher recall – more wrong words can be corrected by the resource – but lower precision, in that some corrections are wrong. A detailed error analysis and evaluation of dictionary-based versus heuristic normalisation is presented in [44].

5.4. Part of Speech Tagging

Part-of-Speech (POS) tagging is concerned with tagging words with their part of speech, by taking into account the word itself, as well as the context in which it appears. A key part of this task is the tagset used and the distinctions that it makes. The main categories are verb, noun, adjective, adverb, preposition, etc. However, tagsets tend to be much more specific, e.g. distinguishing between singular and plural nouns. One commonly used tagset is the Penn Treebank one (referred to as PTB) [45].

In terms of approaches, researchers have achieved excellent results with Hidden Markov models, rule-based approaches, maximum entropy, and many other methods. GATE's English POS tagger [46] is a modified version of the Brill transformational rule-based tagger [47], which produces a part-of-speech tag as an annotation on each word

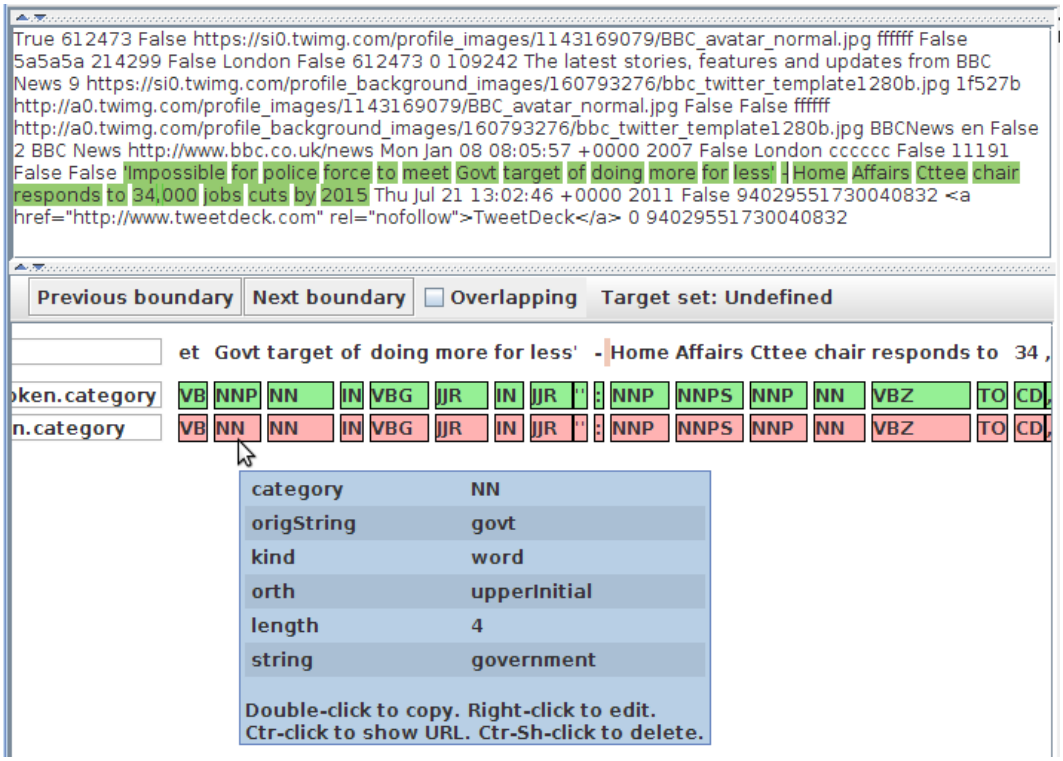

Figure 4. Comparing POS Tagger Output: A Normalisation Example

or symbol, using the Penn treebank tagset. The tagger uses a default lexicon and ruleset (the result of training on a large corpus taken from the Wall Street Journal). Both of these can be modified manually if necessary.

Later, Toutanova et al. [48] introduced the Stanford POS tagger, trained on newswire texts. It has sophisticated feature generation, especially for unknown words, and a highly configurable re-trainable engine. This is generally thought to represent the current state of the art for news texts and is also integrated in GATE. Models using the PTB set are available for both these taggers.

The accuracy of these general purpose English POS taggers is typically excellent (97-98%) on texts similar to those on which the taggers have been trained (mostly news articles). However, they are not suitable for microblogs and other short. noisy social media content, where their accuracy declines to 70-75% [44].

Thus the TwitIE application in GATE now contains an adapted model for the Stanford POS tagger, trained on PTB-tagged tweets. Extra tag labels have been added for retweets, URLs, hashtags and user mentions. The Stanford POS tagger was re-trained [44] using some hand-annotated tweets [4], the NPS IRC corpus [49], and news texts (the Wall Street Journal part of the Penn Treebank [50]). The resulting model achieves 83.14% POS tagging accuracy, which is still below the 97% achieved on news content.

The most common mistakes (just over 27%) arise from words which are common in general, but do not occur in the training data, indicating a need for a larger training POS-

tagged corpus of social media content. Another 27% of errors arise from slang words, which are ubiquitous in social media content and are also often misspelled (e.g. *LUVZ*, *HELLA* and *2night*) and another 8% from typos. Many of these can be addressed using normalisation (see Section 5.3). Close to 9% of errors arise from tokenisation mistakes (e.g. joined words). Lastly, 9% of errors are words, to which a label may be reliably assigned automatically, including URLs, hash tags, re-tweets and smileys, which we now pre-tag automatically with regular expressions and lookup lists.

Another frequently made mistake is tagging proper noun (`NN`/`NNP`) – an observation also made by [4]. Therefore, we use GATE's ANNIE gazetteer lists of personal first-names and cities [6] and, in addition, a manually constructed a list of unambiguous corporation and website names frequently-mentioned in the training data (e.g. *YouTube*, *Toyota*).

Therefore, by combining normalisation, gazetteer name lookup, and regular expression-based tagging of Twitter-specific POS tags, we achieve performance improvement from 83.14% accuracy to 86.93%. By generating additional 1.5M training tokens from automatically annotated tweets using two pre-existing POS taggers (namely [4] and [51]), we improve further the performance of the Twitter-adapted Stanford POS tagger to 90.54% token accuracy. For further details on the evaluation see [44].

The Twitter-tailored Stanford POS Tagger is part of the GATE TwitIE plugin. It is distributed with the specially trained model discussed above. In order to ensure the best possible performance, it needs to be run after the TwitIE tokeniser and normaliser. Since it is currently trained only on English content, it should only be run on tweets identified as English by the TwitIE language identifier.

Figure 4 shows an example tweet, which has been tagged both without normalisation (upper row of POS tags) and with tweet normalisation (the lower row of POS tags). The word "Govt" is normalised to government, which is then tagged correctly as NN, instead of NNP.

5.5. Stemming and Morphological Analysis

Another set of useful low-level processing components are stemmers and morphological analysers. Stemmers produce the stem form of each word, e.g. "driving" and "drivers" have the stem "drive", whereas morphological analysis tends to produce the root/lemma forms of the words and their affixes, e.g. "drive" and "driver" for the above examples, with affixes "ing" and "s" respectively.

GATE provides a wrapper for the widely used, open-source Snowball stemmers, which cover 11 European languages (Danish, Dutch, English, Finnish, French, German, Italian, Norwegian, Portuguese, Russian, Spanish and Swedish) and makes them straightforward to combine with the other low-level linguistic components. The stemmers are rule-based [52] and easy to modify, following the suffix-stripping approach of Porter.

The English morphological analyser in GATE is also rule-based, with the rule language supporting rules and variables that can be used in regular expressions in the rules. POS tags can taken into account if desired, depending on a configuration parameter.

5.6. Named Entity Recognition

Named entity recognition (**NER**) is difficult on user-generated content in general, and in the microblog genre specifically, because of the reduced amount of contextual informa-

System	Precision	Recall	F1
ANNIE	47%	**83%**	60%
TwitIE	77%	**83%**	**80%**
Ritter	73%	49%	59%

Table 3. Named Entity Recognition Performance

tion in short messages and a lack of curation of content by third parties (e.g. that done by editors for newswire). In this section, we examine how the ANNIE NER component from GATE [6] performs in comparison to a Twitter-specific approach, on a corpus of 2 400 tweets comprising 34 000 tokens [4]. Namely, we compare here against Ritter et al. [4], who take a pipeline approach, performing first tokenisation and PoS tagging before using topic models to find named entities. For an in-depth comparison against other approaches (including Stanford NER [53]) and error analysis, see [44].

Results are given in Table 3. We did not consider Percent-type entity annotations in these evaluations because there were so few (3 in the whole corpus) and they were all annotated correctly.

In the first experiment, we compared the default ANNIE pipeline against our TwitIE pipeline, and found an absolute Precision increase of 30% – mainly with respect to Date, Organization and in particular Person. Recall remained identical after TwitIE's customised tokenisation, normalisation, and POS tagging, which led to an increase of 20% in F1. Note that we did not consider the twitter-specific UserID annotation as a Person annotation, since these were all 100% correct. Because the regular ANNIE does not consider these, there were many false positive Person annotations as part of UserIDs.

Both ANNIE and TwitIE currently use the same named entity recognition grammars, so part of our ongoing research is now on adapting these to social media. As we can see microblog domain adaptation is critical to good NER. Thanks to adaptation in the earlier components in TwitIE, we demonstrate a +30% absolute precision and +20% absolute F1 performance increase. However, compared against state-of-the-art NER performance on longer news content, an overall F1 score of 80% leaves significant amounts of missed annotations and generates false positives.

6. Conclusion and Future Work

This chapter discussed the problem of extracting information from social media content and presented a number of state-of-the-art open-source GATE tools, comprising the TwitIE IE pipeline. As can be seen from the evaluation results reported here, even though significant inroads have been made into this challenging problem, there is still a significant gap in accuracy, when compared against performance on news texts, mostly due to insufficient linguistic context and lack of training data. Next we discuss how further improvements can be made.

One source of additional context is the geo-spatial information present in social media. A notable proportion comes explicitly geotagged [54], and studies suggest that it is possible to infer the geo-locations of about half of the remaining such content [55]. Social media also has at least a creation time as temporal context. None of this explicit and implicit spatio-temporal (ST) metadata is currently exploited by the methods discussed

above, which is one promising avenue for future work. Our ongoing work here is on evaluating and adapting the TIMEN approach [56] to social media content.

A second key major open issue is lack of sufficient amounts of training and evaluation gold standard corpora of social media content. For example, there are fewer than 10,000 tweets hand-annotated with named entities, which hampers the development of high-performance machine learning algorithms. Crowdsourcing has recently emerged as a promising method for creating shared evaluation datasets [57]. Adapting these efforts to the specifics of creating large-scale training and evaluation corpora is the second key focus of future work on TwitIE.

Acknowledgements

This work was supported by funding from the Engineering and Physical Sciences Research Council (grants EP/I004327/1 and EP/K017896/1) and by the EU-funded FP7 TrendMiner[10] project.

References

[1] Maynard, D., Bontcheva, K., Rout, D.: Challenges in developing opinion mining tools for social media. In: Proceedings of @NLP can u tag #usergeneratedcontent?! Workshop at LREC 2012, Turkey (2012)

[2] Rout, D., Bontcheva, K., Hepple, M.: Reliably evaluating summaries of twitter timelines. In: Proceedings of the AAAI Symposium on Analyzing Microtext. (2013)

[3] Bontcheva, K., Rout, D.: Making sense of social media through semantics: A survey. Semantic Web - Interoperability, Usability, Applicability (2013)

[4] Ritter, A., Clark, S., Mausam, Etzioni, O.: Named entity recognition in tweets: An experimental study. In: Proc. of Empirical Methods for Natural Language Processing (EMNLP), Edinburgh, UK (2011)

[5] Liu, X., Zhou, M., Wei, F., Fu, Z., Zhou, X.: Joint inference of named entity recognition and normalization for tweets. In: Proceedings of the Association for Computational Linguistics. (2012) 526–535

[6] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: Gate: an architecture for development of robust hlt applications. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, 7–12 July 2002. ACL '02, Stroudsburg, PA, USA, Association for Computational Linguistics (2002) 168–175

[7] Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6). (2011)

[8] Ravikant, N., Rifkin, A.: Why Twitter Is Massively Undervalued Compared To Facebook. TechCrunch (2010) http://techcrunch.com/2010/10/16/why-twitter-is-massively-undervalued-compared-to-facebook/.

[9] Skeels, M.M., Grudin, J.: When social networks cross boundaries: A case study of workplace use of Facebook and LinkedIn. In: Proceedings of the ACM 2009 international conference on Supporting group work. GROUP '09, New York, NY, USA, ACM (2009) 95–104

[10] Bontcheva, K., Cunningham, H.: Semantic annotation and retrieval: Manual, semi-automatic and automatic generation. In Domingue, J., Fensel, D., Hendler, J.A., eds.: Handbook of Semantic Web Technologies. Springer (2011)

[11] Liu, X., Zhang, S., Wei, F., Zhou, M.: Recognizing named entities in tweets. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. (2011) 359–367

[12] Carter, S., Weerkamp, W., Tsagkias, E.: Microblog language identification: Overcoming the limitations of short, unedited and idiomatic text. Language Resources and Evaluation Journal (2013)

[10] http://www.trendminer-project.eu/

[13] Abel, F., Gao, Q., Houben, G.J., Tao, K.: Semantic enrichment of Twitter posts for user profile construction on the social web. In: ESWC (2). (2011) 375–389
[14] Mendes, P.N., Passant, A., Kapanipathi, P., Sheth, A.P.: Linked open social signals. In: Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. WI-IAT '10, Washington, DC, USA, IEEE Computer Society (2010) 224–231
[15] Han, B., Baldwin, T.: Lexical normalisation of short text messages: makn sens a #twitter. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. HLT '11 (2011) 368–378
[16] Gouws, S., Metzler, D., Cai, C., Hovy, E.: Contextual bearing on linguistic variation in social media. In: Proceedings of the Workshop on Languages in Social Media. LSM '11 (2011) 20–29
[17] Pak, A., Paroubek, P.: Twitter Based System: Using Twitter for Disambiguating Sentiment Ambiguous Adjectives. In: Proceedings of the 5th International Workshop on Semantic Evaluation. (2010) 436–439
[18] Baldwin, T., Lui, M.: Language identification: The long and the short of the matter. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Los Angeles, California (June 2010) 229–237
[19] Cunningham, H., Tablan, V., Roberts, A., Bontcheva, K.: Getting more out of biomedical documents with gate's full lifecycle open source text analytics. PLoS Computational Biology **9**(2) (02 2013) e1002854
[20] Tablan, V., Roberts, I., Cunningham, H., Bontcheva, K.: Gatecloud.net: a platform for large-scale, open-source text processing on the cloud. Philosophical Transactions of the Royal Society A **371**(1983) (2013)
[21] Bontcheva, K., Cunningham, H., Roberts, I., Roberts, A., Tablan, V., Aswani, N., Gorrell, G.: GATE Teamware: A Web-based, Collaborative Text Annotation Framework. Language Resources and Evaluation (2013)
[22] Cunningham, H., Tablan, V., Roberts, I., Greenwood, M.A., Aswani, N.: Information Extraction and Semantic Annotation for Multi-Paradigm Information Management. In Lupu, M., Mayer, K., Tait, J., Trippe, A.J., eds.: Current Challenges in Patent Information Retrieval. Volume 29 of The Information Retrieval Series. Springer Berlin Heidelberg (2011) 307–327
[23] Kiryakov, A.: OWLIM: balancing between scalable repository and light-weight reasoner. In: Proceedings of the 15th International World Wide Web Conference (WWW2006), 23–26 May 2010, Edinburgh, Scotland (2006)
[24] Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, San Francisco, CA (October 1999)
[25] Li, Y., Bontcheva, K., Cunningham, H.: Adapting SVM for Data Sparseness and Imbalance: A Case Study on Information Extraction. Natural Language Engineering **15**(2) (2009) 241–271
[26] Cunningham, H., Maynard, D., Tablan, V.: JAPE: a Java Annotation Patterns Engine (Second Edition). Research Memorandum CS–00–10, Department of Computer Science, University of Sheffield, Sheffield, UK (November 2000)
[27] Carbonell, J., Michalski, R., Mitchell, T.: An Overview of Machine Learning. In Carbonell, J., Michalski, R., Mitchell, T., eds.: Machine Learning: An Artificial Intelligence Approach. Tioga Pub. Co., Palo Alto, CA (1983) 3–23
[28] Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann (October 1999)
[29] Gaizauskas, R., Wakao, T., Humphreys, K., Cunningham, H., Wilks, Y.: Description of the LaSIE system as used for MUC-6. In: Proceedings of the Sixth Message Understanding Conference (MUC-6), 6–8 November 1995, Morgan Kaufmann, California (1995) 207–220
[30] Voorhees, E.M., Harman, D.: Overview of the eighth Text REtrieval Conference (TREC-8). In: The Eighth Text REtrieval Conference (TREC-8), 16–19 November 1999, National Institute of Standards and Technology (NIST) (1999) 1–24
[31] Iwayama, M., Fujii, A., Kando, N.: Overview of Classification Subtask at NTCIR-5 Patent Retrieval Task. In: Proceedings of the Fifth NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access, 6–9 December 2005. (2005) 278–286
[32] Li, Y., Bontcheva, K., Cunningham, H.: SVM Based Learning System For Information Extraction. In Winkler, J., Niranjan, M., Lawerence, N., eds.: Deterministic and Statistical Methods in Machine Learn-

ing: First International Workshop, 7–10 September, 2004. Volume 3635 of Lecture Notes in Computer Science., Sheffield, UK, Springer Verlag (2005) 319–339
[33] Li, Y., Bontcheva, K., Cunningham, H.: Cost Sensitive Evaluation Measures for F-term Patent Classification. In: The First International Workshop on Evaluating Information Access (EVIA 2007), 15 May 2007, Tokyo, Japan (May 2007) 44–53
[34] Tablan, V., Roberts, I., Cunningham, H., Bontcheva, K.: GATECloud.net: a Platform for Large-Scale, Open-Source Text Processing on the Cloud. Philosophical Transactions of the Royal Society A: Mathematical, Physical & Engineering Sciences **371**(1983) (2013) 20120071
[35] Geelan, J.: Twenty-one experts define cloud computing. Cloud Computing Journal (2009) 1–5
[36] Cavnar, W., Trenkle, J.: N-gram-based text categorization. In: Proceedings of the Annual Symposium on Document Analysis and Information Retrieval. (1994) 161–175
[37] Lui, M., Baldwin, T.: langid. py: An off-the-shelf language identification tool. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012), Demo Session, Jeju, Republic of Korea. (2012)
[38] Preotiuc-Pietro, D., Samangooei, S., Cohn, T., Gibbins, N., Niranjan, M.: Trendminer: An architecture for real time analysis of social media text. In: Proceedings of the workshop on Real-Time Analysis and Mining of Social Streams. (2012)
[39] Mcnamee, P., Mayfield, J.: Character n-gram tokenization for european language text retrieval. Information Retrieval **7**(1) (2004) 73–97
[40] Foster, J., Çetinoglu, Ö., Wagner, J., Le Roux, J., Hogan, S., Nivre, J., Hogan, D., Van Genabith, J., et al.: #hardtoparse: POS Tagging and Parsing the Twitterverse. In: Proceedings of the AAAI Workshop On Analyzing Microtext. (2011) 20–25
[41] Sproat, R., Black, A., Chen, S., Kumar, S., Ostendorf, M., Richards, C.: Normalization of non-standard words. Computer Speech & Language **15**(3) (2001) 287–333
[42] Steinberger, R., Pouliquen, B., Kabadjov, M., Belyaeva, J., van der Goot, E.: JRC-Names: A freely available, highly multilingual named entity resource. In: Proceedings of the 8th International Conference in Recent Advances in Natural Language Processing. (2011) 104–110
[43] Han, B., Cook, P., Baldwin, T.: Automatically constructing a normalisation dictionary for microblogs. In: Proceedings of the conference on Empirical Methods in Natural Language Processing, ACL (2012) 421–432
[44] Derczynski, L., Maynard, D., Aswani, N., Bontcheva, K.: Microblog-Genre Noise and Impact on Semantic Annotation Accuracy. In: Proceedings of the 24th ACM Conference on Hypertext and Social Media, ACM (2013)
[45] Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The Penn Treebank. Computational Linguistics **19**(2) (1994) 313–330
[46] Hepple, M.: Independence and Commitment: Assumptions for Rapid Training and Execution of Rule-based Part-of-Speech Taggers. In: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics, Hong Kong (October 2000)
[47] Brill, E.: A simple rule-based part-of-speech tagger. In: Proceedings of the Third Conference on Applied Natural Language Processing, Trento, Italy (1992)
[48] Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology. NAACL '03 (2003) 173–180
[49] Forsyth, E., Martell, C.: Lexical and discourse analysis of online chat dialog. In: International Conference on Semantic Computing, IEEE (2007) 19–26
[50] Marcus, M., Santorini, B., Marcinkiewicz, M.: Building a Large Annotated Corpus of English: The Penn Treebank. Computational Linguistics **19**(2) (1993) 313–330
[51] Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., Smith, N.: Part-of-speech tagging for twitter: annotation, features, and experiments. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL (2011) 42–47
[52] Porter, M.: An algorithm for suffix stripping. Program **14**(3) (1980) 130–137
[53] Finkel, J., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics (2005) 363–370

[54] Sadilek, A., Kautz, H., Silenzio, V.: Modeling spread of disease from social interactions. In: International AAAI Conference on Weblogs and Social Media (ICWSM), AAAI (2012) 322–329
[55] Rout, D., Preotiuc-Pietro, D., Bontcheva, K., Cohn, T.: WhereâĂŹs @wally? a classification approach to geolocating users based on their social ties. In: Proceedings of the 24th ACM Conference on Hypertext and Social Media. (2013)
[56] Llorens, H., Derczynski, L., Gaizauskas, R.J., Saquete, E.: Timen: An open temporal expression normalisation resource. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC), Istanbul, Turkey. (2012) 3044–3051
[57] Sabou, M., Bontcheva, K., Scharl, A.: Crowdsourcing research opportunities: Lessons from natural language processing. In: 12th International Conference on Knowledge Management and Knowledge Technologies (I-KNOW). (2012) 17–24