

ASSOCIATING COLLOCATIONS WITH DICTIONARY SENSES

Abhilash Inumella
IIIT Hyderabad, India
abhilashi@students.iiit.ac.in

Adam Kilgarriff
Lexical Computing Ltd., UK
adam@lexmasterclass.com

Vojtěch Kovář
Masaryk Uni., Brno, Cz
xkovar3@fi.muni.cz

Abstract

We describe a project to assign the collocations for a word, as automatically identified from a large corpus, to its distinct senses. The goal is, in the short term, a new English collocations dictionary (Macmillan Collocation Dictionary, MCD) and in the long term, a ‘disambiguating dictionary’ supporting a range of language tasks. The project is fully corpus-based, and the core infrastructure is the Sketch Engine corpus query system. We have explored both automatic methods using Word Sense Disambiguation algorithms and a computer-supported approach which integrates two new technologies: GDEX (Good Dictionary Example finding) and TBL (TickBox Lexicography). As at summer 2009, the lexicography for MCD is proceeding apace using both of these innovative methods.

1. THE DREAM OF THE DISAMBIGUATING DICTIONARY

It is now commonplace to link a dictionary to electronic texts (in word processors, web browsers or other tools) so that, by clicking or hovering over a word, the user can see the entry for the word in the dictionary. Many publishers offer their dictionaries in this form. The basic task is easy: it is one of matching the string in the text to a headword in the dictionary. Publishers have more, or less, successful solutions to the associated issues of correctly identifying dictionary headwords for the inflected forms found in texts and of matching multi-word expressions found in the dictionary.

One thing they do not do is take the user to the correct sense of a polysemous word. This is desirable. The user would no longer need to read the whole entry and work out which sense was relevant. For long entries this can be a forbidding task, particularly for learners who are struggling with the language in the first place. If the dictionary is bilingual, then the correct sense becomes the correct translation: the user could be directly given an appropriate translation.

If the dictionary publisher had the ability to disambiguate in this way for the user, then they would also have the ability to disambiguate offline, and that would be a great boon for automatic translation and a range of other language technology applications including question answering, information retrieval and information extraction. They would have a disambiguating dictionary, and they would have solved the problem of Word Sense Disambiguation (WSD).

WSD has been a challenge for language technology researchers since the earliest days of the field (see Agirre and Edmonds 2006) for a wide-ranging review of the field and description of the state of the art). It remains painfully intractable, with all systems in the SENSEVAL and SEMEVAL competitions making errors in over a quarter of cases.¹ So the disambiguating dictionary (which does not make many, many errors) remains a dream.

But what steps might be made in that direction? In Kilgarriff (2005) we make the case that collocations provide a productive framework for thinking about the issue. Yarowsky (1993) put forward the ‘one sense per collocation’ hypothesis: to the extent that it is true, collocatesⁱⁱ serve to disambiguate. As a step towards disambiguating occurrences of words in running text, we can associate a word’s collocations with one or other of its senses. This will probably be an easier task than full WSD because a collocation is only a collocation if it is a reasonably common pattern of usage for a word, so we will be able to find multiple examples of it in a sufficiently large corpus, so:

- We will not be aiming to disambiguate one-off and anomalous uses

- We will always have multiple contexts of a collocation to use as input to any disambiguation algorithm
- Collocations are often given (implicitly or explicitly) in dictionary entries
- It is a bounded task: whereas a word can appear in any number of contexts, its collocations will count in the tens or possibly in the hundreds.

Also, WSD systems generally work through collocations: they aim to find collocations (as well as grammatical patterns and domains) associated with each sense of the word, and then use them as clues to disambiguate new instances. So, if our goal is just to disambiguate collocations, we are ‘doing WSD’ but stopping before we attempt the most difficult part.

1.1 Manual, Semi-automatic, Automatic

How then might we associate collocations with senses? There are three options: by hand, by computer, or half-and-half. To do it by hand is a very large undertaking: there are perhaps 10,000 polysemous words to be covered (Moon, 2000) and perhaps an average of twenty or thirty collocations to be assigned per sense. Fully automatic methods are possible but are likely to make many errors. Semi-automatic methods look promising, and have been tried in the WASPS project (Kilgarriff and Tugwell, 2002).

2. THE PROJECT

The project arose because a leading dictionary publisher, Macmillan, wished to develop their resources, in particular to develop a collocations dictionary (hereafter MCD). They were also aware that a partial solution to the disambiguating dictionary could lead in many other interesting directions. They were active in English Language Teaching, and had a range of ELT dictionaries including an advanced learners’ dictionary (MEDAL 2002, 2007) and several smaller titles based on it. The project was devised in consultation between Macmillan and Lexical Computing Ltd, a company with expertise as the intersection of lexicography, corpus linguistics and language technology.

MCD is to start from MEDAL (2007), and will provide a full account of the collocations of the core senses of around 4,500 common and highly ‘collocational’ (Kilgarriff, 2006) words. As in other collocations dictionaries such as Oxford’s (OCD 2002, 2009) (and also as in word sketches, see below) collocations will be organised according to grammatical relations. Some collocations will be illustrated with examples in the paper book; all will have examples available by mouse-click in online and other electronic versions.

2.1 Infrastructure: The Sketch Engine

From the outset, the project was to be corpus-driven. Lexical Computing Ltd has a corpus tool, the Sketch Engine (Kilgarriff et al 2004, <http://www.sketchengine.co.uk>) which would provide sophisticated corpus-handling to support the process. In particular, the Sketch Engine creates ‘word sketches’ – one page, corpus-driven accounts of a word’s grammatical and collocational behaviour. Word sketches have been in use in lexicography for ten years now and have received a number of positive reviews; a formal evaluation is underway. The word sketches identify the collocates that we want to assign to senses.

The Sketch Engine was used via its web API: its analyses were available even though the project was proceeding as a separate development in a different continent.

2.2 The Corpus

The output of a corpus tool is only as good as the corpus. The project required a large corpus, so that, even for words and their collocations which are not so common, we would have plenty of evidence. The language was English. The corpus we used is UKWaC (Ferraresi et al., 2008), a corpus of 1.5 billion words for English, drawn from the web, and carefully ‘cleaned’ (to remove

advertisements, navigation bars, copyright statements and other ‘boilerplate’ text) and de-duplicated, with all duplicate and near-duplicate documents removed (Pomikalek, 2008).ⁱⁱⁱ The corpus had already been tokenized, lemmatised and part-of-speech-tagged using the leading tool TreeTagger (Schmid 1994, <http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/>).

3. AUTOMATIC APPROACHES

3.1 Thesaurus method

The thesaurus consists of groups of semantically close word senses. All word senses are defined in the dictionary with each member having a unique ID; this ID is used to refer to them in the thesaurus. On average there are 28 members per class and a total of 10,2531 members are covered in 3,664 thesaurus classes. An example thesaurus class with six members is shown in Figure 1.

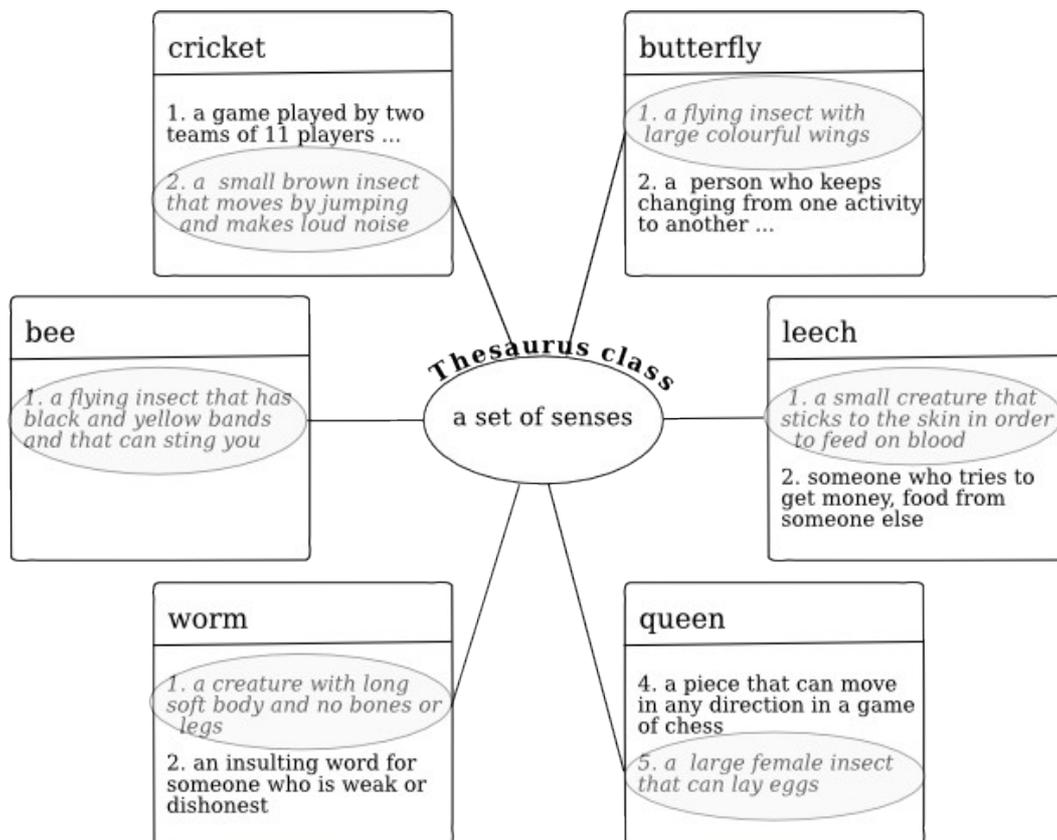


Figure 1: The thesaurus-to-dictionary linking is depicted in the figure. It shows a thesaurus entry with six members (senses) viz.: 2nd sense of cricket, 1st sense of butterfly, 1st sense of leech, 5th sense of queen, 1st sense of worm, 1st sense of bee.

The method works on the premise that a sense shares its collocates with its thesaurus class members. The basic intuition has been used by a number of other WSD researchers: Yarowsky (1992) used Roget’s thesaurus, and Agirre and Martinez (2004) use WordNet and large datasets from the web for finding shared collocates. The method is:

Algorithm 1: Thesaurus method

1. Input: a sense s
2. Output: collocates of sense s
3. Get all the thesaurus classes containing s. Let T be the set of all the members of all these classes
4. Get collocates of all the members of T using Sketch Engine

5. Assert a collocate if it occurs as collocate of *s* and at least 2 other members of *T*.

The method combines two resources, the Sketch Engine and the manually crafted thesaurus. The former is used to find the collocations, the latter to assign them. In the Sketch Engine, collocates are lemmas (rather than word forms) standing in a particular grammatical relation such as *subject*, *object*, *modifier* to the headword. This is an advantage for the current method as the lemmatisation makes useful generalisations, and the grammatical constraints increase the specificity of the collocates, thereby reducing noise and eliminating the need for complex scoring mechanisms.

Let us present a sample run of the method. Consider sense 5 of the word *queen* (see Figure 1). In step 3 we find that the thesaurus class members are $T = \{cricket, butterfly, leech, worm, bee\}$. In step 4 the collocates of *queen* and all the members of *T* are obtained using the Sketch Engine. In step 5 the collocates common to *queen* and two other members of set *T* are identified. A few confirmed collocates are shown in Table 1.

Table 1: Associating collocations to the 5th sense of *queen*: sample output.

Collocation	Supporting Thesaurus Class Members
<queen, young, a_modifier>	{queen, bee, cricket}
<queen, fly, subject_of>	{queen, bee, butterfly}
<queen, lay, subject_of>	{queen, butterfly, leech, worm}
<queen, feed, object_of>	{queen, worm, bee}
<queen, breed object_of>	{queen, worm, butterfly}
<queen, become, object_of>	{queen, worm, butterfly}

While the method produced some promising results, it also made many errors, for reasons including the following:

1. The thesaurus was not ideal for our purposes because many word senses were in the thesaurus class for several senses of a word. Such thesaurus class members are of limited use for disambiguation.
2. Word sense frequencies are typically highly skewed (Kilgarriff, 2004; McCarthy et al., 2004). Then, depending on the skew and the nature of the senses, most of the collocates often relate to the dominant sense. This is an issue for the ‘bee’ sense of *queen* as it is rare.
3. Thesaurus class members are often themselves ambiguous, and the relevant sense will often not be the dominant sense. Then there are unlikely to be many collocates for the relevant sense in the word sketch. For example, the dominant sense for *cricket* is the sport, and the collocates largely relate to the sport sense.
4. Thesaurus classes in MEDAL are not constrained to be words that are distributionally similar to each other. Thus, staying with the *cricket* (sense 1) example, the thesaurus class is {cricket, ashes, bat, lbw, out, misfield, duck, Lord's, googly, ...}. Though the thesaurus class members are all related to the game of cricket they do not share collocates. They belong to different semantic classes, viz *cricket* is a game, *Lord's* is a place, *ashes* is a tournament etc.
5. Collocates of thesaurus class members sometimes occurs with the head word where the headword is not in the relevant sense. For example, the collocation <queen, become, object_of>^{iv} is associated with the 5th sense of *queen* since it is also a collocation of *worm*

and *butterfly*. This is incorrect both linguistically and zoologically: linguistically because <queen, become, object_of> is usually the first ‘human royalty’ sense of *queen*, and zoologically because queen bees get special treatment from birth so non-queens never *become* queens!

6. Some collocates occur with most of the words: For example, the collocation <queen, see, object_of> is valid with all the senses of queen and it is also a collocation of all the thesaurus class members.

We would like to model the frequency distribution of senses. It is very often the highest-scoring WSD strategy to assign all instances to the commonest sense (McCarthy et al., 2004). In recent work we are modelling frequency with the indirect evidence that

- (1) MEDAL senses tend to be in frequency order, commonest first, and
- (2) The frequency of a sense tends to vary with entry length.

To use this model, we need the thesaurus algorithm to output a score instead of a Boolean. This is work in progress.

3.2 Dictionary-parsing

Each sense of a word in MEDAL is provided with definition and examples. There are in all 84,799 example sentences containing a very large number of collocates, particularly since example sentences are selected in order to show collocates (alongside other constraints). To extract collocates we parsed the example sentences.

We used the Stanford dependency parser (Klein et al., 2003). It outputs the dependency structure: a tree where each edge denotes a labelled dependency between two lexical items, ‘edge-parent’ and ‘edge-child’. In Figure 2 the leftmost edge is labelled *nsubj* (edge-label) and it is between *analyse* (edge-parent) and *scientists* (edge-child).

Once the dependency structure is obtained, matching rules are employed to match the edges in the tree and then a collocation generation rule is employed to generate the collocation. For every matching rule there is an associated collocation generation rule. The matching and generation rules have the following structure:

Edge Matching rule: <edgeLabelName, 0/1>
Collocation Generation rule: <gramrel, 0/1>

An edge matching rule can be read as follows: Match an edge only if it has the edge-label edgeLabelName and the headword occurs as the edge-child or edge-parent corresponding to 0 or 1 respectively. Once the edge is matched the following generation rule is employed to generate the collocation.

A collocation generation rule can be read as follows: Generate a collocation with the grammatical relation gramrel and the edge-child/edge-parent as modifier corresponding to 0 or 1 respectively.

Example:

Edge matching rule: <subject_of, 1>
Collocation generation rule: <subject, 0>

The above rule can be read as follows,

If:

there is an edge label subject_of in dependency tree and
the senseHead occurs as edge -parent

then generate collocation with:

subject as grammatical relation and
edge-child as modifier

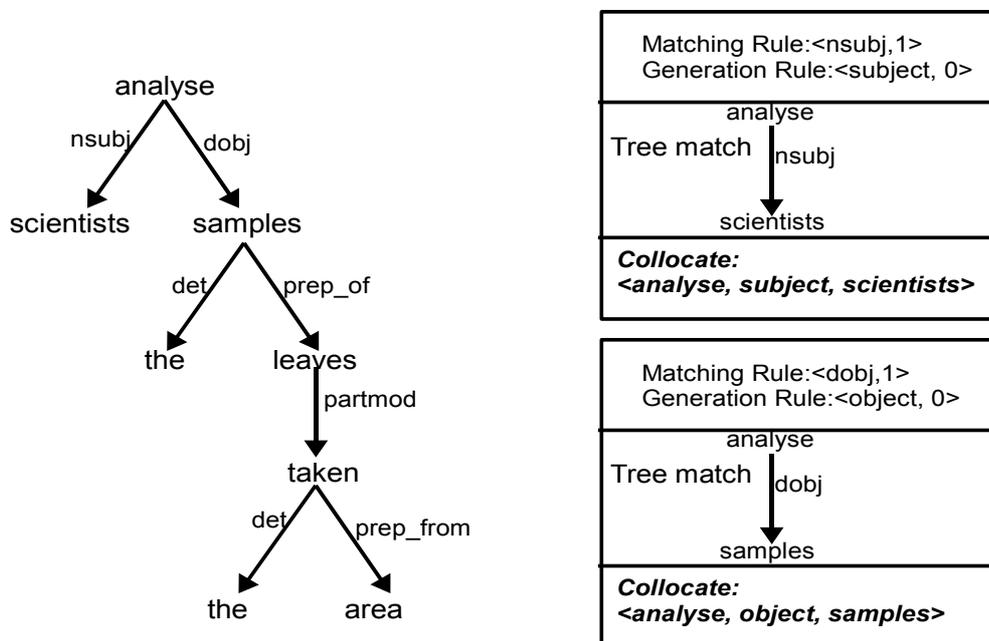


Figure 2: The dependency tree of the sentence *Scientists analysed samples of the leaves taken from the area* is shown on the left. On the right the two rules that matched with the tree and the corresponding generation rules are shown.

Separate rules are written for noun, verb and adjective senses. (Other word classes have not been addressed.) There are nineteen matching rules along with their corresponding generation rules. These are employed to extract 54,575 <headword, grammatical-relation, collocate> triples.

Below are some of problems encountered using the dependency parser.

1. There are a number of stop words listed as collocates. For example, *have, through, he, be*.
2. Sometimes the parser changed the spelling of the word, For example it changed *grey* to *gray*.
3. The parser failed to give the correct dependency structure of some sentences.
4. In some cases the parser did not detect the sentence boundaries. This happens because the parser recognises the sequence <single capitalized letter, period, space, capitalized word> as a person name, for example *John F. Kennedy*. But it is not always correct: it does not hold in the following sentence:

Write your full name in Box A. There were names ...

As a result the parser interpreted the above as a single sentence.

In addition to parsing dictionary examples, we:

- manually crafted rules to extract collocates from the dictionary's 'grammar patterns'
- used style labels for filtering out senses to which collocates would not be assigned. Senses marked *humorous, impolite, offensive, literary, old-fashioned, very formal, very informal* presented other concerns and were not to be included in the analysis
- excluded phrases that had their own entry in the dictionary, for example *Academy Award, Monty Python, Nobel Prize* and *Rhodes Scholarship*. There are 8,885 phrasal headwords in MEDAL.

3.3 Yarowsky

Yarowsky (1995)'s method performs minimally supervised word sense disambiguation. It uses the collocates that are reliable indicators of the sense of the word to perform the disambiguation. It begins with a small set of sense-specific collocates and incrementally augments them using a combination of two heuristics:

- ‘one sense per collocation’: collocations tend to occur only with one sense of a word
- ‘one sense per discourse’: multiple occurrences of a word in the same text, or discourse, tend to all be uses of the same sense of that word (at least at the coarse-grained level: see Krovetz (1998)).

The algorithm first collects a large number of concordances for the word. Then for each sense of the word a small number of examples representative of that sense are identified. These representative examples are called sense seed sets. Collocates specific to the sense are used to identify these seed sets: we used the high-precision collocates obtained from dictionary parsing. After this step a number of concordances still remain unassigned to a sense. It then proceeds in an iterative manner by updating the seed sets in each step. The seed sets are used in training a decision list algorithm. This in turn is used to label the unassigned samples and for adding the high probability samples to seed sets. In this process the sense specific collocates are identified by the decision list algorithm. Finally a stopping criterion is used to terminate the algorithm.

The algorithm is summarised as follows:

Algorithm 2: Yarowsky

1. Get 50000 concordances of the polysemous word using the Sketch Engine
 2. Identify seed collocates obtained by dictionary-parsing.
 3. Build initial seed sets of concordances: the concordance lines containing seed collocates
 4. Training and labelling new samples:
 - 4a Train the supervised classification algorithm on the sense-specific seed sets using the decision list algorithm
 - 4b. Use the resulting classifier to label the entire sample set
 - 4c. Add the high scoring unassigned samples to seed sets
 - 4d. Collect the high scoring collocates.^v
 - 4e. Use one sense per discourse constraint to augment this addition
 - 4f. Go to step 4a
 5. Stopping criteria: the algorithm is run until convergence. (In our experiments we stopped the algorithm after 5 iterations.)
-

4. A REVIEW: AUTOMATIC vs COMPUTER-SUPPORTED

As at January 2009, the performance of the algorithm was still disappointing, with many non-standard features of specific word senses still not handled correctly. The lexicographic work on MCD was to start shortly, and this required a working environment for lexicographers in which the automatic work speeded up their work rather than causing them delay. Editing incorrect work tends to be slower and more awkward than doing it correctly in the first place. A decision was taken to provide computational support for manually assigning Sketch Engine collocates to MEDAL senses rather than attempting the task automatically. The development of the automatic system has recently restarted, in readiness for future projects by which time we expect to be able to assign collocates more accurately, in parallel with the lexicography for MCD.

5. TICKBOX LEXICOGRAPHY

In corpus lexicography we:

- identify the senses for the word

and then, for each sense:

- identify the key patterns, collocates and phrases
- find example sentences.

This is then the base analysis of the language which will serve for the development of a range of dictionaries, monolingual and bilingual (where the language analysed is the source language, and

the same analysis will form the basis whatever the target language) (Atkins, 1994; Atkins and Rundell, 2008: 97-101).

5.1 GDEX (Good Dictionary Example Finder)

Kilgarriff et al. (2008) presents GDEX, a system, integrated with the Sketch Engine, for finding good dictionary examples for a word or collocation. It works by taking the sentences in a concordance and sorting them according to a number of heuristics including sentence length, the proportion of high-frequency words, the number of obscure words (if any) and the number of capital letters, numbers and other non-letter characters. It then either sorts the concordance lines 'best-first' in the Sketch Engine, so typical users see only the best examples, or outputs the best examples to some other process.

GDEX opens the way to further supporting the lexicographer by letting them select collocates and examples by ticking boxes rather than re-typing or using standard cut-and paste. We call this "checkbox lexicography" (TBL).

The process is as follows:

- the lexicographer sees a version of the word sketch with tickboxes beside each collocate
- for each sense and each grammatical relation, they tick the collocates they want in the dictionary
- they click a 'next' button
- they then see, for each collocate they have ticked, a choice of six (by default) corpus example sentences, chosen by GDEX, each with a tickbox beside it
- they tick the ones they like
- they tick a "copy to clipboard" button.

The system then copies the collocates and examples, embedded in an XML structure, onto the clipboard. (Each target dictionary has its own TBL application, to meet the requirements of the user's dictionary-editing system and target dictionary.) The lexicographer can then paste the structure into the dictionary editing system.

Thus, TBL models and streamlines the process of getting corpus data out of the corpus and into the dictionary.

5.2 Customising TBL for MCD

To set up TBL for MCD, we first developed customised word sketches in which the grammatical relations were those to be used in MCD. This required work on the underlying part-of-speech tagging and grammatical-relation-finding software. GDEX was also customised, with the incorporation of a long list of stop words, to minimise the chances that GDEX would select examples containing offensive material.

In the first trials, lexicographers selected all the example sentences (typically six per collocate) that were to be used in the electronic version of MCD, but this proved too slow. We changed to a strategy where only the examples which are to appear in the book are selected by lexicographers. For all others, GDEX will be trusted to deliver good examples. (The manually-selected items will be edited as necessary by lexicographers, whereas the others will be full and unedited corpus sentences.) These sentences will be selected in a batch process after the lexicography is done, as this will reduce the volume of data to be handled by the clipboard and the dictionary editing system, and will allow the GDEX process to be fine-tuned, using experience during the project as a guide, towards the end of the project.

MEDAL 2007 already contains 500 'collocation boxes' for word senses of common words, with collocations classified according to grammatical relations, and further collocations in bold in regular entries. It was desirable to carry them across into MCD in a way which integrated with MCD lexicography. To this end we:

- analysed MEDAL to find all collocations, either in collocation boxes or shown in bold within regular entries
- identified the grammatical relation they stood in to the headword
- checked to see if they were already in the word sketch:
 - if they were (as they usually were), colour them red (in the word sketch) and pre-tick the tickbox, as they will almost always be wanted in MCD
 - if they were not, add them in (in red), with links to their corpus instances and pre-ticked tickboxes.

The dictionary editing software used for MCD accepts XML pasted from the clipboard so, once the lexicographer has

- called up the customised word sketch for the headword
- selected the grammatical relation
- selected collocates
- selected examples for the paper dictionary

they click a 'copy to clipboard' button, and then paste the material (using standard CTRL-V) into the dictionary entry.

5.3 Current Status

The computer-supported approach, with TBL and GDEX, has been in use in production mode, by a team of six lexicographers, since mid 2009. Following three months of intensive development to optimise the process for the specific situation of MCD and MEDAL, both the lexicographers and the Macmillan management team are clear that the approach is both speeding up the compilation and contributing to the accuracy and completeness of MCD. We believe TBL has great potential for both streamlining corpus lexicography and making its outputs more accountable to the corpus.

6. ACKNOWLEDGEMENTS

This work has been partly supported by the Ministry of Education of CR within the Center of basic research LC536.

7. REFERENCES

- Agirre, E. and P. Edmonds (eds.) (2006) *Word Sense Disambiguation: Algorithms and Applications*. Springer.
- Agirre, E., L. Marquez, R. Wicentowski (Editors) (2009) *Language Resources and Evaluation 43 (2)*, Special Issue on Computational Semantic Analysis of Language: SemEval-2007 and Beyond.
- Agirre, E and D. Martinez (2004). Unsupervised WSD based on automatically retrieved examples. *Proceedings of EMNLP*.
- Atkins, B. T. S. (1994) A corpus-based dictionary. In: *Oxford-Hachette English-French Dictionary* (Introductory section). Oxford: Oxford University Press. xix – xxxii.
- Atkins, B. T. S. and M. Rundell (2008) *The Oxford Guide to Practical Lexicography*. Oxford University Press.
- Ferraresi, A., E. Zanchetta, M. Baroni, S. Bernardini. (2008) Introducing and evaluating ukWaC, a very large web-derived corpus of English. In S. Evert, A. Kilgarriff and S. Sharoff (eds.) *Proc. 4th Web as Corpus Workshop (WAC-4) – Can we beat Google?*, Marrakech, Morocco.
- Kilgarriff, A. (2005) Linking Dictionary and Corpus. *Proc. Asialex*, Singapore.
- Kilgarriff, A. (2006). Collocationality (and how to measure it) *Proc. Euralex*. Torino, Italy.
- Kilgarriff, A., R. Evans, R. Koeling, M. Rundell, D. Tugwell (2003) WASPBench: a lexicographers' workstation incorporating word sense disambiguation. Demo and research note. *Proc. European ACL*. Budapest.
- Kilgarriff, A., P. Rychlý, P. Smrz and D. Tugwell (2004) The Sketch Engine *Proc. Euralex*. Lorient, France: 105-116.
- Kilgarriff, A., M. Husák, K. McAdam, M. Rundell, P. Rychlý (2008) GDEX: Automatically finding good dictionary examples in a corpus. *Proc EURALEX*, Barcelona, Spain.
- Krovetz, R. (1998) More than One Sense per Discourse. NEC Princeton NJ Technical Memorandum.
- Klein, D., and C. Manning. 2003. . Fast Exact Inference with a Factored Model for Natural Language Parsing. *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, Cambridge, MA: MIT Press, pp. 3-10.
- McCarthy, D., R. Koeling, J. Weeds and J. Carroll (2004) 'Using automatically acquired predominant senses for word sense disambiguation'. In *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*, Barcelona, Spain. 151-154.

- MEDAL (2002,2007) *Macmillan English Dictionary for Advanced Learners*. First/Second edition. Edited by M. Rundell. Oxford.
- Moon, R. (2000). Lexicography and Disambiguation: The Size of the Problem. In: *Computers and the Humanities* 34 , p. 99-102
- OCD (2002, 2009) *Oxford Collocations Dictionary*. Oxford.
- Pomikálek, J. and P. Rychlý (2008) Detecting Co-Derivative Documents in Large Text Collections. In *Proc. LREC'08*. Marrakech, Morocco: 132-135.
- Schmid H. (1994) Probabilistic Part-of-Speech Tagging Using Decision Trees. Technical report.
<http://www.ims.uni-stuttgart.de/projekte/complex/TreeTagger/>
- Yarowsky, D. (1992) Word sense disambiguation using statistical models of Roget's categories trained on large corpora. *Proceedings of COLING-92* pp 454-460.
- Yarowsky, D. (1993) One sense per collocation. In *Proceedings, ARPA Human Language Technology Workshop*, pp. 266-271.
- Yarowsky, D. (1995) Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. 33rd Annual Meeting of the Association for Computational Linguistics*. Cambridge, MA, pp. 189-196.

ⁱ There are of course many qualifications which might be made here, as figures depend on how and what you count. Readers are referred to Agirre and Edmonds (2006) and Agirre et al (2009) for detailed discussions.

ⁱⁱ We use *collocation* to refer to the two-word unit, and *collocate* to refer to the word that collocates with the headword.

ⁱⁱⁱ Our version of UKWaC is 20% smaller than the one described in Ferraresi et al. (2008). Both Ferraresi's group and ours have undertaken further rounds of removing unwanted material, and both groups now use versions which have benefited from the other group's work.

^{iv} We are aware that *become* is a copula verb and does not take an object. In the simplified grammar of the word sketches, the distinction is not made.

^v In each iteration the top 2% of collocates over all senses are collected.