2nd Hellenic Conf. on AI, SETN-2002, 11-12 April 2002, Thessaloniki, Greece,
Proceedings, Companion Volume, pp. 79-90

Learning Clause Boundaries and Types in Modern Greek Sentences

Katia Kermanidis, Akylini Korsavidi, Nikos Fakotakis and George Kokkinakis

Wire Communications Laboratory
Department of Electrical and Computer Engineering
University of Patras, 26500 Rio, Greece
{kerman, fakotaki, gkokkin}@wcl.ee.upatras.gr

Abstract. In this paper, we present the application of TiMBL on a Modern Greek corpus of 75,000 words for the recognition of main and secondary clauses. TiMBL is a machine learning program that stores a representation of the training examples explicitly in memory (Memory Based Learning), and classifies new cases by extrapolating from the most similar stored cases. Every training example is a vector of *n* feature-value pairs and a field containing the classification information of the vector. During classification, the *distance* of a new unclassified example *x* from every training instance *y* is computed using a similarity metric $\Delta(x,y)$. In the first stage, for the detection of clause boundaries we consider every token to be a candidate boundary. For the training instances, tokens were manually tagged depending on whether they constituted the beginning, the end, the inside of a clause or a one-word clause. In the second stage, for the recognition of clause type, clauses were tagged with one of the twelve types of Modern Greek clauses (one type for main clauses, and eleven types for secondary clauses). Both ML algorithms of TiMBL, IB1 and IGTree, were experimented with and training was performed at two levels, using the classification results of the first level for a second training process, which helps the learning program to learn from its mistakes. Using 10-fold cross validation, a recall of 87% and a precision of almost 90% was reached.

1 Introduction

In recent years, several large-scale applications in information extraction and information retrieval have shown the importance of low-level text processing such as sentence and clause boundary detection, text chunking, proper name detection etc. Unlike approaches employing manually constructed rules, corpus-based techniques for the automatic learning of linguistic knowledge have proven to be much more powerful for such tasks.

One of the learning methods is Memory-based learning (MBL), based on the hypothesis that in real world tasks reasoning is performed depending on the similarity of new situations to *stored representations of earlier experiences* [2]. Without abstraction or restructuring, training examples (feature-value vectors with associated categories) are added to memory. During classification, the similarity of a previously unseen test example to all examples in memory is computed using a similarity metric and the category of the most similar example(s) is the category of the test example. MBL has been successfully applied to a number of natural language processing tasks such as Part-of-Speech tagging [10], noun phrase chunking [8], prepositional phrase attachment [11], shallow parsing [1].

In this paper, we present our work on the automatic detection of boundaries of the clauses within a Modern Greek (MG) sentence, as well as the automatic recognition of the type of the clause (whether it is main or subordinate, etc) from corpora using MBL. This information is important for linguistic processing. Depending on their type, clauses have specific syntactic functions (they may be subjects or objects or adverbial modifiers to the verb that dominates them) and semantic properties. Obtaining this information automatically is not a trivial task, as MG clauses may be introduced by a number of types of closed-class words (called henceforth *keywords*). Many of these belong to more than one part-of-speech category and have therefore multiple syntactic roles, thereby increasing ambiguity. Due to the fact that MG is a free-constituent-order language (phrases within a sentence may appear in almost any permutation without affecting the correctness of the sentence), words introducing a clause might not even appear in the beginning of the clause. Moreover, the same keyword may introduce more than one type of secondary clause. The learning task is completed in two stages. First the boundaries of the clauses are detected. Every token of the input sentence is considered to be a candidate clause boundary. Next, the type of the clause (temporal, causal etc) is predicted.

TiMBL[1] [3] is the MBL software package we used for learning. In order to increase accuracy experiments were carried out on two levels. On the second level we added the classifications of the first level to the input data, creating a cascaded classification task. In this way we are enabling the classifier to learn from its mistakes.

The rest of this paper is organized as follows: The basic concepts of MBL are introduced in the next section along with a presentation of the main learning algorithms. The use of MBL for the Clause Boundary Detection and Clause type Recognition tasks is described in sections 3 and 4. Section 5 reports and discusses in detail the experiments that were carried out as well as the results obtained and section 6 concludes the discussion.

2 Memory-Based Learning

MBL is a form of supervised, inductive learning from *examples* [6]. Examples (also referred to as *cases* or *instances*) are represented as vectors of feature-value pairs with an associated class label. During training, a set of cases is stored in memory. During testing, a new, previously unseen example is presented to the system. Using a distance- (or similarity-) metric, the distance of the test example to all cases in memory is computed and the category of the nearest case(s) is used to predict the class of the test case. Reasoning is performed by directly reusing stored experiences

[1] TiMBL is available for research purposes at http://ilk.kub.nl

instead of applying knowledge (e.g. rules or decision trees) abstracted from experience [9].

Computational as well as memory cost can be high in MBL as nearly all computation takes place at classification time. For the task at hand, testing time is referred to in section 5. On the other hand, for a range of language learning tasks, MBL methods tend to achieve better generalization accuracies [2] than decision-tree learning where some of the training data is considered to be noise and forgotten (pruned away). For such tasks, however, it is very difficult to discriminate between noise and valid exceptions. It is important, therefore, for all training cases to be stored in memory and that no attempt be made to eliminate low frequency events.

Performance of MBL crucially depends on the distance metric used. The most straightforward distance metric, which is based on the classic *k*-nearest neighbor (*k*-NN classifier), is

$$\Delta(X,Y)=\sum_{i=1}^{n} w_i \delta(x_i, y_i) \qquad (1)$$

where *X* and *Y* are the cases to be compared and *n* is the number of feature-value pairs describing a case and w_i is a weight for feature *i*. *k* has a value of 1 throughout our experiments as it is the value most commonly used. The distance of the two cases in the value of the *i*th feature is

$$\delta(x_i, y_i)=\begin{cases} \dfrac{x_i - y_i}{\max_i - \min_i} & \text{if numerical, else} \\ & \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases} \qquad (2)$$

where $\max_i$ and $\min_i$ are the maximum and minimum numeric values of feature *i* respectively. This algorithm with equal weighting for all features is referred to as *IB1*. In the task at hand, however, all features are not equally important. The lemma of the focus word (the candidate boundary token of the sentence) for example contributes more to the classification of the instance than the lemma of two words preceding the focus word. We have experimented with all feature weighting techniques supported by TiMBL in order to discover the one that produces the best results.

- **Information Gain Weighting:** By weighting each feature with its information gain (IG) we express the average reduction in the training set information entropy, when knowing the value of the feature. Information Gain tends to overestimate the importance of features with large numbers of values. For this reason a normalized version called *Gain Ratio* (see eq.3) has been introduced which is Information Gain divided by split info *si(i)*, i.e. the entropy of the feature values.

$$w_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C \mid v)}{si_i} \qquad \textbf{(3)}$$

$$si(i) = -\sum_{v \in V_i} P(v) \log_2 P(v) \qquad \textbf{(4)}$$

where *C* is the set of class labels, V_i is the set of values *v* for feature *i* and *H(C)* is the entropy of the class labels.

- Chi-squared Weighting: Another metric used for feature weighting is the one based on the x^2 distribution and is given by eq. 5:

$$x^2 = \sum_i \sum_j \frac{(E_{ij} - O_{ij})^2}{E_{ij}} \qquad \textbf{(5)}$$

where O_{ij} is the observed number of cases with value v_i in class c_j and E_{ij} is the expected number of cases which should be in cell (v_i , c_j) in the contingency table, if the null hypothesis (of no predictive association between feature and class) is true. Weights can be corrected by using the *Shared Variance* (see eq.6), where $|C|$ and $|V|$ are the number of classes and the number of values respectively.

$$SV_i = \frac{x_i^2}{N \times \min(|C|, |V|) - 1} \qquad \textbf{(6)}$$

The IG weighted IB1 algorithm (IB1-IG) is relatively costly. For each test case, all feature values must be compared to the corresponding feature values of all the training cases. The *IGTree* algorithm, on the other hand, restructures memory in a compressed decision tree structure. Information gain is used to determine the order in which feature values are added as arcs to the tree. During testing, search can be restricted to matching a test case to the training instances that have the same feature value as the test case at the feature with the highest weight. Instance memory can then be optimized further by examining the second most important feature, then the third most important feature and so on. Thereby similar instances share partial paths.

3 MBL for Clause Boundary Detection

In MG clauses can be connected coordinatively with a coordinating conjunction like *και* (and), *ή* (or), *είτε* (either) or with subordinating conjunctions, pronouns, adverbs and particles introducing dependent clauses ([4]; [5]). Two subordinate clauses connected with a coordinating conjunction must be of the same type. A keyword which can be encountered introducing subordinate clauses may also belong to more than one part-of-speech categories and have different syntactic roles, thereby constituting an important source of ambiguity. Sometimes they are not at the beginning of the clause but they are preceded by another constituent of the sentence, for example a noun phrase. Clauses can also be embedded i.e. nested within other clauses. The above reasons show that the Clause Boundary Detection (CBD) task is far from trivial.

We have used a balanced 75,000-word Modern Greek corpus collected from articles from the newspaper *Eleftherotypia (http://corpus.ilsp.gr)*. The corpus has been automatically tagged with the boundaries of the sentences using the sentence splitter described in [7] and manually tagged with the beginning and the end of all the clauses. In Table 1 some statistical information concerning the corpus is shown. Every token of an input sentence is considered to be a candidate clause boundary. By *token* we mean words, numbers, abbreviations, acronyms or punctuation marks in the sentence. The features selected to describe the instances are

- the lemma[2] and part-of-speech[3] of the focus word, (the candidate boundary token)
- the lemma and pos of the token two positions to the left of the focus word
- the lemma and pos of the token one position to the left of the focus word
- the lemma and pos of the token one position to the right of the focus word
- the lemma and pos of the token two positions to the right of the focus word

The window size (-2, +2), regarding the number of tokens preceding and following the focus word, was selected so that cases where clauses are introduced by a combination of words can be dealt with. As shown in the following example the relative clause (the part of the sentence following the comma) is introduced by the three words *πάνω στον οποίο*, which constitute a relative phrase.

Αποδοκίμασε τον εθνικιστικό φανατισμό, πάνω στον οποίο πολλοί πολιτικοί έχουν επενδύσει το μέλλον τους.

He condemned the nationalistic fanaticism, in which many politicians have invested their future.

[2] We use the lemma information for verbs, conjunctions, pronouns and adverbs. For the remaining POS categories the lemma is not important for the task at hand and is substituted by a common symbol for all.

[3] For relative and interrogative pronouns as well as for coordinating and subordinating conjunctions information of the type of the pronoun or conjunction is included in the POS tag.

The classes used for the CBD task are *s* for the beginning of the clause, *f* for its end , - for interior tokens of the clause and *sf* for a clause consisting of only one token.

Table 1. Statistical information for the corpus.

Clause Type	Number of clauses	Percentage of clauses (%)
Main	3648	50.1
Declarative	534	7.32
Indirect Commands	888	12.18
Of Fear	4	0.05
Indirect Questions	78	1
Relative	1388	19.03
Causal	146	2
Conditional	165	2.26
Of Contrast	142	1.95
Of Result	36	0.5
Of Purpose	165	2.26
Temporal	98	1.35

4 MBL for Clause Type Recognition

Recognizing the type of a clause (CTR) can be fairly straightforward if the keyword, or keyphrase is unambiguous. It is common, however, for keywords (-phrases) to introduce more than one type of clause. Verb groups, according to the semantic category (interrogative, volitive verbs etc.) they belong to, often determine the type of the subordinate clause they dominate. The information of the verb dominating the clause is in such cases very important for disambiguating the clause type. In the following example, the keyword *να* (to) may introduce clauses of purpose (first example) or indirect commands (second example). The verb *θέλω* (to want) resolves the ambiguity in the second example as it is a verb of will.

Πήγε να φέρει το βιβλίο.
He went to get the book.

Θέλω πολύ να φύγω.
I very much want to leave .

This example shows another case where the window of size (-2, +2) is crucial, as the verb is located two positions to the left of the focus word. Intuitively, the bigger the distance of a token from the focus word, the weaker its impact to classification.

As shown in Table 1, there are 11 types of subordinate clauses in MG. At this stage, clauses in the corpus have been manually tagged according to the type of clause they belong to. More precisely, the start symbol of the previous stage has been replaced by one of 12 (1 for main and 11 for subordinate clauses) symbols. The symbols *f*, - and *sf* remain the same as in the previous stage and the same holds for the rest of the feature-value vectors.

The output of the classifier is shown in the following example. Consider the input sentence:

Για το λόγο αυτό ο ασφαλισμένος πρέπει να ζητάει αναλυτική καταγραφή για το πώς προκύπτει το ποσό που του υπόσχεται ο ασφαλιστής .

For this reason, the insured individual needs to ask for an analytical record about the way in which the amount promised to him by the insurer is determined.

After classification the above input is transformed as shown below. *kp, dbp, dplep* and *dap* are the main clause, indirect command, indirect question and relative clause start tags respectively.

Για /kp το /_ λόγο /_ αυτό /_ ο /_ ασφαλισμένος /_ πρέπει /f να /dbp ζητάει /_ αναλυτική /_ καταγραφή /_ για /_ το /f πως /dplep προκύπτει /_ το /_ ποσό /f που /dap του /_ υπόσχεται /_ ο /_ ασφαλιστής /_ . /f

5 Experiments and Results

The *Eleftherotypia* Corpus[4] is balanced, i.e. it consists of texts of different genres (news, articles, reports, interviews) and of varying domains (political, social, financial, cultural, athletic etc.) of size proportional to the distribution (relevance) of a certain text type within the Modern Greek language and created by the Institute of Language and Speech Processing (ILSP). *Eleftherotypia* is a wide-circulation newspaper the articles of which are fairly complicated in syntactical structure. The percentage of embedded clauses, for instance, is significantly high and in these cases the detection of boundaries and especially of the end of a clause becomes very complicated. For our experiments we used tenfold cross-validation. Recall and precision were defined as follows:

Recall = the number of correctly predicted boundaries (types) divided by the total number of boundaries (types) appearing in the input text.

Precision = the number of correctly predicted boundaries (types) divided by the total number of boundaries (types) predicted by the program.

Accuracy = the number of correctly predicted classes divided by the total number of classes appearing in the input text.

[4] http://corpus.ilsp.gr

Experiments were carried out in two levels. In the first, we experimented with both IB1 and IGTree algorithms for the CBD and CTR tasks. In the second series, we used cascaded processing to improve the performance of the first level by adding the classification of the first level as an extra feature to the new training instances. The rest of the feature-value pairs of the vectors remain the same as before.

5.1 IB1

In Table 2, the results are shown for the above tasks using algorithm IB1 with different weighting functions. As expected, Gain Ratio weighting produces the best scores. Scores drop in the CTR task for the number of classes increases from 4 to 15 and the number of training instances for every class decreases significantly. As can be seen, for the task at hand, the correction that variance weighting provides over x^2 weighting does not seem to have an impact on accuracy.

In Table 3, recall and precision for every clause type are presented. For the vast majority of types the results are more than satisfactory. In the case of clauses of fear, the sample in the training corpus is very small (see Table 1). For clauses, however, appearing in the training data often (e.g. main, relative, indirect commands, declarative clauses) accuracy is high. The same holds for clauses that are introduced by a small number of unambiguous keywords like relative, conditional clauses and clauses of result. Indirect questions can be easily confused with direct questions as they are introduced by the same keywords. Clauses of purpose are introduced by the particle *να* which also introduces indirect commands and clauses of result and this ambiguity is impossible to resolve without semantic processing.

5.2 IGTree

IGTree, as mentioned before, is a decision tree approximation of IB1. IGTree does not prune exceptional instances. It is only allowed to disregard information redundant for the classification of the instances presented during training. Carrying out the same experiments using IGTree, accuracy drops. This is the case because IGTree makes the assumption that differences in relative importance among features can always be exploited, which is not always true. With IB1IG, all training instances are possible sources for classification, while IGTree abstracts from low-frequency events, which is harmful in language learning tasks where irregularities are a legitimate source of information.

Training time with IGTree is slightly longer. Testing time, however, is significantly shorter since each test case does not have to be compared to all the train cases. In our experiments, testing time using IGTree is under 10 seconds, while with IB1 it is over 2 minutes.

The results of the tests with IGTree are shown in Tables 4 and 5.

Table 2. Results for CBD and CTR using IB1 for the first series of experiments with various weighting sets.

Weighting	CBD			CBD and CTR		
	Re (%)	Pr (%)	Ac (%)	Re (%)	Pr (%)	Ac (%)
None	84.7	89.7	94.8	80.6	85.6	94.0
Gain Ratio	87.2	91.1	95.7	84.4	89.0	94.8
Info Gain	86.6	90.2	95.2	83.6	87.1	94.6
X^2 distribution	85.1	88.9	94.4	82.2	86.0	93.8
Variance	85.1	88.9	94.4	82.2	86.0	93.8

Table 3. Recall and Precision for every clause type with algorithm IB1-IG.

Clause Type	Recall (%)	Precision (%)
Main	84.2	88.1
Main (1-word clause)	63.5	65.2
Declarative	91.2	91.7
Indirect Commands	81.8	89.4
Of Fear	66.7	72.6
Indirect Questions	60.1	70.4
Relative	89.2	89.5
Causal	74.3	76.2
Conditional	89.4	89.9
Of Contrast	88,5	94.0
Of Result	79.7	86.4
Of Purpose	58.8	62.1
Temporal	72.6	85.4

Table 4. Results using IGTree for the first series of experiments with various weighting sets.

Weighting	CBD			CBD and CTR		
	Re (%)	Pr (%)	Ac (%)	Re (%)	Pr (%)	Ac (%)
None	73.4	86.1	91.3	67.1	83.1	90.5
Gain Ratio	79.6	88.9	93.7	77.8	87.4	93.1
Info Gain	79.2	88.1	93.4	77.8	85.6	92.5
x^2 distribution	79.5	88.3	93.4	77.4	84.5	92.2
Variance	79.5	88.3	93.4	77.4	84.5	92.2

Table 5. Recall and Precision for every clause type with algorithm IGTree.

Clause Type	Recall (%)	Precision (%)
Main	74.5	88.1
Main (1-word clause)	32.5	47.8
Declarative	92.2	95.3
Indirect Commands	88.9	83.3
Of Fear	66.7	72.6
Indirect Questions	40.5	74.1
Relative	88.7	93.3
Causal	79.6	79.1
Conditional	87.7	92.4
Of Contrast	93.5	94.6
Of Result	84.0	91.4
Of Purpose	51.5	66.1
Temporal	70.7	88.1

5.3 Second Training Level

As mentioned before, a second series of experiments was carried out in order to improve the results obtained in the first series. The main idea is to enable the classifier to learn from its mistakes. The class label predicted in the first training level is added as a new extra feature to the feature-value vectors, creating thereby a new set of training instances. To be more precise, each instance is enriched with five more features: the predicted class label of all five tokens constituting the vector. In this way, information relevant to the classifier is added to the training file making it possible for the learning program to detect typical errors and rectify them. Especially recall values are improved, which indicates that more clauses are correctly detected than in the first level. These improvements are shown clearly in Table 6 and detailed results for all clause types in Table 7.

5.4 Discussion

Throughout all of the experiments, precision values were higher than recall, indicating that strict restrictions for the classification of a new case to a particular class are set by the training examples. This could be attributed to the fact that the contribution of the lemma of the focus word to classification is very important. The use of a larger and more complete training set would probably have led to looser classification criteria.

Table 6. Recall and Precision for CBD and CTR with IB1IG and IGTree in one and two levels.

	Recall (%)	Precision (%)	Accuracy (%)
IB1IG1	84.4	89.0	94.8
IB1IG2	87.0	89.8	95.5
IGTree1	78.1	87.4	93.1
IGTree2	82.2	88.9	94.4

As mentioned earlier, CBD can de difficult when clauses are embedded. As an example, in the following sentence, the boundaries of the embedded relative clause, introduced by *που,* cannot be detected. *Detp* is the start tag for the causal clauses.

[Αναγνώρισε το τραγικό λάθος]<kp> [στο οποίο είχε περιέλθει]<dap> [αφού αυτοί [που βρίσκονται κάτω από το όριο της φτώχειας στη χώρα μας]<dap> είναι τελικά οι εισοδηματίες.]<detp>

He realized the tragic mistake that he had come to, for those who live under the poverty line in our country are in the end the gentlemen at large.

It was observed, however, that in cases where the embedded clause has a subject position (precedes directly the verb of the embedding clause) correct boundary detection is achieved:

[Ο πρόεδρος είπε] <kp> [οτι [όποιος θέλει,] μπορεί [να θέσει υποψηφιότητα.]]
The president said that anyone who wants to can be a candidate.

In cases where two subordinate clauses are connected with a coordinating conjunction and the keyword introducing the second clause is missing, CTR is not straightforward. In the following example, the coordinating conjunction is *και.* The conjunction *οτι* that actually introduces the second clause is omitted (is therefore in parentheses) and the type of the second clause cannot be determined.

Ο υπουργός είπε οτι το νομοσχέδιο που πρόκειται να κατατεθεί είναι πολύ σημαντικό και (οτι) όλοι πρέπει να το ψηφίσουν.

The minister said that the draft of law, which is to be presented, is very important and (that) everyone should vote for it.

Conclusion

Clause Boundary Detection and Type Recognition provide valuable information that is very important for the following stages of a language processing system. In this paper, we have shown that MBL is able to deal competently with these tasks by automatically learning the appropriate knowledge from Modern Greek newspaper corpora. Complicated syntactic phenomena are discussed and the way they are dealt with by the classifier. IB1IG, by taking into consideration all the training instances, seems to outperform IGTree. Cascaded training was employed to improve performance even further.

Table 7. Recall and Precision for every clause type with both algorithms at the second level.

Clause Type	IB1IG2		IGTree2	
	Re (%)	Pr (%)	Re (%)	Pr (%)
Main	85.5	89.4	76.3	89.8
Main (1-word clause)	73.6	76.1	52.5	69.2
Declarative	92.6	93.3	92.8	95.8
Indirect Commands	94.4	90.5	91.4	90.7
Of Fear	66.7	76.2	66.7	76.2
Indirect Questions	65.1	73.3	50.8	65.7
Relative	89.9	90.2	88.7	93.9
Causal	75.8	79.2	80.2	83.9
Conditional	89.6	91.9	87.0	92.8
Of Contrast	89.8	92.7	93.5	94.8
Of Result	80.2	86.6	84.7	91.8
Of Purpose	64.5	72.3	53.4	70.8
Temporal	73.1	81.9	71.8	88.5

References

1. Daelemans W., Buchholz S. and Veenstra J.: Memory-Based Shallow Parsing. *Proceedings of CoNLL-99*, Bergen, Norway (1999).
2. Daelemans W., van den Bosch A. and Zavrel J.:. Forgetting Exceptions is Harmful in Language Learning. *Machine Learning*, 34, pp. 11-41 (1999).
3. Daelemans W., Zavrel J., van der Sloot K. and van den Bosch A.: TiMBL: Tilburg Memory Based Learner, version 3.0, Reference Guide. *ILK Technical Report 00-01* (2000).
4. Karanikolas A. 1997. Modern Greek Syntax. (in Greek). Organisation of Publishing of Educational Books, Athens.
5. Kosma D., Sarischouli S. 1988. Analytical Modern Greek Grammar and Syntax.(in Greek).
6. Mitchel T. 1997. Machine Learning. McGraw –Hill International Editions.
7. Stamatatos E., Fakotakis N. and Kokkinakis G. 1999. Automatic Extraction of Rules for Sentence Boundary Disambiguation. Proceedings of the Workshop in Machine Learning in Human Language Technology, (ACAI'99) Chania, Greece.
8. Veenstra J. and Buchholz S. 1998. Fast NP Chunking Using Memory-Based Learning Techniques. *Proceedings of Benelearn 1998,* Wageningen, The Netherlands, pp. 71-79.
9. Zavrel J. and Daelemans W. 1997. Memory-Based Learning: Using Similarity for Smoothing. *Proceedings of the 35th Annual Conference of the Association of Computational Linguistics, ACL,* Madrid.
10. Zavrel J. and Daelemans W. 1999. Recent Advances in Memory-Based Part-of-Speech Tagging. *VI Simposio Internacional de Comunicacion Social*, Santiago Cuba, pp. 590-597.
11. Zavrel J. Daelemans W. and Veenstra J. 1997. Resolving PP attachment Ambiguities with Memory-Based Learning. *Proceedings of the Workshop on Computational Language Learning,* Madrid.