2009 Ninth International Conference on Intelligent Systems Design and Applications

Evolutionary Shallow Parsing

John Atkinson
Department of Computer Sciences
Universidad de Concepcion,
Concepcion, Chile
atkinson@inf.udec.cl

Juan Matamala
Department of Computer Sciences
Universidad de Concepcion,
Concepcion, Chile
atkinson@inf.udec.cl

***Abstract*—In this work, a new approach to natural-language chunking using an evolutionary model is proposed. This uses previously captured training information to guide the evolution of the model. In addition, a multi-objective optimization strategy is used to produce the best solutions based on the internal and the external quality of chunking. Experiments and the main results obtained using the model and state-of-the-art approaches are discussed.**

***Keywords*-Natural Language Parsing; Chunking; Evolutionary Computation**

I. Motivation

The most basic form of syntactical processing known as *Parsing*, is the process of analyzing a sequence of tokens (i.e., morphosyntactic or lexical categories) to determine its grammatical structure with respect to a given formal grammar. Traditional full parsing aims to provide as detailed as possible analysis of the sentence structure. Full parsing is a challenging task involving the development of the full grammar for the language as well as the computational challenges involved in identifying the most plausible parse of a given sentence. However many natural-language processing (NLP) applications do not necessarily require a complete syntactic analysis. On the other hand, parsers can usually generate multiple syntax trees for the same input text leading to ambiguity and efficiency problems. Many of the these tasks can adequately be performed by identifying the noun phrases (NP), verb phrases (VP), etc and the relationships between these entities. Shallow (or partial) parsing can be used to recover some limited syntactic information from natural language sentences [2]. This often involves *chunking* which refers to the process of unnested grouping the words into chunks given their morphosyntactical tags [2].

This work proposes a new approach to chunking which makes good use of the search capabilities of a *Genetic Algorithm* (GA) and previous training data obtained from annotated corpus. We hypothesize that an evolutionary model for chunking can produce competitive results when comparing with other state-of-the-art techniques.

This research is partially sponsored by the National Council for Scientific and Technological Research (FONDECYT, Chile) under grant number 1070714 *"An Interactive Natural-Language Dialogue Model for Intelligent Filtering based on Patterns Discovered from Text Documents"*

II. Related Work

Natural-language parsing involves the procedure of bringing basic morphosyntactic categories into high-level syntactic relationships with each other. This is probably the most commonly encountered form of corpus annotation after *Part-of-Speech* (POS) tagging (aka. lexical tagging). Usually a parser looks for valid tree parses for an input natural-language sentence. Since this must recursively find different structures, techniques usually have serious efficiency problems in analyzing massive amounts of texts.

To reduce the search space and resolve ambiguity issues, partial (or shallow) analysis for some specific syntactical groups of the input sentence can be carried out. The most popular shallow parsing strategy is known as *Chunking* which identifies the non-recursive cores of various phrase types in text, possibly as a precursor to full parsing or information extraction. The paradigmatic shallow parsing problem is NP chunking, which finds the non-recursive cores of Noun Phrases.

Computationally, text chunking consists of dividing a text in syntactically correlated parts of words. For example, the sentence "`He reckons the current account deficit will narrow to only 1.8 billion in September.`" can be divided into chunks as follows:

```
[NP He][VP reckons][NP the current
                account deficit]
[VP will narrow] [PP to] [NP only
                1.8 billion][PP in]
[NP September] .
```

State-of-the-art chunking techniques can be divided into two types: those based on grammar rules and those using supervised machine-learning techniques [3], [2]. Methods using explicit rules for chunking are not accurate when dealing with huge amounts of texts. Furthermore, this is an extremely time-consuming task as these rules are usually manually built. An early significant improvement to chunking was achieved by using *Memory-based Learning*

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.132

(MBL) in which the parser automatically learns the classification rules based on previously annotated examples. One of the problems with this approach is the availability and preparation of examples which is an intensive task. On the other hand, the search algorithm used for finding chunks is based on similarity measures which restrict the type of chunk pattern to be looked for in the training corpus. A more flexible and adaptive rule-based technique for chunking is based on *Transformation-based Learning* (TBL). This was originally applied to lexical, syntactical and semantic tagging but further adapted to shallow analysis. Initially, TBL hypothesizes a set of very simple learning rules. The technique then iteratively looks for rules that better correct the errors made by the previously proposed rules. Despite its high computational cost to validate the rules in each iteration, variations of the strategy have incorporated restrictions on the search space for each iteration to improve the chunker performance [4]. Statistical-based methods such as *Maximum Entropy* parsers ($MaxEnt$) have also been used to address these issues. A related method which uses probabilities for prediction and classification producing better performance by reducing the number of required training data is based on *Hidden Markov Models* (HMM) in which hidden states of the model keep likely chunk tags and visible states contain the part-of-speech (POS) or lexical tags [5]. An important drawback is that as the number of hidden states grows the method becomes less effective in exploring the search space than other techniques such as Support Vector Machines (SVM) [4], [6], [2], [7]. Non-traditional techniques such as evolutionary computation have also been used in related tasks such as POS tagging, and text mining. Performance is observed to be similar to that of traditional methods, however exploring the space of hypotheses becomes more robust.

III. A New Approach to Natural Language Chunking based on Genetic Algorithms

This work proposes a new approach to natural-language chunking using evolutionary computation techniques (i.e., *Genetic Algorithms* which have demonstrated substantial improvement over a variety of random and local search methods [8].

Our GA-based chunking model receives a natural-language text and assigns the corresponding syntactical chunks for each sentence based on previously computed training information from annotated corpus. The model can be divided into two phases: training and chunking.

From a set of natural-language scientific texts (training texts), the training task computes probabilities of POS tags and chunks based on an n-gram language model. The model is then capable of receiving new natural-language documents and have them chunked. Specifically, the training phase extracts two kinds of underlying information from the annotated texts:

1) *Associations between lexical and chunk tags:* based on Bayesian models, information regarding the likely associations between POS tags and chunks is captured.
2) *Sequencing data:* based on a statistical language model, sequences of n-grams are obtained using Hidden Markov Models.

Resulting training information is further used to guide the GA that automatically assigns chunks to new natural-language texts. The initial population for the GA is generated by randomly combining candidate chunks.

In order to assess the quality of the generated hypotheses, a multi-objective optimization was applied to determine how good a sequence of chunks is for a sentence based on intra-chunk and inter-chunk associations. This is carried out using structural and lexical information obtained from an annotated corpus.

A. Training

In order to carry out the chunk parsing, an annotated corpus of natural-language texts was used. This contains sentences annotated with standard POS and chunk tags. For capturing training data, statistical language models were applied to extract knowledge which will guide the GA. In particular, n-gram language models were adapted and used.

A n-gram model predicts the occurrence of a symbol in a sequence based on the $n-1$ previous contexts, in words, this makes explicit the structure of the symbols. For our approach, a 2-gram (aka. bi-gram) language model was applied to the training corpus with the symbols being the chunk tags. The benefit of bi-gram modeling is based on the assumption that there is a relevant connection between contiguous chunks in a sentence.

The bi-gram model computes the probability of a sequence of chunks c based on the previous context: $P(c) = P(c_1)\prod_{i=2}^{n} P(c_i \mid c_{i-1})$. Therefore, the probability of assigning a chunk c_i (of a sentence) given a previous chunk c_{i-1} is computed as: $P(c_i \mid c_{i-1}) = \frac{N(c_{i-1},c_i)}{N(c_{i-1})}$, where $N(c_{i-1}, c_i)$ is the number of occurrences of sequence of chunks (c_{i-1}, c_i) within the training corpus, and $N(c_{i-1})$ is the number of occurrences of chunk c_{i-1} within the same corpus.

B. Evolutionary Chunking

In our evolutionary approach, chunk classification can be seen as an optimization problem in which the best chunk tags should be assigned to the words of a sentence. For this, the GA requires new representation schema, genetic operators and evaluation metrics for the hypotheses.

An initial population is created containing a predefined number of individuals (chunked sentences), each represented by a genetic string (incorporating the variable information). Each individual has an associated fitness measure, typically representing an objective value. The concept that fittest (or best) individuals in a population will produce fitter offspring

is then implemented in order to reproduce the next population. Selected individuals are chosen for reproduction (or crossover) at each generation, with an appropriate mutation factor to randomly modify the genes of an individual, in order to develop the new population. The result is another set of individuals based on the original subjects leading to subsequent populations with better (min. or max.) individual fitness. Therefore, the algorithm identifies the individuals with the optimising fitness values, and those with lower fitness will naturally get discarded from the population.

Hypothesis Representation

In order to code each chromosome representing a sequence of chunks, a three-dimensional structure is used. This contains the words of a sentence, their POS tags and their automatically assigned chunks. Note that as the GA goes on, the words and POS tags remain unchanged for a chromosome. An example of the representation of the hypothesis for the sentence below can be seen in figure 1:

```
He reckons the Current account deficit
will narrow to only #1.8 billion in
September
```

where the chromosome shows the 3-dimension structure containing components extracted from a standard *Wall Street Journal* (WSJ) corpus (an empty gene shows that the previous chunk is kept).

At the beginning, a sequence of words of a sentence is assigned a random chunk. As the GA goes on, the size of this chunk becomes bigger or smaller based on the quality of the assigned chunk. Genetic operators can modify the content or the size of the chunk hence its structure may vary accordingly. However, the size of the chromosome is fixed as a word must always be tagged with a chunk.

An initial population of hypotheses for the GA is created from a set of randomly generated combinations of chunks for each word of an input sentence. Overall, chunks can be assigned from a set of 36 types of chunks extracted from the standard *Penn Treebank II tagset*[1]. Note that dimensions 1 and 2 remain unchanged whereas the dimension representing the chunk depends on the assigned chunks. Each chunk covers segments of the input sentences from 1 (one word) to the length of the sentence.

Fitness Evaluation

In order to automatically assess the quality of the chunks generated for each individual, a fitness function is proposed. For our model, the evaluation considers two objective functions:

1) *Quality of the sequence of chunks (aka. inter-chunk objective):* assesses the structure of the sequence of chunks for a sentence based on the frequencies of chunks obtained from training data.

2) *Quality of an individual chunk (aka. intra-chunk objective):* assesses the sequence of POS tags for each individual chunk of a sentence based on the frequencies of lexical tags assigned to chunks in the training data.

Both objectives are computed by using data generated from a training annotated corpus. In order to compute the intra-chunk measure (F_{intra}), sequences of n-grams are obtained using HMM. This considers probabilities of POS tags associated to chunks. This is, the probability that a POS tag is associated to a chunk is computed for all the POS tags of a sentence. The rationale for this is that some POS tags are assumed to be more likely to occur within a chunk than other. For each gene, the metrics calculates $P(POS_i/Chunk_j)$ which represents the conditional probability that POS_i tag occurs given that the current chunk is $Chunk_j$. Next, the resulting intra-chunk measure considers how likely the sequence of pairs (POS tags and Chunks) is for the sentence of $n-1$ words as: $F_{intra} = \prod_{j=1}^{N_c} \sum_{i=0}^{L(j)-1} P(POS_i/Chunk_j)$, where N_c is the number of chunks of a sentence and $L(j)$ is the length of the chunk j.

Computating the objective value of the example is as follows: for the inter-chunk measure (F_{inter}), the objective value is obtained by computing the probability of the sequence of chunks for the input sentence and having then multiplied so to obtain an objective function value. For example, assume the following sequence of chunks in a sentence: "$NP \mid VP \mid NP \mid VP \mid PP \mid NP \mid PP \mid NP$". For the first chunk ($NP$), the probability of this being the start of the sentence is first computed. Next, the probability that the second chunk being VP giving that the previous one was NP, and so on. The product of these values represents the inter-chunk fitness for a sequence of chunks.

In order to compute a unique fitness value, several evolutionary multi-objective optimization techniques including aggregation (i.e., weighted sum of the objective functions), SPEA-II (the improved *Strength Pareto Evolutionary Algorithm*) and the *Non-Dominated Sorting Genetic Algorithm* (NSGA-II) were assessed [9]. However, the *Precision* and *Recall* values of the model using NSGA-II were observed to outperform the other methods hence it was used for the current experiments.

The overall fitness of an individual which considers the objective functions above is computed by a multi-objective optimization strategy based on the *NSGA-II* algorithm [9]. This uses a ranking method that emphasizes on the good solution points and tries to maintain a population of such points throughout the procedure. NSGA-II maintains diversity in its population by a crowding method, which eliminates focusing on certain regions of the solution space, and explores different regions in the Pareto front. The concept of non-dominated sorting is underlined by the ranking selection method which keeps track of the good solution points, and

[1] http://www.cis.upenn.edu/~ treebank/

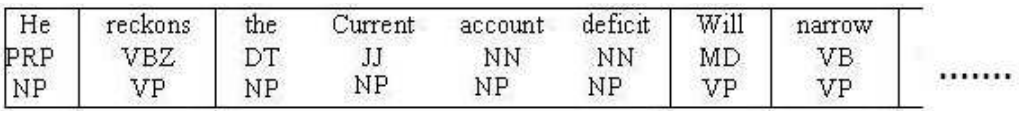

Figure 1. Example of a Multi-dimensional Representation for a Chromosome

the niche method which maintains stable subpopulations of these solutions [9].

Genetic Operators

In order to modify and improve the individuals of the population, new genetic operators have been designed including selection, crossover, and mutation.

- **Selection**
  A selection operator picks up the best hypotheses to be reproduced based on the fitness values. For this model, a binary tournament selection strategy was applied [8]. This randomly selects pairs of individuals from the whole population and those having highest fitness values are considered for reproduction.
- **Crossover**
  A new constrained crossover operator was designed to exchange chunks from two different hypotheses of the population. For this, two random individuals are randomly selected, a single crossing point is chosen and the chunks are exchanged. The single-point crossover exchanges chunks from two selected individuals at a ramdonly selected point and generates two new offspring according to a crossover probability P_c. Since that only valid hypotheses must be generated, if the crossing point falls within a chunk's segment, the operator automatically corrects the resulting crossover so to obtain valid chromosomes. Consequently, the size of the chunk may vary as a result of a new change in a word's assigned chunk.
- **Mutation**
  For the mutation operator, a random modification of each chunk is made for a sequence of words with a mutation probability (P_m) using a pool of available chunks.

The overall structure of the GA for evolutionary chunking using these new operators can be seen at figure 2.

The first step is to generate the initial population of sequences of chunks for an input sentence. Next, fitness evaluation using the NSGA-II algorithm is used to rank the population on the basis of their Pareto dominance (i.e., the fitness is proportional to the number of solutions dominated by each hypothesis). A large dummy fitness value is assigned to all the non-dominated individuals of the population, after which they are shared. In the sharing procedure, a selection operation divides the original fitness value by a number proportional to the size of that group. Then these individuals are temporarily kept aside, and the rest of the population is ranked in a similar manner, assigning lower dummy fitness values, as the procedure proceeds. This dummy fitness value plays an important part in the reproduction of the next generation, and intuitively, individuals with larger fitness values produce more offspring than the rest of the population. Hence, NSGA-II reduces multiple objectives into a single dummy fitness function, which allows it to work with two objective functions for chunking. Once the population has been evaluated hypotheses are selected for reproduction based on a binary tournament as previously explained. Next, crossover and mutation operations are applied to the selected chromosomes. Finally, the new offspring replaces the previous individuals to generate the next set of hypotheses.

IV. Experiments and Results

In order to assess the effectiveness of the proposed evolutionary model for chunking, a series of experiments was carried out. A first part aimed to tune the different parameters of the GA and a second part assessed how accurate the model was compared to other state-of-the-art approaches.

The different experiments were based on the standard annotated corpus *Wall Street Journal* (WSJ) from which chapters 15 to 18 were used for training (200.000 words) and chapter 20 was used for testing (50.000 words). This corpus contains English sentences annotated with POS tags. These sentences also contain their corresponding chunks which are later removed for testing purposes. Approximate chunk distribution for the WSJ corpus is as follows: 51% for *Noun Phrase* (NP), 20% for *Verb Phrase* (VP), 20% for *Prepositional Phrase* (PP), 4% for *Adverb Phrase* (ADVP), and minor proportions for the other syntactical groups. For evaluation and comparison purposes, the methods only use the NP chunk as this becomes one of the most frequent, useful and popular chunks for several current chunkers.

A. Tuning the Model

In order to adjust the evolutionary model, different configurations for the GA were evaluated using the training data from WSJ. These considered experiments aiming at analyzing the robustness of the method under different settings such as the population size ($PopSize$), the number of generations ($NumGen$), and the probabilities of crossover (P_c) and mutation (P_m), respectively. Tests are carried out by considering the best of six runs of the GA for every setting. The best configurations were determined by using standard performance metrics of *Precision* (P),

```
Generate an initial population from random sequences of chunks
Evaluate fitness   based on NSGA-II using objective values
                (intra-chunk and inter-chunk)
t=0
While (max. number of generations is not achieved)
   Select hypotheses for reproduction based on tournament
   Apply Crossover operator with P_c
   Apply Mutation operator with P_m
   Evaluate fitness based on NSGA-II using objective values
   (intra-chunk and inter-chunk)
   Generate next population from modified chromosomes
   t = t + 1
End-While
```

Figure 2. Multi-Objective GA for Chunking

Recall (R) and the $F - score$. Final experiments suggested that best performance ($F - score > 0.92$) is achieved for $PopSize = 800$, $NumGen = 800$, $P_m = \%1$ and $P_c = \%100$.

B. Effectiveness of the Model

In order to assess the relative effectiveness of our evolutionary model for chunking, the results obtained for the different real experiments were compared with those of the best competitive state-of-the-art chunking techniques using chapter 20 of WSJ as test data. The main results of the NP chunkers can be seen in table I.

The results of our evolutionary model for NP chunking can be seen in table II using different population sizes. Parameters of the GA considered $P_c = 1.0, P_m = 0.01$ on the WSJ corpus and produced a *Precision* of 92.5% and a *Recall* of 93.1% ($F - score = 92.7\%$). This shows the promise of the method when comparing with current methods. Note that most of the current methods requires a significant training corpus to produce these results.

Num	PopSize	Precision	Recall	F-score
1	400	89.3%	87.2%	88.2%
2	820	92.3%	89.1%	90.6%
3	850	92.5%	93.1%	92.7%

Table II
PERFORMANCE OF OUR MODEL FOR NP CHUNKING

The table suggests that the effectiveness of our model might not be significantly affected when reducing the size of the training corpus. In order to investigate the robustness of model on different sizes of training corpus, further experiments were carried out. The rationale for this is that creating training corpus for chunking is an extremely costly task. In addition, current techniques strongly depend on this corpus to produce better results. Hence methods which are less dependent on the size of the training data are preferred.

The model was trained by using the whole training data as used by other chunking techniques (table I) and the results are shown in figure 3 in terms of the $F - score$ metrics. The GA considered the parameters $PopSize = 800$, $NumGen = 800, P_c = 1.0, P_m = 0.01$.

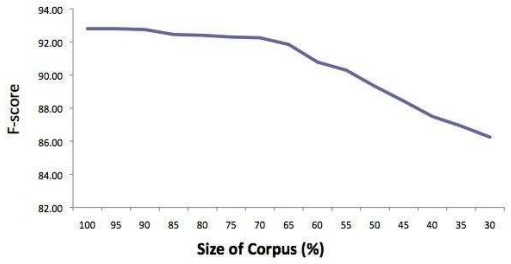

Figure 3. Performance versus size of training data

The graphic shows that as the size of the training corpus slightly decreases, the effectiveness does not show a significant drop for $F - score$. Significant decreases are not observed until reducing 70% of the training corpus. This suggests that the model is not highly dependent of the size of the corpus so in terms of used resources this may be even more efficient than some of the state-of-the-art chunking methods.

Despite outperforming some of the current techniques, the lower effectiveness when compared with the best of table I may be due to the distribution of the tags in the training corpus. This may not directly affect the generation of hypotheses, but the way the fitness evaluation is computed.

V. CONCLUSIONS

In this work, a new evolutionary model for natural language chunking was proposed. The model uses Genetic

Technique	Precision	Recall	F-score
Memory-based Learning	94.04%	91.00%	92.50%
Support Vector Machine	93.89%	93.92%	93.91%
Hidden Markov Model	92.30%	92.68%	92.49%
Transformation-based Learning	91.80%	92.30%	92.05%

Table I
PERFORMANCE OF SOME STATE-OF-THE-ART NP CHUNKING APPROACHES

Algorithms and multi-objective optimization techniques in order to assess candidate solutions in terms of quality metrics involving intra-chunk and inter-chunk probabilistic associations. A training corpus of annotated sentences with lexical and syntactical tags is used to capture training information that guides the GA toward the best solutions.

Experiments for NLP chunking, show the promise of the GA-based model for chunking syntactical groups such as noun phrases from a corpus of unseen natural-language texts. Compared with state-of-the-art chunking techniques, the performance of the model makes it very competitive. In addition, settings involving reductions on the amount of required training corpus also show that the model's performance does not significantly drop when reducing the size of the training data. That is, the method may require less training corpus to achieve the previous results which is a practical advantage as annotating training corpus for text analysis purposes is an extremely demanding task.

REFERENCES

[1] Y. Tsuruoka and J. Tsujii, "Chunk parsing revisited," *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT 2005)*, pp. 133–140, 2005.

[2] X. Li and D. Roth, "Exploring evidence for shallow parsing," *Proc. of the Annual Conference on Computational Natural Language Learning, Toulouse, France*, pp. 1–7, 2001.

[3] S. Argamon and I. Dagan, "A memory-based approach to learning shallow natural language patterns," *Journal of Experimental and Theorethical Artificial Intelligence*, vol. 10, pp. 1–22, 1999.

[4] T. Kudo and Y. Matsumoto, "Chunking with support vector machines," *Proc. of NAACL 01*, 2001.

[5] A. Molina and F. Pla, "Shallow parsing using specialized HMM," *The Journal of Machine Learning Research*, vol. 20, no. 2, pp. 595–613, 2002.

[6] H. Takamura and Y. Matsumoto, "Feature space restructuring for SVMs with application to text categorization," in *Proceedings of EMNLP-01, 6th Conference on Empirical Methods in Natural Language Processing*, L. Lee and D. Harman, Eds. Association for Computational Linguistics, Morristown, US, 2001, pp. 51–57.

[7] E. Tjong and K. Sang, "Memory-based shallow parsing," *The Journal of Machine Learning Research*, vol. 20, no. 2, pp. 559–594, 2002.

[8] K. DeJong, *Evolutionary Computation*. MIT Press, 2004.

[9] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley and Sons, 2001.