High-throughput identification of chemistry in life science texts.

Peter Corbett[1] and Peter Murray-Rust[1]

[1] Unilever center for Moleclular Sciences Informatics, Lensfield Road, Cambridge, CB2 1EW.

Abstract.. OSCAR3 is an open extensible system for the automated annotation of chemistry in scientific articles, which can process thousands of articles per hour. This XML annotation supports applications such as interactive browsing and chemically-aware searching, and has been designed for integration with larger text-analysis systems. We report its application to the high-throughput analysis of the small-molecule chemistry content of texts in life sciences, such as PubMed abstracts.

1 Introduction

Chemical knowledge is an important component of bio-literature, as show by the recent prominence of ontolgies and resources on bio-informatics sites. ChEBI[1] (at the European Bioinformatics Institute) lists 7592 compounds of relevance to bioscience. The NCBI has recently provided PubChem as the repository of chemical data created in the NIH's Molecular Libraries Roadmap. Its importance has been shown by the addition of the ZINC database of commercially available compounds, bringing PubChem's size to about 5 million compounds and covering almost all those in common use.

Currently few publishers actively add chemical semantics to their products. Recently Nature Chemical Biology has produced connection tables (the formal computer representation of the structure) and made them available to PubChem. For almost all articles, however, the identity of the chemistry in the text has had to be extracted by human experts.

Natural language processing (NLP) of the biological, biochemical and biomedical literature is a well-developed area with a lot of activity. There is considerable activity from both a number of commercial entities (eg. Temis, Linguamatics, PubGene) and academic projects (eg. PennBioIE[2], FlySlip[3], GENIA[4]). There are a number of freely-usable web tools based on NLP technology (eg. MEDIE[5], Info-Pubmed[6], iHOP[7], EBIMed[8], Textpresso[9]). The biological NLP community is assisted by a variety of publically-available resources. Hand-annotated corpora have been made available by the PennBioIE and GENIA groups. Large amounts of the literature are available in the form of PubMed/MEDLINE, which is commonly used as a corpus,

M.R. Berthold, R. Glen, and I. Fischer (Eds.): CompLife 2006, LNBI 4216, pp. 107-118 2006.

for example by the web tools mentioned above, and other resources such as the Gene Ontology and other members of the Open Biomedical Ontologies family (which includes ChEBI). Finally, the field is assisted by competitive evaluations like BioCreAtIvE[10] and the TREC genomics track[11].

Development of natural language methodologies for chemistry lags behind that of the biochemical world. We have been able to find several works in the area[12-20], but not the same level of activity as is observed in the bioscience.

In this paper we describe an open source system, OSCAR3 which can identify much of the chemical terminology in articles, and extract molecular connection tables with useful recall and precision. OSCAR3 is being developed as a part of the SciBorg project which aims to use deep parsing of text (described in [21]) to analyse chemistry papers. Here we present the architecture and strategy of OSCAR3 as an open extensible modular framework for individual and high-throughput chemical text processing. The source code to OSCAR3 is on sourceforge.net[22].

1.1 Chemical language in bioscience

Among the useful information in manuscripts is
1. mention of chemical compounds.
2. details of synthesis (in vivo and in vitro) of compounds.
3. proof of structure (spectra and analytical data).
4. methods and reagents in bioscience bio-protocols.
5. properties of compounds.
6. reactions and their properties, both in enzymes and enzyme-free systems.

We consider both abstracts and full-text articles (a typical example on "Proton-sensing G-protein-coupled receptors" [23], (reproduced with permission of Nature))

Buffers and pH Experiments were carried out in a physiological salt solution (PSS) containing 130 mM NaCl, 0.9 mM NaH2PO4, 5.4 mM KCl, 0.8 mM MgSO4, 1.0 mM CaCl2, 25 mM glucose. This solution was buffered either with HEPES alone (20 mM) or HEPES/EPPS/MES (8 mM each; HEM-PSS), to cover a wider pH range. HEPES-buffered PSS was used in all experiments unless HEM-PSS is specifically mentioned. HEPES is 4-(2-hydroxyethyl) piperazine-1-ethanesulphonic acid…

This type of chemistry is very well understood and has a simple generic vocabulary which has not changed over decades. Unlike much bioscience, where ontological tools are an essential part of reconciling the domain-dependent approaches, much chemistry has an implicitly agreed abstract description. The problems are primarily reconciling syntax and semantics. This is because chemists use abbreviated methods of communicating data, relying on trained readers to add information from the context. We have reviewed current problems of machine-understanding of chemistry in a typical chemistry journal[24] and requirements for parsing chemistry in life sciences [25].

2 OSCAR3

OSCAR3 is being developed as part of the SciBorg system for the deep parsing and analysis of scientific texts [21] and Fig 1. shows its role. Scientific articles in bespoke XML are converted to a canonical XML ("SciXML", a schema developed by the Cambridge natural language group). A selection of modules (including OSCAR3) then annotate various concepts in the text. These annotations are collected together in a standoff annotation document – a separate document that contains pointers to elements in the source text, allowing the integration of the results from a range of different parsers, and for earlier parsers to pass information to later parsers. The role of OSCAR3 in the analysis of full text has informed many of our design and prioritisation decisions. For example, it will be easier for the SciBorg system to deal with false positives generated by OSCAR3 than with false negatives. Also, it is important for OSCAR3 to be able to recognise as many entities as possible, even if it cannot assign any semantics to many of them.

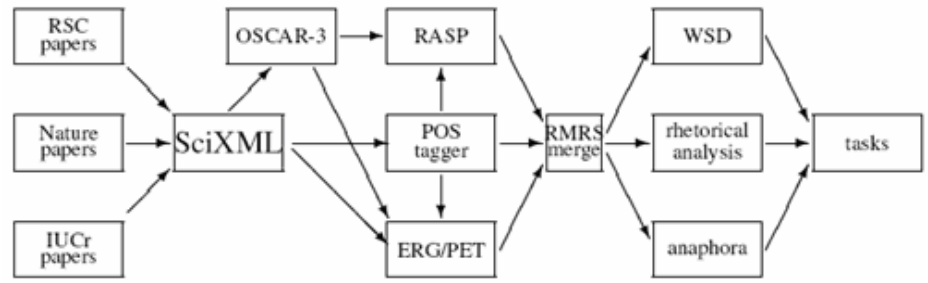

Fig. 1. Architecture of SciBorg. POS = Part Of Speech, RASP and ERG/PET are parsers for English text, RMRS is a format for representing the semantics of parsed language, WSD = Word Sense Disambiguation.

However, OSCAR3 can also be used in contexts other than the SciBorg framework. For example, we have been developing a set of web tools for analysing, viewing and searching texts using OSCAR3 in an essentially standalone manner. OSCAR3 could also potentially be incorporated into a different framework – for example, integrating OSCAR3 with biological NLP tools such as those mentioned in the introduction. To expedite this, we have been developing OSCAR3 with thought for modification, extensibility and integratability, making components modular and supporting them with XML data files. For example, it should be possible to configure OSCAR3 to recognise many GO terms by simple lookup, to recongise names of computation chemistry codes and to configure OPSIN to parse many semi-systematic names without editing the OSCAR3 source code.

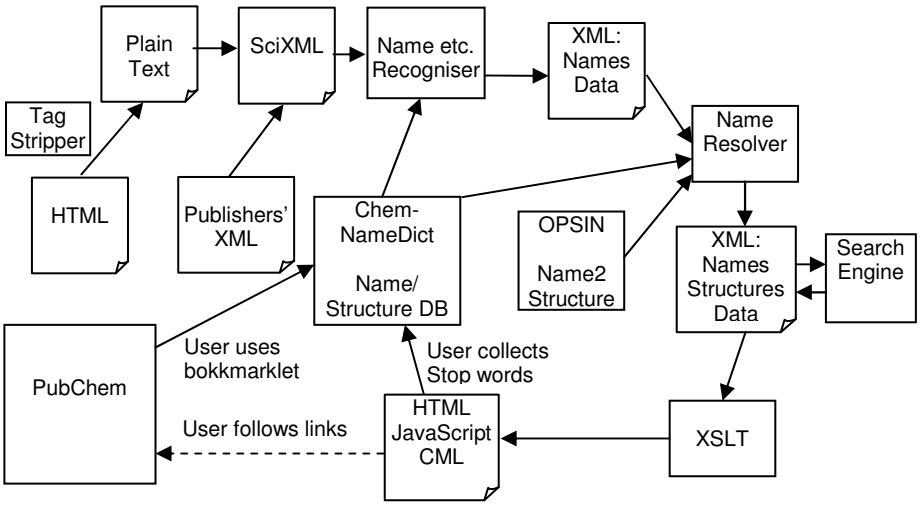

Fig. 2. Architecture of OSCAR3. Boxes represent OSCAR3 modules (except for PubChem), boxes with turned corners represent documents at various stages of processing. See below for further details.

The overall architecture of OSCAR3 is shown in Figure 2. The preferred input format is SciXML, but rudimentary facilities exist for producing SciXML from plain text or HTML, allowing documents to be fed to OSCAR3 from the world-wide web via a Javascript bookmarklet. Abstracts may also be fetched from PubMed. An initial module – the recogniser - finds chemical names, data and other entities in the text. The names can be systematic (eg. *propan-2-ol*), trivial (*morphine, water*), semisystematic (*diacetylmorphine*), acronyms and other abbreviations, (*DMSO, 5-HT*) or formulae ($C_6H_{12}O_6$, *EtOAc*). We also consider names of groups or other fragments (*methyl*) and names in plural, verb or adjective form (*azidomethanones*, *demethylation, pyrazolic*). In OSCAR3, "chemical" is intended to include simple polymers as well as small molecules, but not complex biopolymers such as proteins and nucleic acids. Elements are listed as a separate class (as it is often nontrivial to determine whether "carbon" (for example) refers to carbon atoms within a molecule or to the pure substance). "Data" refers to conventional representations of experimental results – e.g. "[α]22D +10.0 (c 1.00, MeOH)", which represents the optical rotation of a substance. "Other entities" are other nonwords that occur in chemical free text, for example "C(1)-N(6)" which refers to a particular bond in a molecule.

A second module – the resolver - examines the chemical names and attempts to assign structures to them. Two forms of output are produced – a standoff annotation document (for the rest of the SciBorg framework), and an enhanced SciXML document that incorporates the annotations inline while preserving all of the formatting, bibliographic, metadata and other markup that was in the source document.

Prior to name recognition, the text must be tokenised, by splitting on whitespace and recognising full stops and other similar characters. A particular difficulty is posed by hyphens, as these sometimes occur within chemical names (eg *tert-butyl peroxide*) and sometimes divide names from other words (eg "*hexane*-*ethyl acetate*" is a common phrase used to describe a mixture of the two solvents). Zamora and Blower [17] only considered hyphens with two alphabetic characters on either side to be word boundaries; we also have lists of strings that denote non-word-boundaries if they come before the hyphen (eg. *tert-* is often a part of chemical names), and strings that denote word boundaries if they occur after the hyphen (eg –containing, -induced).

To cope with the variety of forms of names and data to be recognised, several methods are used in parallel for name recognition. OSCAR3 keeps an internal lexicon of chemical names and structures – we have initially populated this using ChEBI[1] This collection is by no means comprehensive, but it does cover many key solvents, reagents and biomolecules. This lexicon can be extended at run-time.

Names are also recognised using a naïve-Bayesian method based on overlapping 4-Grams[26]. Wilbur et al. [13] have reported the use of a simple 4-Gram based approach to recognise chemical names to obtain high precision and recall on their test data. However, Vasserman [12] found that their simple approach gave poor results in the more difficult context of PubMed abstracts, and experimented with a variety of smoothing algorithms for the 4-Gram models to improve the performance of their classifier. Neither approach has been used to find the bounds of chemical names in free text. Here, we use modified Knesser-Ney smoothing[27] to produce a refined 4-Gram model. Further improvements are obtained by applying a penalty to the scores of words in a standard English dictionary, and by rejecting names that did not possess a suffix from a list of 49 common chemical suffixes. A lexicon of stop words exists to catch the most common errors. There are rules for which words to group together to make multiword chemical names – for example a group name can add onto the front of a chemical name, as in *ethyl acetate*. Finally, a set of cascaded regular expressions is used to recognise chemical data[28], chemical formulae and other forms of notation.

Structures are assigned to chemical names *via* two methods – lookup (using the lexicon above), and parsing of systematic nomenclature (see below). The structures are stored as SMILES, InChIs (International Chemical Identifier – an algorithmically-generated, canonical identifier for chemical compounds developed by IUPAC) and CML (Chemical Markup Language).

After parsing, the enhanced SciXML can then be rendered into HTML using an XSLT stylesheet, and displayed in a browser. Javascript routines in the HTML are used to feed the molecular structures back to the server when the mouse is moved over them to produce and display a structural diagram. The pages are indexed by a search engine based on Apache Lucene, which indexes the compounds by their InChI identifiers, allowing them to be searched for specific compounds. The search engine also contains the full structures of all of the compounds indexed; these may be queried using substructure and similarity searches. This produces a list of InChIs

which are then used to retrieve documents relevant to the query. As well as retrieving the documents, it is possible to list all of the compounds occurring in the documents, sorted by their frequencies of occurrence.

The browser-based architecture also provides a useful way of retraining OSCAR3. If a name does not have an associated structure, the structure (and synonyms) may be fetched from PubChem. We have not incorporated PubChem into OSCAR3 wholesale, partly out of a concern for efficiency, and partly as not all of the names in PubChem are accurate (see CIDs 26, 311) or appropriate (see CID 446220). Likewise, if a word has been misclassified as a chemical name, it can be fed to a list of the list of stop words, so that future occurrences of the word will not be classed as chemical. The stop words are also used to retrain the Bayesian classifier, further improving its accuracy. These techniques are especially useful in conjunction with the sorted compound lists described above, as this allows for common compounds and errors to be identified and dealt with appropriately, reducing the amount of effort required to adapt OSCAR3 to new domains.

2.1 Chemical names

The conversion of systematic chemical names to structures is a surprisingly difficult task: the official description[29] of the nomenclature is not presented in a formal way, and so it falls to software authors to convert the descriptions provided into working code. This is not a trivial problem and the various commercial offerings on the market compete on the range of names that they can parse.

It is clear that OSCAR requires a chemical name parser, and there are a number of reasons for us to write our own. The first is a sheer practicality – an open-source parser made in-house is easier to integrate with other systems, and it is legally easier to deploy and distribute the results. More importantly, having such a parser will allow ourselves and others to experiment with parsing many of the ways nomenclature is used in the chemical literature. We have therefore developed OPSIN – an Open Parser for Systematic IUPAC Nomenclature.

A number of parsing schemes have been described in the literature. The earliest approach we are aware of was reported by Vander Stouw et al.[30-31]. Later, Kirby et al.[30] fitted names to a formal context-free grammar using a modified SLR parser. In building a commercial parser for CambridgeSoft, Brecher expressed a frustration with the formal grammars, and appears to have chosen a less formal approach.[32]

The published descriptions of these parsers are not sufficiently detailed to allow a reconstruction of their work: for example Vander Stouw [30] reports the use of a “special routine” to disambiguate the possible meanings of the token “hex”, with little further explanation. These omissions are at least partly due to a lack of space in which to present full details; to overcome this, we have deposited the source code for OPSIN on sourceforge.net under an open-source license, along with the rest of OSCAR3.

Previously, we have worked on an ad-hoc approach to nomenclature parsing. However, this ran into severe difficulties owing to ambiguity. Therefore, we are currently pursuing a hybrid approach. Rather than using context-free grammars, we chose to partially interpret the names using finite-state grammars, which are less expressive but more tractable. A series of informal rules is then used to construct a full interpretation, thus escaping the limitations of the finite-state parsing.

The input to the parser is a complete systematic name: the current parser cannot detect and remove extraneous verbiage (e.g. *glacial* in *glacial* acetic acid) from its input, nor does it request additional information if it is only given a partial name to work with. There are a number of stages of processing – key ones are shown below.

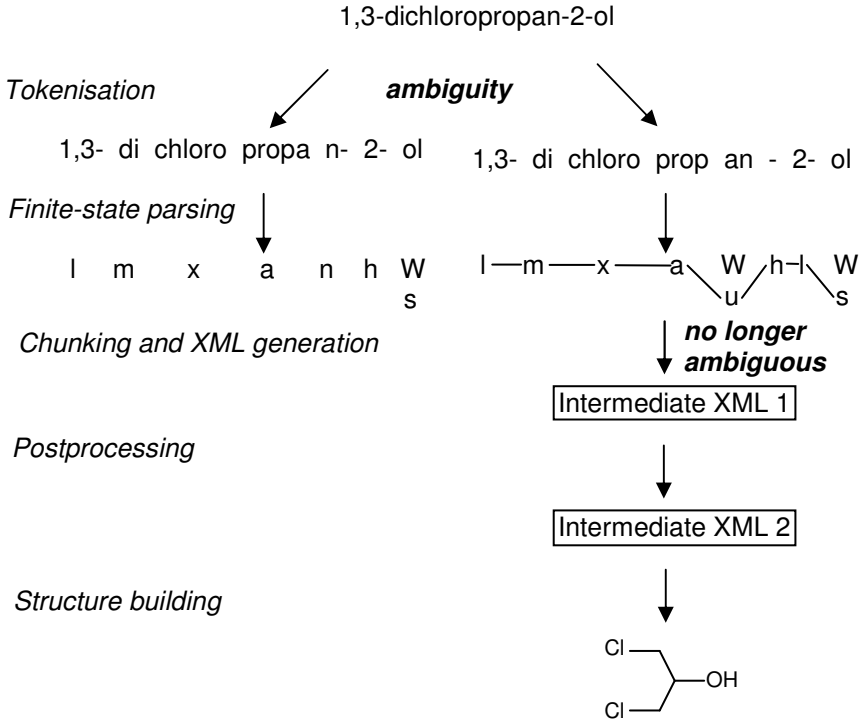

Fig. 3. Steps in the parsing of chemical names by OPSIN. Key to "finite-state parsing": l = locant group, m = multiplier, x = simple substituent, a = alkane name stem, n = "n-" (indicating a straight-chain alkane), h = hyphen, s = suffix, W = Hantzsch-Widman ring system.

The first step (not shown in Figure 3) breaks the chemical name into its constituent words, by splitting on spaces, and assigns those words to particular roles. A set of regular expressions are used to detect whether the compound is represented as a one-word compound (eg. ethanol), an ester (ethyl acetate), an acid (ethanoic acid), a salt (sodium acetate) or one of a number of other possibilities. Having done this, the

various words are labelled as being root-like (eg “ethanol”), substituent-like (eg “ethyl”) or a simple token (“acid”).

The second stage (*tokenisation*) breaks the individual words into a list of multi-character tokens. Tokens come from two sources – a set of lists of tokens, along with data on what they mean, and a set of regular expressions.

In the third stage, *finite-state parsing*, roles are assigned to the various tokens – some, such as “chloro”, are unambiguous. Others, such as “hex” and “ane”, could have multiple roles – each of these are represented by a 1-letter symbolic code. The first could act as a multiplier (as in hexachloro-, “m” in Figure 3) or to indicate six carbon atoms in a row (as in hexane), whereas the latter could indicate an alkane (hexane again, “a” in Figure 3) or a six-membered aliphatic ring (as in dioxane, “W” in Figure 3). The various possibilities for each token in the word are tabulated, and a finite automaton generated from a regular expression (representing the grammar of chemical names) is used to select a valid path (if present) through the word. At this stage, an essentially ‘flat’ parse is generated – further processing is needed to construct a full parse tree.

The next stage is to group together the tokens into chunks, each representing the “root group” of the compound, or a substituent. These chunks also include metadata such as brackets and locants and multipliers.

Once the chunk structure of the name has been determined, an XML representation of the name is constructed, with one XML element per token in the chemical name. Note that at this stage the XML marks-up the name that OPSIN has been given – removing the tags would result in the recovery of the original name.

A number of aspects of chemical grammar which do not fit into the finite-state formalism are now processed. The effects of multipliers (di-, tri- etc.) are applied by duplicating the groups that they refer to, locant groups (eg. 1,2-) are parsed into their individual locants (1, 2) and matched with the elements that they apply to, and so on. Also at this stage, OPSIN resolves the matter of which groups attach to which, by looking at the bracketing. For example, in (2-chloroethyl)benzene the chloro- attaches to the ethyl, whereas in 2-chloro-1-ethylbenzene the chloro- attaches to the benzene.

Next, the chemical structure is built, according to the information provided by the XML. In each chunk, a group is specified and a connection table is built for that group. The group is then modified, for example by adding –OH to it if the suffix –ol is present, and attached to other groups that have been constructed. A validation step then occurs, that sanity-checks the results of the parse, and rejects structures with obvious problems, such as pentavalent carbon atoms.

Finally, the connection table that has been made is converted to CML, which is returned to the application that called OPSIN.

At many stages, OPSIN encounters ambiguities where more than one interpretation is possible. Where it is efficient to do this, all possible interpretations are considered in the later stages of processing, in the hope that all but one of these will fail to produce a structure. Cases where more than one structure is produced are treated as parsing failures.

OPSIN has been tested against a set of machine-generated IUPAC names. The entries for PubChem compounds with IDs from 1 to 10,000 were collected, and the IUPAC names (designated by PUBCHEM_IUPAC_NAME in the .sdf files) were harvested, along with the associated InChIs. This gave 8183 names (average length: 58 characters) with associated InChI pairs. The names were passed to OPSIN, and where OPSIN produced a structure for the name, the structure was converted to an InChI and compared against the published InChI. From the 8183 names, OPSIN produced 4475 correct InChIs (54.7%), 162 incorrect InChIs (2.0%) and did not produce a result for the remaining 3546 of the names (43.3%). This test run took 442.5 seconds – equal to 18 parse attempts per second.

There are a number of areas where OPSIN is currently lacking in functionality. One of these is in the handling of "generic" nomenclature, where the chemical structure is underspecified by the name – for example in "aminopyrazoles" where the amino group attaches to the pyrazole is not specified. Often, these appear as plurals, and are used to specify a class of chemical compounds. Currently, OPSIN will reject plurals that are handed to it, but if they are re-cast as singulars by removing the terminal 's', OPSIN will attempt to parse them. Often, where these names differ from fully-specified names in that locants are missing, the current OPSIN will assume that "default" locants are meant – for example aminopyrazole is treated as 1-aminopyrazole. It should be noted that to the best of our knowledge none of the other parsers are able to produce underspecified structures from chemical names either. Parsing these remains as one of our research objectives.

Features not currently implemented include the handling of tri- and higher polycyclic nomenclature (eg. tricyclo[5.4.0.$0^{2,9}$]undecane), fused aromatic nomenclature (eg. thieno[3,2-b]furan – however fused systems with trivial names such as naphthalene are included, as is the common system cyclopenta[a]phenanthrene, which is included as a special case, as it has a special locant numbering for compatibility with steroid nomenclature), and many forms of nomenclature specific to complex inorganic chemicals and natural products. Another problem is that the bracketing that is used to show which group attaches to which is frequently left out – for example, in 2-chloroethylbenzene it is not trivial to determine whether the chloro group attaches to the ethyl group or to the benzene. We are therefore looking into ways in which this information can be inferred. Possible approaches involve looking for common fragments of chemical names (eg. chloromethyl-, dimethylamino-), or by seeing which interpretation requires the fewest assumptions later on (for example if the chloro group above attaches directly to the benzene, the ethyl group has three inequivalent places where it could attach. If the chloro group attaches via the ethyl group, the symmetry of the benzene ring removes this problem).

3 Evaluation

An initial informal evaluation was carried out the 1100 PubMed abstracts in the section of the BioIE[2] corpus dealing with cytochrome P450 biochemistry, suggesting an average precision of around 75%. These took about 1300 seconds to parse. After further development OSCAR3 was re-tested on unseen abstracts.

Batches of ca. 200 abstracts were fetched from PubMed using five different search terms. “Smith” was used to get an essentially random collection of abstracts. “Bacterial metabolism” was used to represent a fairly large field, “veterinary toxicology” for a small one. “Porcine skin” and “grapefruit” were also selected for their ability to give lots of interesting abstracts.

From them, abstracts containing five or more chemicals (or other entities to be annotated by OSCAR3) were selected. For each batch, the following procedure was followed: the abstracts were auto-annotated by OSCAR3, and a gold standard was created by correcting the annotations by hand. Per-abstract precision and recall (“P1”, “R1”) were calculated for the concepts annotated. Two annotations matched if they were in the same place, the same length and had the same type (compound, group, element etc.). Currently the name resolution is not advanced enough for formal testing. A second, larger, batch of abstracts (ca 500 where available, not overlapping with the first) was fetched using the same search term, and auto-annotated with OSCAR3. The most commonly occurring compounds were then tabulated, and exactly five minutes was spent adding mis-recognised words (eg. *swine, prostate*) as stop-words, starting with the most common mistakes and working down the list in sequence. The first batch was re-annotated with OSCAR3, and precision and recall (“P2”, “R2”) were recalculated. The parser was then retrained, and precision and recall (“P3”, “R3”) were recalculated again. Finally, exactly five minutes was spent selecting various abbreviations that had been mis-recognised as formulae (eg. IC50, P450, H5N1, CI), and final precision and recall (“P4”, “R4”) were calculated. This procedure demonstrated that the second batch of abstracts could act as training data for the first, without the need for laborious hand-annotation of the entire batch.

Table 1. Results from evaluation of OSCAR3. P1 etc. as defined above.

Search term	Smith	Veterinary Toxicology	Bacterial Metabolism	Porcine Skin	Grapefruit
Abstracts fetched	168	91	185	196	180
Abstracts selected	48	31	63	64	129
Abstracts in batch 2	432	147	483	483	234
P1	60.8%	67.5%	67.7%	72.1%	65.0%
R1	69.6%	74.0%	80.8%	69.7%	72.0%
P2	62.9%	70.5%	69.9%	74.0%	66.7%
R2	69.3%	74.0%	80.8%	69.8%	72.0%
P3	63.0%	71.4%	70.5%	73.7%	66.9%
R3	69.3%	74.0%	80.8%	69.2%	72.0%
P4	64.1%	74.3%	71.6%	75.3%	70.5%
R4	69.6%	74.0%	80.8%	69.1%	72.0%

From this work, it is clear that there are a number of areas in which OSCAR3 could be improved. Some changes should be trivial to implement – for example the list of allowable chemical suffixes was missing a few key entries (*-am*, *-fil*, *-id*, *-vir*, *-oids*) that were common in drug names. More training data would also help.

Other changes are not so straightforward. Acronyms and other abbreviations, for example, are a real problem. The regex-based chemical formula recogniser mis-classifies many of these as chemicals (some fortuitously do represent chemicals eg. BP (*benzopyrene*), but most represent non-chemicals eg. HIV), and many acronyms/abbreviations for chemical names are missed by the parser. Fortunately, many of these acronyms are defined in the text of the abstract – in the SciBorg framework, we see finding these and using them to handle acronyms appropriately as being a task for word sense disambiguation (or anaphora/coreference resolution) modules. Likewise, words like In, No and At are commonly mis-labelled as chemical elements – these errors could be removed with the aid of a part-of-speech tagger.

Currently OSCAR3 uses a medium-size lookup for names, but we believe that almost all common trivial and drug names are to be found in PubChem whose name, synonym and connection table data are freely downloadable and could be configured with OSCAR3.

References

1. de Matos, P., Ennis, M., Guedj, M., Degtyarenko, K., Apweiler, R. ChEBI – Chemical Entities of Biological Interest, Nucleic Acids Res., Database Summary Paper 646.
2. http://bioie.ldc.upenn.edu
3. http://www.cl.cam.ac.uk/users/av308/Project_Index/index.html
4. http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA
5. http://www-tsujii.is.s.u-tokyo.ac.jp/medie
6. http://www-tsujii.is.s.u-tokyo.ac.jp/info-pubmed
7. http://www.ihop-net.org/UniPub/iHOP/
8. http://www.textpresso.org/
9. http://www.ebi.ac.uk/Rebholz-srv/ebimed/index.jsp
10. http://pdg.cnb.uam.es/BioLINK/BioCreative.eval.html
11. http://ir.ohsu.edu/genomics/
12. Vasserman, A.: Identifying Chemical Names in Biomedical Text: An Investigation of the Substring Co-occurrence Based Approaches. Proceedings of the Student Research Workshop at HLT-NAACL, 2004.
13. Wilbur, J. W., Hazard, G. F., Divita, G., Mork, J. G., Aronson, A. R., Browne, A. C.: Analysis of Biomedical Text for Chemical Names: A Comparison of Three Methods. Proc AMIA Symp 1999, 176-80.
14. Chowdhury, G. G., Lynch, M. F., Semantic Interpretation of the Texts of Chemical Patent Abstracts. 1. Lexical Analysis and Categorization. Journal of Chemical Informatics and Computer Science 32 (1992) 463-467.

15. Chowdhury, G. G., Lynch, M. F., Semantic Interpretation of the Texts of Chemical Patent Abstracts. 2. Processing and Results. Journal of Chemical Informatics and Computer Science 32 (1992) 468-473.
16. Al, C. S., Blower, P. E. Jr., Ledwith, R. H., Extraction of Chemical Reaction Information from Primary Journal Text. Journal of Chemical Informatics and Computer Science 30 (1990), 163-169.
17. Zamora, E. M., Blower, P. E. Jr.: Extraction of Chemical Reaction Information from Primary Journal Text Using Computational Linguistics Techniques. 1. Lexical and Syntactic Phases. Journal of Chemical Informatics and Computer Science 24 (1984), 176-181.
18. Zamora, E. M., Blower, P. E. Jr.: Extraction of Chemical Reaction Information from Primary Journal Text Using Computational Linguistics Techniques. 2. Semantic Phase. Journal of Chemical Informatics and Computer Science 24 (1984), 181-188.
19. Postma, G. J., van der Linden, B., Smits, J. R., Kateman, G.: TICA: A System for the Extraction of Data from Analytical Chemical Text. Chemometrics and Intellegent Laboratory Systems, 9 (1990) 65-74.
20. Cooper, J. W., Boyer, S., Nevidomsky, A., Coden, A. R.: Automatic discovery and annotation of organic chemical names in patents, 229th ACS National Meeting 2005.
21. Copestake, A., Corbett, P. T., Murray-Rust, P., Rupp, C. J., Siddharthan, A., Teufel, S., Waldron, B.: An Architecture for Language Technology for Processing Scientific Texts, submitted for UK e-Science All Hands Meeting 2006.
22. http://sourceforge.net/projects/oscar3-chem.
23. Ludwig, M.-G., Vanek, M., Guerini, D., Gasser, J. A., Jones, C. E., Junker, U., Hofstetter, H., Wolf, R. M., Seuwen, K.: Proton-sensing G-protein-coupled receptors, Nature 425 (2003), 93-98.
24. Murray-Rust, P., Mitchell, J. B. O., Rzepa, H. S.: Communication and re-use of chemical information in bioscience, BMC Bioinformatics 6 (2005), 180.
25. Murray-Rust, P., Mitchell, J. B. O., Rzepa, H. S.: Chemistry in Bioinformatics, BMC Bioinformatics 6 (2005), 141.
26. Townsend, J., Copestake, A., Murray-Rust, P., Teufel, S., Waudby, C., Language Technology for Processing Chemistry Publications, in Proceedings of the fourth UK e-Science All Hands Meeting, 2005.
27. Chen, S. F., Goodman, J.: An empirical study of smoothing techniques for language modeling. Computer Speech and Language 13 (1999), 359-394.
28. Townsend, J. A., Adams, S. E., Waudby, C. A., de Souza, V. K., Goodman, J. M., Murray-Rust, P.: Chemical documents: machine understanding and automated information extraction, Organic & Biomolecular Chemistry 2 (2004), 3294.
29. A Guide to IUPAC Nomenclature of Organic Chemistry, Recommendations 1993 (including Revisions, Published and hitherto Unpublished, to the 1979 Edition of Nomenclature of Organic Chemistry), IUPAC (1993)
30. Vander Stouw, G. G., Naznitsky, I., Rush, J. E.: Procedures for Converting Systematic Names of Organic Compounds into Atom-Bond Connection Tables. Journal of Chemical Documentation 7 (1967) 165-169.
31. Vander Stouw, G. G., Elliott, P. M., Isenbert, A. C.: Automated Conversion of Chemical Substance Names into Atom-Bond Connection Tables. Journal of Chemical Documentation 14 (1974), 185-193.
32. Cooke-Fox, D. I., Kirby, G. H., Rayner, J. D.: Computer Translation of IUPAC Systematic Organic Chemical Nomenclature. 1. Introduction and Background to a Grammar-Based Approach, J. Chem. Inf. Comp. Sci. 29 (1989) 101.
33. Brecher, J.: Name=Struct: A Practical Approach to the Sorry State of Real-Life Chemical Nomenclature, J. Chem. Inf. Comp. Sci. 39 (1999) 943.