Validation and Tuning of Wordnet Mapping Techniques*

Jordi Daudé and **Lluís Padró**
TALP Research Center
Universitat Politècnica de Catalunya
Barcelona, Catalonia
{daude,padro}@lsi.upc.es

German Rigau
IXA Group
Euskal Herriko Unibertsitatea
Donosti, Basque Country
{rigau}@si.ehu.es

Abstract

In this paper we present an accurate, quantitative and qualitative validation of the methodology used by (Daudé *et al.* 01) to map two WordNet versions. We check the accuracy of the technique by applying it to map a WN version onto itself, which enables not only quantitative evaluation but also a qualitative study of the error cases and algorithm tuning. In addition, we also evaluate the behaviour of the technique when mapping non-identical hierarchies by randomly erasing synsets from either the target or the source copy of the used WordNet.

1 Introduction

Building appropriate resources for broad–coverage semantic processing is a hard and expensive task, involving large research groups during long developement periods. The outcomes of these projects are, usually, large and complex semantic structures, not compatible with resources developed in previous projects and efforts. This fact has severely hampered Human Language Technology (HLT) development.

Thus, in order to integrate in a common multilingual resource several, already developed, large-scale knowledge sources, a powerful and robust mapping tool is required to solve version gaps and minimize side effects. Some examples are the MultiWordNet Domains –aligned to WN1.6 (Magnini & Cavaglià 00)–, the Spanish, Catalan and Basque wordnets –aligned to WN1.5 (Atserias *et al.* 97; Benítez *et al.* 98)–, the EuroWordNet Top Ontology (connected to WN1.5), or Balkanet and EuroTerm initiatives (aligned to WN1.7).

Nevertheless, automatic ontology mapping methods are difficult to evaluate. Hand checking of a small –statistically significant– sample of the performed connections, provides a quantitative idea of the accuracy of the technique, but does not allow to draw qualitative conclusions.

*This research has been partially funded by the Spanish Research Department (TIC2000-0335-C03-02, TIC2000-1735-C02-02), by the European Comission (IST-2001-34460), and by the Catalan Research Department (CIRIT 1999SGR-150).

In this paper we check the accuracy of the (Daudé *et al.* 01) technique by mapping WN1.5 version onto itself, which enables not only quantitative evaluation, but also the qualitative study of error cases. In addition, we also evaluate the behaviour of the technique in the more realistic case of mapping non-identical hierarchies, by randomly erasing synsets from either the target or the source copy of the used WordNet.

2 Method Description

Relaxation labelling (RL) is a generic name for a family of iterative algorithms which perform function optimization, based on local information, but with global effects. See (Torras 89) for a summary, or (Padró 98; Atserias *et al.* 01) for previous applications to NLP tasks. One of its most remarkable features is that the focus problem is modelled in terms of compatibility/incompatibility constraints (which may be hand-written, statistical, machine-learned, ...) between variable–label pairs.

RL uses constraints to increase or decrease the weight for a variable label. In our case, constraints increase the weights for the connections between a source synset and a target synset. Increasing the weight for a connection implies decreasing the weights for all the other possible connections for the same source synset. To increase the weight for a connection, constraints take into account already connected nodes that have the same relationships in both taxonomies.

The problem is modelled with a variable for each node in the source taxonomy, which has as possible labels all candidate connections for that node (see Figure 1). Used constraints rely on checking the existence of a connected ancestor/descendant for both ends of a candidate connection. Complexity of constraints varies on the allowed distance from the candidate connection and in the simultaneously checked conditions. The RL algorithm will select the label assignment

for all variables (i.e. the connection for each node) which better satisfies all constraints. More details on the algorithm and constraints can be found in (Daudé *et al.* 00; Daudé *et al.* 01).

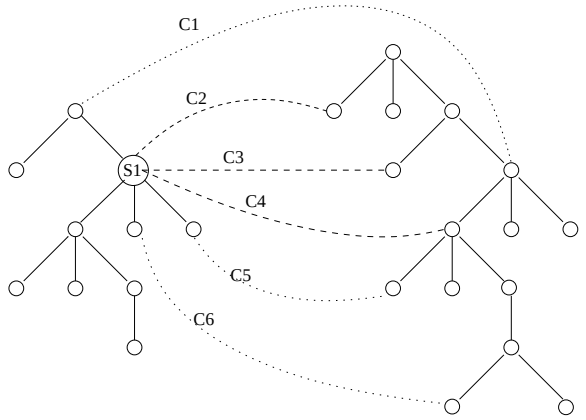

Figure 1: Example of candidate connections

Figure 1 shows an example of possible connections between two taxonomies. For source node S_1, connection C_4 will have its weight increased due to C_5, C_6 and C_1, while connections C_2 and C_3 will have their weights decreased. Eventually, label C_4 will be assigned to variable S_1.

3 Validation via *automapping*

In order to evaluate the performance of the algorithm, we mapped the nominal part of WN1.5 onto itself. The nominal WN1.5 is almost a tree –few nodes have more than one hyperonym–, and consists of 60,557 nodes, 11 of which are root nodes, and 47,110 (77.79%) are leafs.

The candidate conections for a source node are obtained retrieving all synsets in the target taxonomy for all words contained in the source synset. Since the target taxonomy contains a copy of the source synset, all synsets have at least one candidate connection. In WN1.5, 37,204 synsets are *single-link*, that is, they have only one candidate connection. They don't need to be disambiguated, but are helpful to solve ambiguity for other nodes connected with them. The remaining 23,353 synsets (38.56%) are *multiple-link*, i.e. have more than one candidate connection. The number of candidates per multiple-link sysnset ranges from 2 to 66, with an average of 4.26.

Using the algorithm with the same taxonomy as source and target not only is useful to evaluate its correctness and efficiency, but also to tune some of the used constraints, and to detect existing gaps and incorporate new constraints to cover them.

In this paper we analyze the behaviour of the algorithm on an incremental basis, starting with the simplest constraint configuration, and progressively extending the used model to enhance its performance.

3.1 Immediate connection (II) constraints

The simplest constraint set checks for the existence of a connection between *immediate* (II) hypernyms or hyponyms at both ends of the candidate connection, such as (C_4, C_1) in Figure 1.

Table 1 presents the results obtained using II constraints. Precision and recall are given over single and multiple link synsets. Recall is computed as the percentage of source nodes that keep the correct connection among their proposed targets. Precision is computed as the number of proposed targets that are correct connections.

Over trivial single-link synsets, the performace is obviously perfect. Over the multiple-link subset, some correct links are discarded by the algorithm, yielding a recall below 100%. There are only ten error cases –grouped in four clusters– which can be found in Figure 2.

In each cluster, the error in one of the synsets causes the error in the others. For instance, case A in Figure 2 is more detailed in Figure 3, where we can observe that the target synset `00145061` is only reinforced by constraint C1, while target `08150656` receives support from constraints C2 and C3, causing it to be wrongly selected.

		II	IIB
	#NODES	PREC.-RECALL	PREC.-RECALL
single-link	37,204	100%-100%	100%-100%
multiple-link	23,353	93.80%-99.96%	93.86%-100%
Total	60,557	97.51%-99.98%	97.54%-100%

Table 1: Precision-recall results obtained using II and IIB constraint sets

II constraints provide support for a link from the existence of either a linked hyperonym or hyponym, but not from the simultaneous existence of them both. IIB constraints extend the II set with an extra support for those links with a *simultaneously* linked hyperonym and hyponym. This is precisely the case in the above mentioned errors, since for instance in case A, both hyperonym and hyponym for the source `00145061` are linked with the respective hyperonym and hyponym for target `00145061`, while the hypernym for source `00145061` is not linked with the hyperonym for

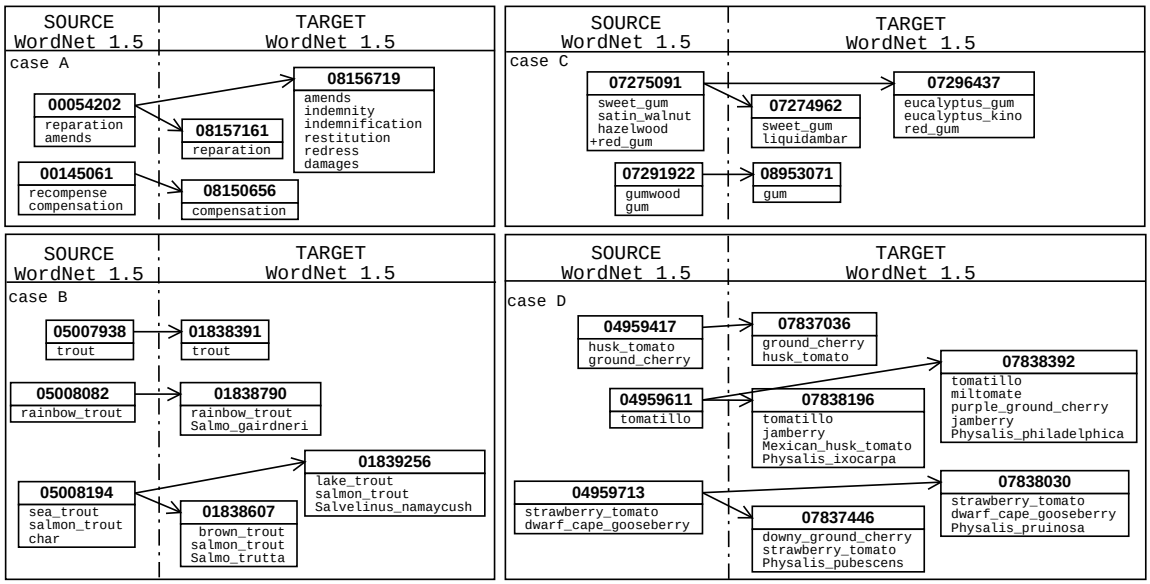

Figure 2: All wrong links selected by II constraint.

the other candidate target 08150656.

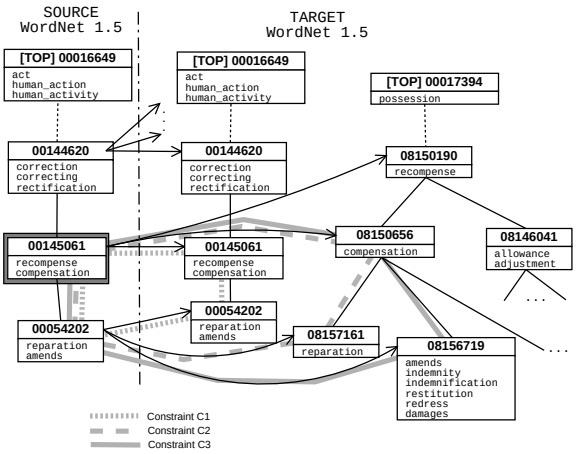

Figure 3: Details of wrong link in Fig. 2, case A.

The use of IIB constraint will provide additional evidence in favour of the correct link, that should overwhelm the evidence provided by two hyponym constraints supporting the wrong candidate. As can be seen in Table 1, the use of these constraints produces a recall of 100% and an increment in precision, solving all wrong links presented in Figure 2.

This confirms the need for B constraints to help the disambiguation in cases such as those presented in the example. Note that this is a general statement, valid for any hierarchy, since only class/subclass relationships are being used.

3.2 Using extra hyponym information

Although we have a 100% recall, precision is not perfect yet. This is due to remaining ambiguity in some nodes. Figure 4 presents an example of such a node (00026244) that occurs either with II or IIB constraints. Details on the involved relationships are also depicted: We can observe that source 00026059 is correctly linked since its hyponyms (00029218 and 00171746) provide the necessary evidence. Contrarily, source 00026244 is not disambiguated because both candidates have the same supporting evidence: constraint C1 for one candidate and C2 for the other.

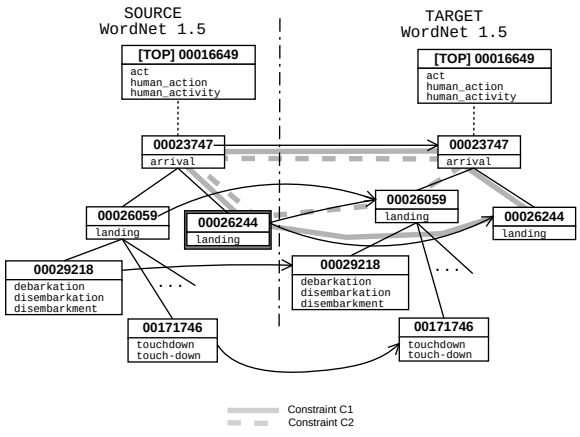

Figure 4: Relationship structure for ambiguous node example

This cases could be solved if knowledege about the number of daughters of each node was taken into account. We tested the following two ways of

using this information (see Table 2 for results):

1. ZD constraint (Zero Daugthers): A simple boolean check consisting of a constraint that reinforces a connection between two leaf nodes (i.e. when both have zero daugthers).
2. ED constraint (Equal Daugthers): A generalization of the previous, consisting of a reinforcement of a connection between nodes with equal number of daughters.

		IIB+ZD	IIB+ED
	#NODES	PREC.-RECALL	PREC.-RECALL
single-link	37,204	100%-100%	100%-100%
multiple-link	23,353	94.90%-100%	94.93%-100%
Total	60,557	97.97%-100%	97.98%-100%

Table 2: Precsion-recall results when using constraints on the number of daughters.

When using constraints IIB+ZD, 1,136 nodes remain ambiguous, all but three of which are leaf nodes. One of these three is synset `02323757`, presented in Figure 5.

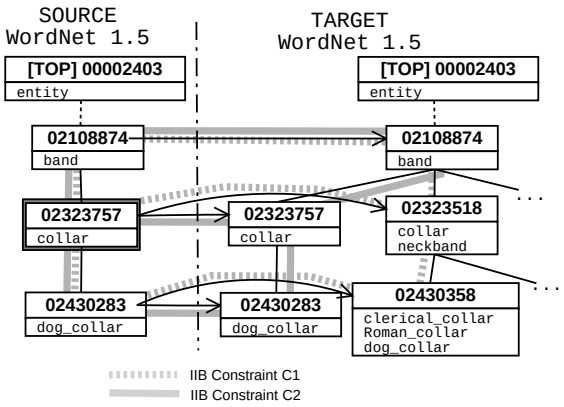

Figure 5: Example of non-leaf ambiguous node.

It can be observed that the ambiguity between targets `02323757` and `02323518` is caused by IIB constraints C1 and C2 in Figure 5, and since `02323757` is not a leaf, ZD constraint does not apply. If constraint ED is used instead, the ambiguity is correctly solved, since the synset for `dog_collar` is correctly linked, causing its hyperonym to be also correcly disambiguated.

When using IIB+ED constraints, the amount of remaining ambiguous nodes is 1,129, all of them leafs. Leaf nodes are the weakest point of the algorithm, since they have no descendants to provide information. Thus, when a node has as candidate targets two leaf sibling synsets, disambiguation is not possible using only hyper/hyponymy relationships. Example of such cases are the three leaf nodes in Figure 6.

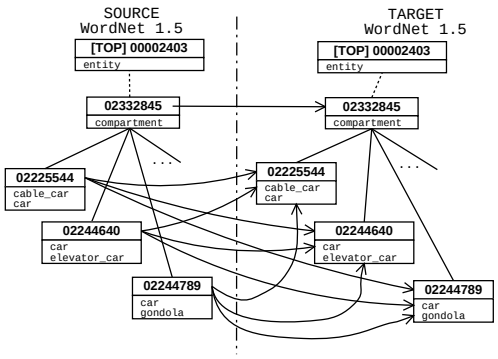

Figure 6: Example of ambiguity in leaf nodes

3.3 Using other relationships

Although the main structure of WordNet relies in the taxonomical hyper/hyponymy relationships, it contains many other relationships. The nominal part includes also antonymy, meronym, holonymy and attribute. The former three are *noun-to-noun*, i.e. internal to the nominal part, and the later relates *noun-to-adj*.

Since each ambiguous synset has different meronyms, using an II constraint on this relationship enables the algorithm to solve those ambiguity cases. Results when using all *noun-to-noun* relationships (plus ED constraints) are presented in the *Structural* column in Table 3.

With this model, there are 765 nodes that still remain ambiguous, since they do not have any other relationship we can use to provide extra information to help the disambiguation process. Thus, the use of non-structural information (i.e. not related to node relationships but to node similarity measures) will be necessary. Some of those cases appear in Figure 7.

		Structural	Structural+WG
	#NODES	PREC.-RECALL	PREC.-RECALL
single-link	37,204	100%-100%	100%-100%
multiple-link	23,353	96.54%-100%	99.991%-100%
Total	60,557	98.64%-100%	99.997%-100%

Table 3: Precision-recall results obtained with each constraint model

3.4 Using non-structural information

To disambiguate cases in which a decision is not possible using only relationship-based constraints,

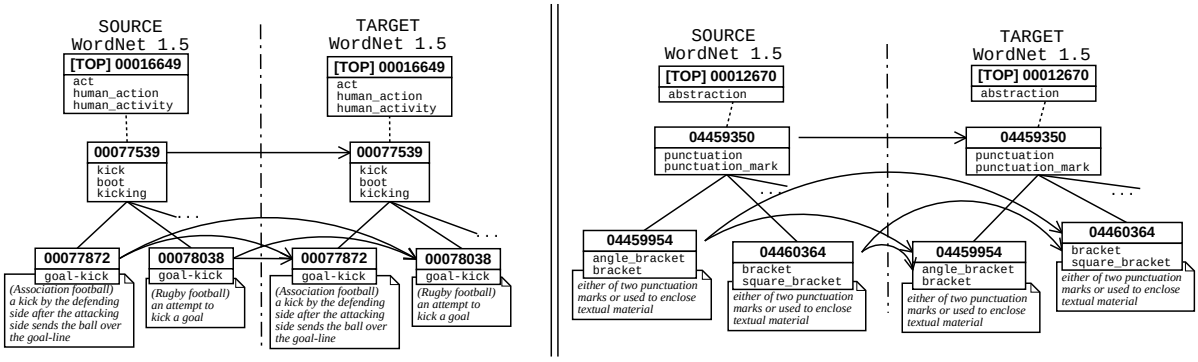

Figure 7: Example of nodes than can not be disambiguated with only relationship structure information

we may extend our model with non-structural information which supports the connection between similar nodes. This obviously requires a way of computing node similarity that does not depend on the relationships among them. In the case of WN we may use information internal to the node:

1. W constraint (coincident Words). The larger the number of coincidences in the words of two synsets, the more similar they are considered.
2. G constraint (coincident Gloss). The larger the number of coincidences in the words of both synsets glosses, the more similar they are considered. Non-content words (articles, prepositions, etc.) are excluded.

Using W constraint (word coincidence count) correctly disambiguates the example presented on the left of Figure 7. Similarly, the G constraint (gloss coincidence count) correctly disambiguates the right hand side example. Thus, to disambiguate as many cases as possible, we will use both constraints, though the G constraint will have a low coverage, since many WN1.5 synsets do not have a gloss.

Rightmost column in Table 3 shows the results obtained with all structural and non-structural constraints. There are only two remaining ambiguous synsets, one of which is presented as sample in Figure 8. It can be seen that there is not enough information in the taxonomy (even for humans) to disambiguate those cases, nevertheless, one may wonder if they are actually different senses or merely an error in the hierarchy.

Thus, our validation method via the mapping of a hierachy onto itself turns out to be also useful to detect possibly duplicated concepts in the semantic network.

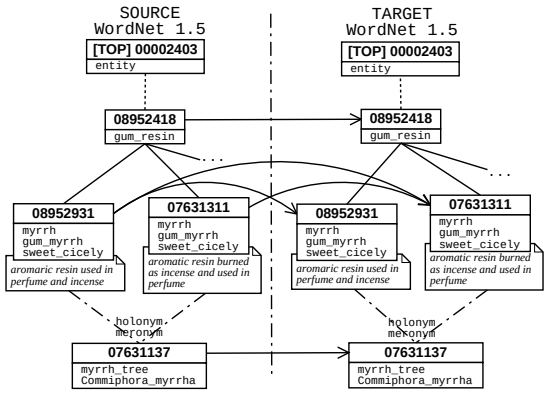

Figure 8: Example of node that can not be disambiguated with all the used constraints.

4 Mapping non-identical hierarchies

We have used relaxation labelling to map a hierarchy onto itself, and this has allowed us to validate and tune the technique. But since we are interested in using it on non-identical taxonomies (different WN versions, or different resources, even for different languages), we must consider whether the algorithm will we equally useful when used on different taxonomies.

In that case, II constraints may not adequate to our needs, since although both structures may be build under similar criteria, there may be local differences that may result in node insertions or deletions in one side respect to the other.

For this reason, more general constraints will be necessary, in order to allow the connection of candidate nodes that do not have the same relationships with their neighbours, but do have similar relationship patterns. The used constraints will be the same than above, but the connected ancestor/descendant is no longer required to be *immediate*, thus, *any* (AA constraints) matching

node will be recursively searched, this is the case for pair (C_4, C_6) in Figure 1.

Results when using AA constraints over two identical hierarchies (WN1.5-WN1.5) are presented in Table 4. As could be expected, since II constraints are a particular case of AA, precision results are identical to those in tables 2 and 3. Nevertheless there is a slight decrease in recall, which correspod to two wrongly solved synsets, with a similar cause: One of those cases is synset 02526527 in Figure 9.

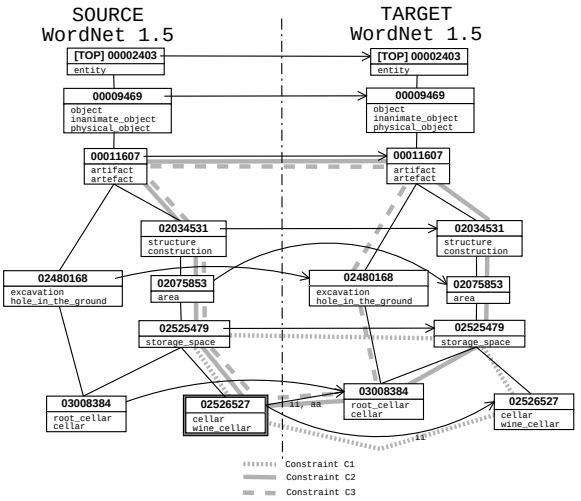

Figure 9: Example of wrong connection when using AA constraints.

If we use only II constraints, this node remains ambiguous, since its two candidates have the same supporting evidence (a connected hyperonym), while if we use AA constraints, there is only one constraint (C1 in the figure) supporting the right candidate, but two constraints (C2 and C3) for the wrong one. This behaviour is difficult to avoid, though a promising possibility would be performing bidirectional mapping, that is, instead of assigning a passive role to the target taxonomy, have it simultaneously mapped to the source, and take advantadge of the coincidences in both sides.

To evaluate the algorith perfomance when mapping different hierarchies, we start mapping two identical taxonomies (WN1.5), progressively introducing differences between them. The introduced diferences consist of node deletions from one of the taxonomies (either target or source). Note that the deletion of one node in one taxonomy can be seen as an insertion of its corresponding synset in the other. The nodes to be deleted are randomly selected, restructuring the hierarchy to maintain consistency:

- If a root node is deleted, all its daughters become roots.
- If an intermediate node is deleted, all its daughters become daughters of the parent of the deleted node. If the parent is also eliminated, this rule applies recursively. This also applies to all transitive relationships other than hyper/hyponymy (namely, all kinds of holonymy/meronymy).
- Non-transitive relationships arriving to the deleted node are simply deleted. This is the case of antonymy.

The simplest case is keeping the target taxonomy untouched, and mapping onto it a progressively distorted version of the same hierarchy. In this case, all remaining source nodes after the distortion still have the same candidate connections they had in the original taxonomy, since target structure is unchanged. Evaluation for this case using II, AA and AA+WG is presented in Figure 10. Values are computed over multi-link synsets.

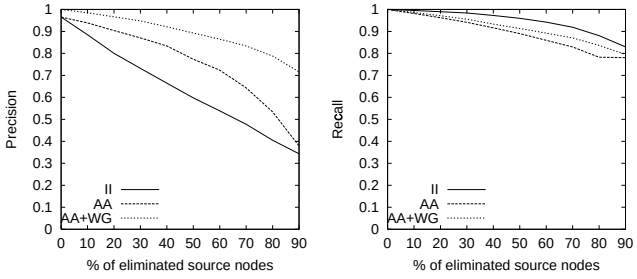

Figure 10: Results when mapping a progressively distorted WN1.5 taxonomy to a complete one.

The points at zero value for the x-axis correspond to the unmodified WN1.5 mapping presented above. It can be observed how the AA and AA+WG constraint sets are more robust that II when divergence between both taxonomies is increased. II maintaints a higher recall, but the cost is a much lower precision. The compromise evaluated via $F_{\beta=1}$ yields the conclusion that recursive constraints better deal with taxonomy distortion.

The reverse case, i.e. mapping a complete taxonomy onto a distorted yields lower results, since there are source nodes for which their correct connection is removed from the target, or even, all possible connections for that node are removed from the target, causing a larger decrease both in precision and recall.

	#NODES	AA+ED PREC.-RECALL	AA Structural PREC.-RECALL	AA Structural+WG PREC.-RECALL
single-link	37,204	100%-100%	100%-100%	100%-100%
multiple-link	23,353	94.93%-99.99%	96.54%-99.99%	99.991%-100%
Total	60,557	97.98%-99.99%	98.64%-99.99%	99.997%-100%

Table 4: Results when mapping WN1.5 onto itself using recursive constraints.

Apart from keeping either source or target taxonomy untouched, we may consider the general case where both taxonomies are distorted in some degree. This is more similar to a real case in which the source taxonomy has both insertions and deletions with respect to the target taxonomy.

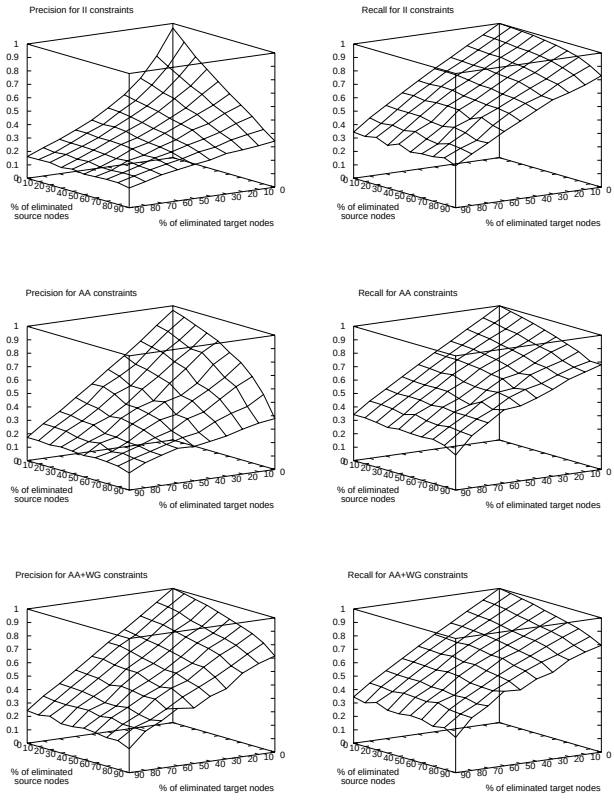

Figure 11: Results when mapping two progressively distorted WN1.5 taxonomies.

Figure 11 presents the results for all combinations of distortion levels. Obviously the list of randomly deleted nodes is not the same for each taxonomy, although there may be nodes that, by chance, belong to both lists. It is worth noting that for a high distortion levels such as 90%-90%, each taxonomy is only 10% in size of the original (since 90% of the nodes have been deleted), and that the expected proportion of nodes present in both taxonomies is the probability that a node is not deleted in any of both taxonomies (that is, $0.1 \times 0.1 = 0.01$), i.e. 1% of the original taxonomy (or 10% of the distorted versions, since they are ten times smaller). Even in these conditions, results keep about 10%, that is, the algorithm is connecting a large amount of the nodes that can be connected, since the remaining 90% have no candidate connections.

5 Conclusions

We have validated a WN mapping technique based on relaxation labelling through the analysis of the results of mapping WN1.5 onto itself. The main conclusions of this work are that the proposed method is reliable and degrades gracefully when differences between mapped taxonomies increase, maintaining a reasonable precision and recall level since the lower results correspond to a higher amount of unsolvable nodes more than to a higher error rate. We also detected which constraints are usefeul to solve several errors commited by the base model, and spotted a set of WN1.5 synsets that present neither structural nor lexical differences.

References

(Atserias *et al.* 97) J. Atserias, S. Climent, X. Farreres, G. Rigau, and H. Rodríguez. Combining Multiple Methods for the Automatic Construction of Multilingual WordNets. In *proceedings of International Conference on Recent Advances in Natural Language Processing (RANLP'97)*, Tzigov Chark, Bulgaria, 1997.

(Atserias *et al.* 01) J. Atserias, L. Padró, and G. Rigau. Integrating Multiple Knowledge Sources for Robust Semantic Parsing. In *proceedings of International Conference on Recent Advances in Natural Language Processing (RANLP'01)*, Tzigov Chark, Bulgaria, 2001.

(Benítez *et al.* 98) L. Benítez, S. Cervell, G. Escudero, M. López, G. Rigau, and M. Taulé. Methods and Tools for Building the Catalan WordNet. In *Proceedings of ELRA Workshop on Language Resources for European Minority Languages*, Granada, Spain, 1998.

(Daudé *et al.* 00) J. Daudé, L. Padró, and G. Rigau. Mapping WordNets Using Structural Information. In *38th Annual Meeting of the Association for Computational Linguistics (ACL'2000).*, Hong Kong, 2000.

(Daudé *et al.* 01) J. Daudé, L. Padró, and G. Rigau. A Complete WN1.5 to WN1.6 Mapping. In *NAACL Workshop "WordNet and Other Lexical Resources: Applications, Extensions and Customizations" (NAACL'2001).*, Pittsburg, PA, USA, 2001.

(Magnini & Cavaglià 00) B. Magnini and G. Cavaglià. Integrating subject field codes into wordnet. In *In Proceedings of the Second Internatgional Conference on Language Resources and Evaluation LREC'2000*, Athens. Greece, 2000.

(Padró 98) L. Padró. *A Hybrid Environment for Syntax–Semantic Tagging.* Phd. Thesis, Dep. Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya, February 1998. http://www.lsi.upc.es/~padro.

(Torras 89) C. Torras. Relaxation and Neural Learning: Points of Convergence and Divergence. *Journal of Parallel and Distributed Computing*, 6:217–244, 1989.