PUC-Rio - Certificação Digital Nº 0921334/CA

5 Experiments

In this chapter, we present our obtained results for both the application of a chunk feature to other NLP problems and the automatic extraction of chunks employing ETL models. The discussion is also incremented with details about the division of the used corpus into training, development and test sets, evaluation metrics, configuration settings for the models and discussions concerning the observed results.

5.1 Application of a chunk feature

As described in chapter 3, we add a chunk feature to ETL-based systems that solve the clause identification and dependency parsing problems. These systems have a fixed configuration consisting of a set of other features and ETL settings, and those are kept the same between the different runs that test each chunk definition for each problem. Because the mentioned systems also use data derived from the *Bosque* corpus, we are able to extract golden chunk values for their training datasets.

5.1.1 Clause identification

For the clause identification problem, we use the three-step model described in the work of Fernandes et al. (23). This model is similar to the subtasks classifier we dissect in chapter 4. Before extracting full clauses, it first identifies start and end clause boundaries. It then classifies every pair of start and end tokens as valid or not valid through another learning approach using ETL.

The performance of this system is measured through the $F_{\beta=1}$ metric. $F_{\beta=1}$ is the harmonic mean of two other rates: precision and recall. Precision is the rate between the number of correct items the system extracted and all items, correct or incorrect, extracted by the system. Recall is the rate between the number of correct items the system extracted and all existing items. $F_{\beta=1}$ is a kind of *F-measure*, which are weighted harmonic means of precision and recall; since in $F_{\beta=1}$ precision and recall have the same weight, it is calculated through the following formula:

$$F_{\beta=1} = \frac{2 \times precision \times recall}{precision + recall}$$

We display the results for each chunk definition applied, as well as a run with no chunk feature used, in table 5.1.

Chunk definition	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
No chunk	77.84	59.24	67.28
(NP, VP)	82.30	65.28	72.81
(NP, VP, PP)	84.42	66.07	74.13
(NP, VP, PP, ADVP, ADJP)	82.34	62.52	71.07

Table 5.1: Impact of chunk definitions on clause identification performance

5.1.2 Dependency parsing

The work of Crestana (17) details Machine Learning approaches to dependency parsing that use a special tagging style. This style is what allows the problem to be treated as a classification problem. We evaluate the influence of a chunking feature on this problem using the ETL-based direct classifier also proposed in that work.

Crestana's research focus on the identification of tokens' heads using this classification method. The metric used to evaluate this system is the unlabeled attachment score (UAS), which is simply the percentage of tokens for which the system predicts the correct head.

Table 5.2 shows the scores achieved by this system for each chunk definition.

Chunk definition	**UAS**
No chunk	87.50
(NP, VP)	89.04
(NP, VP, PP)	88.68
(NP, VP, PP, ADVP, ADJP)	88.70

Table 5.2: Impact of chunk definitions on dependency parsing performance

It is noticeable that distinct chunk definitions have a different impact on the tested problems. For instance, clause identification benefits slightly more from the (NP, VP, PP) definition, while the (NP, VP) definition results in the best performance for dependency parsing. Nevertheless, the results confirm that a chunk feature is valuable for both problems, since there is a

PUC-Rio - Certificação Digital Nº 0921334/CA

significant performance difference between the model trained with no chunk feature and the ones trained with it. Thus, they also encourage the research on the application of a similar attribute to systems targeted at other NLP tasks for Portuguese.

Depending on the time and memory limitations of an automatic extractor that takes advantage of a chunk feature, one possibility to be considered is the use of all available chunk definitions as separate features. This way, the system can automatically discover the best patterns from any of the definitions.

5.2 Chunk extractors

This section details our chunk extraction models. First, we discuss how the *Bosque* corpus is divided for our experiments. We proceed to show our results and compare them to the performance of another model applied to the same task. Finally, we analyze the obtained results more thoroughly.

The evaluation metric we apply to the chunking problem is $F_{\beta=1}$, the same used for clause identification.

5.2.1 Dataset description

Since our implementation of the chunk extraction heuristic targets the *Bosque* corpus, we also use this corpus to train and evaluate our ETL models. As is common when developing Machine Learning-based systems, the corpus is divided into three parts: the training set, the development set and the test set.

The training set, as its name implies, is the corpus used during the learning phase. The development set is used to tune the parameters of our models. Using a corpus other than the test one for parameter tuning allows us to assess the generalization capability of our model, since the chosen parameters may improve the results for a given set only because of particular characteristics of this set. Therefore, we eventually apply our model to the chosen test set and collect the final result metrics from it.

The sentences that compose each of the aforementioned sets are chosen randomly. We establish the following size proportion: 70% of the sentences from *Bosque* belong to the training set, 15% to the development set and the remaining 15% to the test set.

Our models are trained for all three derived chunk definitions mentioned in chapter 3, and results are reported for each definition. We apply our

PUC-Rio - Certificação Digital Nº 0921334/CA

derivation heuristic to generate the three corresponding sets of chunk tags for every dataset.

We do not use the revised POS tags provided with the *Bosque* corpus. To obtain results that are be more compatible with real situations, where such a golden POS tag feature would not be available, we run a state-of-the-art POS tagger (22) beforehand for the whole corpus. This POS tagger reportedly achieves an accuracy of over 96%. The tags derived by this system are described in appendix A.

5.2.2 The OpenNLP model

To have a reliable way of determining the effectiveness of our models, we also run a chunking system for Portuguese from the OpenNLP project. This project has recently shown an effort on developing a Machine Learning model for Portuguese text chunking using the chunk derivation heuristic described in this work, which has been preliminarily presented by Fernandes et al. (23).

The OpenNLP chunking model has been built on a maximum entropy framework (3), inspired by Ratnaparkhi's research (52). In addition, the model uses features proposed by Sha et al. for their Conditional Random Fields approach (54).

When training and testing the OpenNLP model, we use the same corpora used for our own proposed models to guarantee the comparability between them. Moreover, the model is trained using the following configuration parameters:

- Number of iterations: 100
- Cutoff value: 5

5.2.3 Direct classifier

For the direct classifier, we use an ETL window size of 3, a rule score threshold of 2, and determine that all rule templates have a minimum of 2 and a maximum of 7 features.

The way we decide on these parameters is by testing a whole range of possibilities for each configuration on our development set, and then choosing the most effective combination. This combination is then used to extract the target attribute from the test set.

We train the direct classifier using the ETL template evolution technique, to make the learning phase faster.

PUC-Rio - Certificação Digital Nº 0921334/CA

PUC-Rio - Certificação Digital Nº 0921334/CA

5.2.4 Subtasks classifier

Both chunk start and chunk end subtasks' models have the following configuration: ETL window size of 3, rule score threshold of 3, and minimum of 2 and maximum of 7 features for rule templates. Again, many parameter variations have been previously assessed before the one mentioned was settled.

We use template evolution for the chunk end subtask model, but not for the chunk start one. This is mainly because we verify that the chunk start task has less memory requirements than the chunk end model, so training without template evolution is feasible.

Table 5.3 demonstrates the number of generated templates for each task to which we apply an ETL model: the direct classifier and the start chunk and end chunk tasks of the subtasks classifier. Also, table 5.4 shows the total number of learned rules based on those templates for each system.

		ETL systems	
Chunk definition	**Direct**	**Chunk start**	**Chunk end**
(NP, VP)	602	682	617
(NP, VP, PP)	534	839	456
(NP, VP, PP, ADJP, ADVP)	577	875	451

Table 5.3: Number of generated templates for each ETL system

		ETL systems	
Chunk definition	**Direct**	**Chunk start**	**Chunk end**
(NP, VP)	2448	596	893
(NP, VP, PP)	1969	632	650
(NP, VP, PP, ADJP, ADVP)	1900	771	599

Table 5.4: Number of learned rules for each ETL system

5.2.5 Results analysis

In tables 5.5, 5.6 and 5.7, we present the results of our ETL models and their corresponding baseline systems for the chunk start and chunk end subtasks. The tables correspond to the results for the (NP, VP), (NP, VP, PP) and $(NP, VP, PP, ADJP, ADVP)$ chunk definitions, respectively. We remind that these baseline systems are based on the frequency of association of the tags for each POS tag in the training data, and that they are used as an initial classifier before ETL starts learning its correction rules. The lists of associations between POS tags and target tags that compose those systems are shown in appendix B.

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Chunk start	92.45	87.60	89.96
Start BLS	88.70	62.40	73.26
Chunk end	88.39	86.16	87.26
End BLS	72.09	51.41	60.02

Table 5.5: Subtasks performance for (NP, VP) definition

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Chunk start	93.62	90.65	92.11
Start BLS	88.30	72.95	79.90
Chunk end	91.61	90.12	90.86
End BLS	73.34	86.88	79.54

Table 5.6: Subtasks performance for (NP, VP, PP) definition

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Chunk start	92.48	90.41	91.43
Start BLS	83.02	72.69	77.52
Chunk end	91.22	90.29	90.76
End BLS	77.29	85.30	81.10

Table 5.7: Subtasks performance for $(NP, VP, PP, ADJP, ADVP)$ definition

PUC-Rio - Certificação Digital Nº 0921334/CA

The $F_{\beta=1}$ scores obtained by our direct and subtasks classifier, along with the results for the OpenNLP model, are listed in tables 5.8, 5.9 and 5.10. Once more, each table reports results for a distinct chunk definition. The results for the IOB2 baseline system, used as the basis for the direct classifier, are also shown in these tables. This BLS is detailed in appendix B as well.

The results demonstrate that the subtasks classifier has a consistently better performance than the direct one. We can also verify that the subtasks classifier focus on precision, while the direct classifier presents a slightly better recall for all tested chunk definitions. Furthermore, both ETL-based extractors outperform the OpenNLP model by a good margin, which is a good indication of its effectiveness.

Chunking based on the (NP, VP) definition is clearly harder, judging from the results presented by all tested models. This seems to indicate that, in this case, the derivation heuristic creates chunks that are too general, and whose characteristics are more difficult to identify and predict. This hypothesis is reinforced by the fact that the classifiers yield better results when the complexity of the definition increases.

It is hard to directly compare the results obtained for Portuguese using our heuristic with the ones got for the CoNLL-2000 English corpus. Nevertheless, we believe that the presented performance is consistent with

PUC-Rio - Certificação Digital Nº 0921334/CA

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Subtasks	84.50	77.99	81.11
Direct	81.18	80.25	80.71
OpenNLP	80.44	79.32	79.88
BLS	54.14	54.79	54.47

Table 5.8: Extraction results for (NP, VP) definition

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Subtasks	89.61	85.41	87.46
Direct	87.15	86.30	86.72
OpenNLP	86.19	85.66	85.92
BLS	72.66	75.53	74.07

Table 5.9: Extraction results for (NP, VP, PP) definition

Classifier	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
Subtasks	89.48	86.47	87.95
Direct	87.31	87.09	87.20
OpenNLP	86.83	86.67	86.75
BLS	70.04	74.82	72.35

Table 5.10: Extraction results for $(NP, VP, PP, ADJP, ADVP)$ definition

the results for English, since English text chunking is a very well researched problem, whereas chunking for the Portuguese language is still to be thoroughly researched. Additionally, our ETL models are relatively simple compared to some state-of-the-art approaches available for English, and thus we believe that our results can be improved with the use of more complex techniques and through an intensified modeling effort.

Finally, we list the performance metrics of the subtasks classifier for every chunk type in each definition in tables 5.11, 5.12 and 5.13.

Chunk type	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
NP	81.81	73.68	77.53
VP	93.18	93.44	93.31

Table 5.11: Subtasks classifier results by chunk type for (NP, VP) definition

Chunk type	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
NP	86.21	81.06	83.55
VP	93.61	93.76	93.69
PP	93.77	89.04	91.34

Table 5.12: Subtasks classifier results by chunk type for (NP, VP, PP) definition

Chunk type	**Precision (%)**	**Recall (%)**	$F_{\beta=1}$
NP	86.41	83.10	84.72
VP	93.08	93.67	93.37
PP	94.58	90.57	92.53
ADJP	87.15	84.09	85.59
ADVP	86.28	80.75	83.42

Table 5.13: Subtasks classifier results by chunk type for $(NP, VP, PP, ADJP, ADVP)$ definition

This analysis reveals that NP chunks are among the most difficult to classify for all chunk definitions. Since NP chunks are the most numerous in all cases, as table 3.4 shows, the error rate on their classification has a high impact on the final results for this model. A promising way to carry on research on our models, then, is to look for additional methods to capture essential information related to NP chunks using the proposed extraction heuristic, and increment the models with this information.

PUC-Rio - Certificação Digital Nº 0921334/CA