1
Introduction

During the last decades, technological development has greatly increased information availability for both individuals and organizations. Nevertheless, such huge volume of information becomes an issue when the user is searching for a specific content. It soon became clear that computational methods for retrieving the desired information were needed. Moreover, approximately 85% of the available content is in a *unstructured format* [1] - what includes video, audio, but mainly text. Hence, *Natural Language Processing* (NLP) became a growing field in this area.

Since the aim of a linguistic science is to be able to characterize and explain the multitude of linguistic observations circling around us [2], NLP helps us to understand how information is conveyed in natural language and how one can recognize and extract it by computational means. Because this is a vast objective, most of NLP research focus on different tasks that help understand language structure. It starts with simpler ones like *Tokenization*, *Lemmatization* and *Part-of-Speech tagging*, advancing to more difficult and complex ones as *identification of Semantic Roles* and *Hedge detection*.

Since the 80s, though, but more intensively on this century, Machine Learning (ML) algorithms have been applied to different NLP tasks. These algorithms have some powerful properties that make them especially fit for NLP problems. The ones that use supervised learning find widespread use in several NLP tasks. They are strongly language independent, which makes their findings in one language fairly usable in other languages. Also, they have a low dependency on domain knowledge, enabling one to achieve results in tasks that were only possible to human experts.

However, ML algorithms are dependent on great amounts of data, most of which need to be manually annotated. A great effort has been put throughout the years by the NLP community to build annotated text data sets (*corpus* in the singular, *corpora* in the plural). Accordingly, excellent results were obtained on a great variety of tasks, such as *Part-of-Speech Tagging* [3], *Chunking* [4], *Clause Identification* (or *clausing*) [5], *Named Entity Recognition* [6, 7] and *Semantic*

Role Labeling [8].

A common practice to achieve better results in a task is to use information from some previous tasks as input feature to the ML algorithms being used. For instance, information as tokenization, part-of-speech tagging, chunking, clausing and full syntactic parsing is used as input to solve Semantic Role Labeling (SRL) [9]. Thus, one of the most important tasks in NLP is syntactic analysis or *parsing*, where the structure of a sentence is inferred according to a given grammar. As a result, the syntactic analysis tells us how to determine the meaning of the sentence from the meaning of the words in it.

Although most of the work in syntactic parsing has been done based on *Phrase Structure grammars*, in recent years the NLP community has drawn its attention to syntactic parsing based on *Dependency grammars*. Phrase Structure grammars break natural language sentences down into its constituent parts, namely phrasal categories and lexical categories, thus generating a *constituent tree*. Phrasal categories comprise noun phrases, verb phrases and prepositional phrases. Lexical categories, or parts-of-speech, include noun, verb, adjective, adverb, and preposition. On the other hand, in Dependency grammars, each word has a *direct head-dependent* relation to another word it depends on, thus generating a *dependency graph*, or sometimes a dependency tree. Additionally, each of these dependency relations is classified according to its type. For instance, a *modifier* relation can describe the relation between an adjective and a noun. Also, an *argument* relation can describe the relation between a noun and a verb.

Dependency syntactic parsing (or just *dependency parsing)* is known to be useful in many applications, such as Question Answering, Machine Translation, Information Extraction, Natural Language Generation, but particularly SRL greatly benefits from dependency parsing as shown in [10]. Accordingly, dependency parsing has been proposed as the Conference on Natural Language Learning (CoNLL) Shared Task, both in 2006 and 2007, and part of the Shared Task in 2008 and 2009.

There are two main Machine Learning paradigms for dependency parsing modeling: *transition-based* parsers and *graph-based* parsers, both using supervised learning. Transition-based parsers build dependency trees by performing a sequence of actions, or transitions. These transitions represent either an iteration step over the sequence of tokens or the creation of a dependency relation between two tokens. In this case, the trained model has to correctly predict the next parsing transition of a given sentence.

Instead of locally treating dependency relations, graph-based parsers learn models that treat the sentence as a whole. Given a sentence, they assign a score to every possible dependency relation. Next, the parser uses this information to

find the best dependency tree for the sentence. The learning step is then used to find good scoring functions and estimate its parameters. These two different approaches also lead to different types of errors, as shown in [11].

We present a *token classification* approach for dependency parsing, providing a simpler modeling to this problem. Preliminary results for this approach are reported in [12]. Here, the trained model assigns a class to each token that uniquely identifies its head, thus allowing the use of any Machine Learning classification algorithm. Using this approach we apply and evaluate the Entropy Guided Transformation Learning (ETL) algorithm to three languages: Danish, Dutch and Portuguese. As far as we know, this is the first study that effectively solves the dependency parsing task using a token classification approach, achieving results competitive with the state-of-the-art, showing that this approach is a promising one.

This dissertation is structured as follows. In Chapter 2, we further describe the Dependency Grammar, comment on issues on dependency parsing as dealing with non-projectivity, as well as its applications and the approaches used to solve this problem. In Chapter 3, we present our Token Classification approach, how the classes are created and how this approach allows the dependency parsing to be broken down onto subtasks and the creation of a baseline classifier. Chapter 4 describes the Machine Learning algorithms used in this work, while Chapter 5 describes the languages addressed and *corpora* used, as well as the results achieved in each one. In Chapter 6, we present our conclusions and the future works that build upon the presented results.