A New Machine Learning Algorithm for Neoposy: coining new Parts of Speech

Eric Atwell,
School of Computing, University of Leeds
eric@comp.leeds.ac.uk
http://www.comp.leeds.ac.uk/eric

1. Introduction: Unsupervised Natural Language Learning

According to the Collins English Dictionary, "neology" is: *a newly coined word, or a phrase or familiar word used in a new sense; or the practice of using or introducing neologies*
.
We propose "neoposy" as a neology meaning "*a newly coined classification of words into Parts of Speech; or the practice of introducing or using neoposies*".

Unsupervised Natural Language Learning, the use of machine learning algorithms to extract linguistic patterns from raw, un-annotated text, is a growing research subfield; for examples, see Proceedings of annual conferences of CoNLL: Computational Natural Language Learning, or the membership list of ACL-SIGNLL, the Association for Computational Linguistics – Special Interest Group in Natural Language Learning. Corpora, especially tagged and parsed corpora, can be used to train `machine learning' or computational language learning models of complex sequence data. (Jurafsky and Martin 2000) divide Machine Learning systems into Supervised and Unsupervised approaches (p118):

"… The task of a machine learning system is to automatically induce a model for some domain, given some data from the domain and, sometimes, other information as well… A supervised algorithm is one which is given the correct answers for some of this data, using those answers to induce a model which can generalize to new data it hasn't seen before… An unsupervised algorithm does this purely from the data. While unsupervised algorithms don't get to see the correct labels for the classifications, they can be given hints about the nature of the rules or models they should be forming… Such hints are called a learning bias."

Hence, in Corpus-based Computational Language Learning, a supervised algorithm is one trained using an annotated corpus; for example, a supervised ML parser such as (Atwell 1988; 1993) is trained with a Treebank of example sentences annotated with their parses. An unsupervised algorithm has to devise an analysis from raw, un-analysed corpus data; for example, an unsupervised ML parser such as (van Zaanen 2002) is trained with raw text sentences and has to propose phrase-structure analyses "by itself".

2. Clustering words into word-classes

A first stage in Unsupervised Natural Language Learning (UNLL) is the partitioning or grouping of words into word-classes. A range of approaches to clustering words into classes have been investigated (eg Atwell 1983, Atwell and Drakos 1983, Hughes and Atwell 1994, Finch and Chater 1993, …, Roberts 2002). In general these researchers have tried to cluster word-types whose representative tokens in a Corpus appeared in similar contexts, but varied what counts as "context" (eg all immediate neighbour words; neighbouring function-words; wider contextual templates), and varied the similarity metric and clustering algorithm.

This approach ultimately stems from linguists' attempts to define the concept of word-class in term of syntactic interchangeability; the Collins English Dictionary explains "part of speech" as: *a class of words sharing important syntactic or semantic features; a group of words in a language that may occur in similar positions or fulfil similar functions in a sentence.* For example, the previous sentence includes the word-sequences *a class of* and *a group of* ; this suggests *class* and *group* belong to the same word-class as they occur in similar contexts.

Clustering algorithms are not specific to UNLL: a range of generic clustering algorithms for Machine Learning can be found in the literature (eg Witten and Frank 2000). These generic clustering systems require the user to formalise the problem in terms of a feature-space: every instance or object to be clustered must be characterised by a set of feature-values, so that instances with same or similar feature-values can be lumped together. Generally clustering systems assume each instance is independent; whereas when clustering words in a text, it may be helpful to allow the "contextual features" to either be words or be replaced by wordclass-labels as clustering proceeds (as in Atwell 1983).

3. Ambiguity in natural language word-classification

A common flaw, from a linguist's perspective, is that these clustering algorithms assume all tokens of a given word belong to one cluster: a word-type can belong to one and only one word-class. This results in neoposy which passes a linguist's "looks good to me" evaluation (Hughes and Atwell 1994, Jurafsky and Martin 2000) for some small word-clusters corresponding to closed-class function-word categories (articles, prepositions, personal pronouns): the author can claim that at least some of the machine-learnt word groupings "look good" because they appear to correspond to linguistic intuitions about word-classes. However, the basic assumption that every word belongs to one and only one class does not allow existing word-clustering systems to cope adequately with words which linguists and lexicographers perceive as syntactically ambiguous. This is particularly problematic for isolating languages, that is, languages where words are generally not inflected for grammatical function and may serve more than one grammatical function; for example, in English many nouns can be used as verbs, and vice versa.

The root of the problem is the general assumption that the word-type is the atomic unit to be clustered, using the set of word-token contexts for a word-type as the feature-vector to use in measuring similarity between word-types, applying standard statistical clustering techniques. For example, (Atwell 1983) assumes that a word-type can be characterised by its set of word-types and contexts in a corpus, where the context is just the immediately preceding word: two word-types are merged into a joint word-class if the corresponding word-tokens in the training corpus show that similar sets of words tend to precede them. Subsequent researchers have tried varying clustering parameters such as the context window, the order of merging, and the similarity metric; but this does not allow a word to belong to more than one class.

4. Classifying word types or word tokens?

One answer may be to try clustering word tokens rather than word types. In the earlier example, we can say that the specific word-tokens *class* and *group* in the given sentence share similar contexts and hence share word-class, BUT we need not generalise this to all other occurrences of *class* or *group* in a larger corpus, only to occurrences which share similar context. To illustrate, a simple Prolog implementation of this approach, which assumes "relevant context" is just the preceding word, produces the following:

```
?- neoposy([the,cat,sat,on,the,mat],Tagged).
Tagged = [[the,T1], [cat,T2], [sat,T3], [on,T4], [the,T5], [mat, T2]]
```

The Prolog variable Tagged is instantiated to a list of [word, Tag] pairs, where Tag has an arbitrary name generated by the program, letter T follwed by an integer. These integers increment starting from T1, unless a word has a "context" seen earlier, in which case it repeats the earlier tag. In the above example, word "mat" has the same context (preceding word "the") as earlier word "cat", so it gets the sameT2 tag instead of a new tag T6.

We see that the two tokens *the* have distinct tags T1 and T5 since they have different contexts; but the token *mat* is assigned the same tag as token *cat* because they have the same context (preceding word-type). This also illustrates an interesting contrast with word-type clustering: word-type clustering works best with high-frequency words for which there are plenty of example tokens; whereas word-token clustering, if it can be achieved, offers a way to assign low-frequency words and even hapax legomena to word-classes, as

long as they appear in a context which can be recognised as characteristic of a known word-class. In effect we are clustering or grouping together word-contexts rather than the words themselves.

5. How many token-classes will be learnt?

However, this prototype demonstrator also illustrates some problems with clustering on tokens. If we no longer assume all tokens of one type belong to one class, do we allow as many classes as there are tokens? Presumably not, as there would then be no point in clustering; in fact it would be hard to justify even calling this clustering. In the above example sentence there are 6 words ad 5 Tags: at least two words share a cluster, because they share a context. This means there are as many clusters as there are "contexts"; in the above case, a context is the preceding word-TYPE, so this implies we will get as many clusters as there are word-types. A million-word corpus such as the Lancaster-Oslo/Bergen (LOB) corpus yields about 50,000 word-types, so our simple token-clustered would yield about 50,000 word-classes. This is a lot less than a million (one per token), but arguably still too many to be useful.

A "learning hint" to guide clustering may be a constraint on the number of word-classes for a word-type, for example to say all tokens of a word-type must partition into at most 5 word-classes; but how is this to be achieved?

6 Clustering by constraint-based reasoning

In supervised Part-of-Speech tagging research, statistical approaches (eg Constituent Likelihood Automatic Word-tagging System CLAWS, Leech et al 1983) have been challenged (and arguably surpassed) by constraint-based taggers, exemplified by Transformation-Based Learning taggers (Brill 1995), and the ENGTWOL English Constraint Grammar tagger, (Voutilainen 1995). So, a constraint-based approach to neoposy may be worth considering. However, the constraints in Brill and Voutilainen taggers were based on surrounding PoS-tags, not words. If there are no "given" (supervising) PoS-tags, all constraints in our neoposy system would have to framed in terms of word-contexts. This introduces significant computational complexity: Voutilainen's tagger had over 1000 constraints based on parts of speech within a window context, but if each PoS-context had to be replaced with specific word-contexts the number of constraints could easily grow to unmanageable proportions.

For the neoposy task, we would like a word-type to be allowed to belong to more than one PoS-class. This means the set of tokens (and their contexts) for each word-type needs to be partitioned, into a small number of subsets, one for each PoS the word can have. We would like this partitioning to yield similar context-subsets for pairs of words which share a PoS. We can try to formalise this:

Each word-type W*i* is represented in a training corpus by word-tokens {w*i1*,w*i2*,…w*in*} and their corresponding contexts {c*i1*,c*i2*,…c*in*}

We wish to partition the set of contexts for every word-type W*i* in a way which maximises the similarity of context-subsets between words which have the same PoS:

We want to find partitions of context-sets for W*i*, W*j*, W*k,*… of the form:
{ {c*i1*,c*i2*,…c*ia*}, {c*ia+1,...*c*ib*}, {c*ib+1,...*}},…, {…, c*in*} },
{ {c*j1*,c*j2*,…c*ja*}, {c*ja+1,...*c*jb*}, {c*jb+1,...*}},…, {…, c*jn*} },
{ {c*k1*,c*k2*,…c*ka*}, {c*ka+1,...*c*kb*}, {c*kb+1,...*}},…, {…, c*kn*} }, …

in a way that maximises "reuse" of similar context-subsets between words:
{c*i1*,c*i2*,…c*ia*} ≈ {c*j1*,c*j2*,…c*ja*} ≈ {c*k1*,c*k2*,…c*ka*} ≈ … ,
{c*ic*,…c*id*} ≈ {c*je*,…c*jf*} ≈ {c*kg*,…c*kh*} ≈ … ,
…

This is a horrendously large constraint-satisfaction problem, potentially more challenging than traditional constraint-satisfaction applications such as scheduling, e.g. see (Atwell and Lajos 1993). A million-word training corpus such as LOB contains 1000K word-tokens and about 50K word-types, yielding an average of about 20 word-tokens, and hence word-token-contexts, per word-type. Cross-comparing all full context-sets, on the assumption that each word can belong to only one word-class, is already a heavy computational task. A set of 20 contexts could be partitioned in a very large number of ways, so it would take an impossibly long time to cross-compare every possible partitioning of every word with every other partitioning of every other word.

7. Semi-supervised clustering via a language discovery toolkit

A general observation in machine learning is that completely unsupervised learning is very hard, but even a little guidance can yield much more plausible results. The speculation above suggests that wholly unsupervised machine learning of token-based clustering may be unachievable, but perhaps some hybrid of token-based and type-based clustering, combined with limited sensible "learning hints" may be more manageable. To explore the range of possible components and parameters which might make up a successful hybrid solution, we need a "toolkit" of compatible "language discovery" software modules, to try putting together in various combinations.

8 Further research aims

This vague concept needs to be explored further, to pin down the sort of model and hints which could work; we need a programme of research combining Corpus Linguistics resources and Machine Learning in a Language Discovery Toolkit. (Atwell 2003) outlines a programme of further research:

1) to explore theories and models from Machine Learning and Corpus Linguistics, to synthesise generic frameworks and models;
2) to fit past and current research projects and software into a coherent generic framework and architecture;
3) to collate and develop a general-purpose software toolkit for experiments with a wide range of algorithms for Corpus-based Computational Language Learning: discovery of language characteristics, patterns and structures from linguistic training data;
4) to explore applications of this toolkit for Language Education, and for language-oriented knowledge-mining, in discovery of language characteristics, patterns and structures in a range of datasets from bioinformatics, astronomy, and multimedia datasets;
5) to disseminate the Language Discovery Toolkit to a wide range of potential users and beneficiaries, to uncover new unforeseen applications, by providing an internet-based showcase demonstrator for the wider research and education community.

References

E Atwell, 1983 "Constituent-Likelihood Grammar" in ICAME Journal Vol.7

E Atwell, 1988 "Transforming a Parsed Corpus into a Corpus Parser" in Kyto, M, Ihalainen, O & Risanen, M (eds), "Corpus Linguistics, Hard and Soft: Proceedings of the ICAME 8th International Conference on English Language Research on Computerised Corpora", pp61-70, Amsterdam, Rodopi

E Atwell, 1993 "Corpus-based statistical modelling of English grammar" in S Souter and E Atwell (eds), "Corpus-based computational linguistics: Proc 12th ICAME",pp195-214, Amsterdam, Rodopi

E Atwell, 2003. Combining Corpus Linguistics resources and machine Learning in a Language Discovery Toolkit. Internal research proposal, School of Computing, University of Leeds

E Atwell, N Drakos, 1987 “Pattern Recognition Applied to the Acquisition of a Grammatical Classification System from Unrestricted English Text” in B Maegaard (ed), “Proceedings of EACL’87: the Third Conference of European Chapter of the Association for Computational Linguistics”, Copenhagen, ACL

E Atwell, G Lajos, 1993 ”Knowledge and Constraint Management: Large Scale Applications” in E Atwell(ed), “Knowledge at Work in Universities: Proc 2nd HEFCs-KBSI” pp21-25, Leeds, Leeds University Press

S Finch, N Chater,1992 “Bootstrapping Syntactic Categories Using Statistical Methods” in “Proceedings 1st SHOE Workshop”, Tilburg University, The Netherlands

J Hughes, E Atwell, 1994 ”The automated evaluation of inferred word classifications” in A Cohn (ed), “Proc 11th European Conference on Artificial Intelligence”, pp535-539, Chichester, John Wiley

D Jurafsky, J Martin, 2000 “Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition”, Prentice-Hall, New Jersey

F Karlsson, A Voutilainen, J Heikkila, A Anttila (eds), 1995 ”Constraint Grammar”, Mouton de Gruyter, Berlin

A Roberts, 2002. Automatic acquisition of word classification using distributional analysis of content words with respect to function words, Technical Report, School of Computing, University of Leeds

M. van Zaanen, 2002. Bootstrapping Structure into Language: Alignment-Based Learning, PhD Thesis, School of Computing, University of Leeds.