Iúri Chaer

A study on the Theory of Prediction applied to the semantic analysis of Natural Languages

Dissertation presented to the Escola Politécnica da Universidade de São Paulo for the Title of Master in Engenharia Elétrica.

São Paulo
2010

Iúri Chaer

A study on the Theory of Prediction applied to the semantic analysis of Natural Languages

Dissertation presented to the Escola Politécnica da Universidade de São Paulo for the Title of Master in Engenharia Elétrica.

Área de concentração:
Sistemas Digitais

Orientador:
PhD. Ricardo Luis de Azevedo da Rocha

São Paulo
2010

To my wife, Aline

Acknowledgments

To my wife, Aline. Her work as revisor was essential and her insistency – delicate but constant – against me procrastinating is one of the main reasons why this project has been taken to a good end.

To my friend and orientator, the Professor Doctor Ricardo Luis de Azevedo da Rocha, always present and ready to point me the right direction.

To my mother, who has always supported me and actively helped me revising and suggesting changes, from the initial phases of this work. Her perseverance is a model that has never stopped motivating me.

To my father who, without even noticing, taught me to love the scientific method and to choose the path that utimately brought me to this place. His biography should be in encyclopaedias illustrating the "pragmatism" entry.

To my friends, that have supported me almost as much as they have interrupted me. Sometimes we just need a break.

Finally, to the Brazilian National Scientific Development Council (the *Conselho Nacional de Desenvolvimento Científico e Tecnológico*, or CNPq) for the resources provided and to the Reuters news agency that, through the National Institute of Standards and Technology (NIST) of the United States, provided me with the material needed for some of this work's most important tests.

Abstract

In this work, computer learning is studied as a problem of induction. Starting with the proposal of an architecture for a system of semantic analisys of Natural Languages, the two modules necessary for its construction were built and tested independently: a pre-processor, capable of mapping the contents of texts to a representation in which the semantics of each symbol is explicit, and an inductor module, capable of formulating theories to explain chains of events.

The component responsible for the induction of theories implements a restricted version of the Solomonoff Predictor, capable of producing hypotheses pertaining to the set of Regular Languages. Such device presents elevated computational complexity and very high processing time even for very simple inputs. Nonetheless, this work presents new and interesting results showing its functional performance.

The pre-processing module of the proposed system consists of an implementation of Latent Semantic Analisys, a method which draws from statistical correlation to build a representation capable of approximating semantical relations made by human beings. It was used to index the more than 470 thousand texts contained in the first disk of the Reuters RCV1 *corpus*, resulting, through dozens of parameter variations, 71.5*GB* of data that were used for various statistical analises. The test results are convincing that the use of that pre-processing module leads to considerable gains in the system proposed.

The integration of the two components built into a full-fledged semantical analyser of Natural Languages presents itself, at this moment, unachievable due to the processing time required by the inductor module, and remains as a task for future work. Still, Solomonoff's Theory of Prediction shows itself adequate for the treatment of semantical analysis of Natural Languages, provided new ways of palliating its processing time are devised.

Keywords: Computer learning; Artificial intelligence; Natural language; Formal semantics

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

In 1989, Richard Saul Wurman, an award winning American architect, gained notoriety for his book *Information Anxiety*, which noted that a single copy of the *The New York Times* newspaper contained more information than the average person would be able to accumulate throughout her whole life 300 years before (WURMAN, 1989). Eight years later, in 1997, a report published by the *Reuters* Magazine entitled "Information overload causes stress" used statistical data collected by the Humboldt University, USA, to demonstrate that the amount of written information available in the world doubled every 5 years (INFORMATION..., 1997). Such growth is being intensified and its effects have become evident since the end of the twentieth century with the popularization of telecommunication systems. An interesting indicator of the phenomenon is in Wurman's decision to publish a new volume of his book in 2001, which states, in (WURMAN, 2001, p. 8):

ecause we have greater access to information, many of us have become more involved in researching and making our own decisions, rather than relying on experts. [...] The opportunity is that there is so much information, the catastrophe is that 99 percent of it isn't meaningful or understandable. [...] We need to rethink how we present information because the information appetites of people are much more refined. [...] Everyone needs a personal measure to distinguish useful information from raw data.

In fact, nowadays, the volume of information seems to have reached a point far beyond human capacity for management. Presented with a performance problem associated with scale, it is expected that a computer science student start their investigation by the method that underlies the system. That because there are usually fundamental approach or method changes that might yield an improvement on the system's efficiency. However, such approach has not been the most widely used. Nowadays, we see a great effort on the study of systems for information retrieval, machine translation of texts and other such devices, which are apparently dedicated to palliatively cope with the symptoms of a fundamental problem: the incompatibility between the way in which information is stored and the way it gets used.

The word *logos* comes from the Ancient Greek (CRAIG, 1998, p.818):

Logos became an important term in almost all philosophical schools. It emerged about 700 BC as the accepted term for discourse at any length, though seldom if ever naming a single word [...] Plato uses the term logos *in* almost all senses [...] moreover for intellect, thought or intelligence [...] contrasted with sense-experience which Plato associates with instability and untruth [...] Aristotle defines *logos* as a composite significant utterance, but actually uses it in a wide variety of senses [...] In his metaphysics *logos* indicates and sometimes equates with the substance, form or essence of things...

Although it has acquired various meanings over time, synthetically *logos* may be understood as referring to thought, speech, reason or meaning. For centuries scholars have presented controversial theories about whether thought precedes language or vice versa. Philosophers like RenÃ© Descartes and Wilfrid Stalker Sellars and educators such as Jean Piaget and Lev Semenovich Vytgotsky hold divergent views on the subject (ASHER, 2005). Here, we are interested primarily in setting one of the base assumptions of this work: when considering language processing by machines, it is essential to understand thought and language as strongly bound factors.

1.1 Purpose

The purpose of this research is to investigate the feasibility and the characteristics of a system of semantic analysis of Natural Languages based on Solomonoff's Theory of Prediction. To that end, the relevant modules of this system were developed and tested so that their behavior could be observed.

Concerning the operationalization of the proposal, there are a few important factors to observe. On the one hand, you need an inductive device capable of making useful theories about a universe as a whole from the observation of parts of it, ie: a mechanism which, from a finite number of samples, can come up with generalizations, analyses and syntheses that can respond satisfactorily well to queries about the whole set of data. On the other, no system can be useful if it can't respond within a reasonable timeframe. Given the complexity of Natural Language, that means one needs mechanisms to convert texts to a more treatable form without significant loss of content. To deal with these requirements, the proposal is to build a semantic analyzer for Natural Language restricted in scope and scale, but that within its domain, is able to translate information into a representation of knowledge. It is known beforehand that the computational complexity of the mechanism proposed by Solomonoff is exceedingly high to build a full application at this time, so that the main objective of this study is to investigate the behavior and properties of the modules of a system of semantic analy-

sis of Natural Language as proposed. The result helps to establish the feasibility, the qualities and defects of the system as a whole.

The specific objectives, important motivating factors for the research involved, is the investigation of human cognition, mechanisms of induction, models of linguistics and computational formalisms appropriate to the requirements of adaptivity and inductive capacity imposed by the problem. Specifically, we intend to research and develop a device which induces hypotheses based on the observation of a limited number of events, to research and develop a system capable of transposing texts in Natural Language into a representation more appropriate to automatic computation, and to experiment with these devices. Thus the goal of finding ways to integrate the contributions of these different lines of research is made achievable.

1.2 Methodology

The conventional approach to semantic analysis of Natural Languages is restricted to the establishment of relations between terms, both in computing (eg (MCGUINNESS; HARMELEN et al., 2004)) and linguistics (an example is the Formal Semantics proposed by (MONTAGUE, 1973)). Such treatment can be done relatively simply, but presents serious limitations (its characteristics are discussed in chapter 3). Solomonoff's proposal for the formulation of theories for event sequences (SOLOMONOFF, 1964) contains, in its construction, a different way of approaching the problem. The process of condensation of observations in algorithms loses much in simplicity, but the description of the resulting knowledge can solve the problems discussed earlier in this chapter, particularly the incompatibility between description and use of knowledge.

The final solution of the predictor device proposed by Solomonoff (SOLOMONOFF, 1989) is incomputable no matter its input. Even applying clever strategies to obtain suboptimal solutions, the computational complexity of the system is exponential. Bearing in mind that, in Natural Languages, the relation between signifier and signified is arbitrary – as advocated by linguist Saussure (SAUSSURE, 1983 apud ASHER, 2005, Vol X P. 758), one of the fathers of modern linguistics – the chances of obtaining significant results in feasible time are not promising. To improve the chances that our implementation of the device is usable over such complex chains of events as snippets of text in Natural Language, it is proposed to preprocess the input data using methods that expose semantic relationships between terms.

Latent Semantic Analysis is a statistical technique that uses linear algebra to reveal indirect relationships between words in texts in Natural Language. It was proposed in

(DEERWESTER et al., 1990) as a method for processing documents for information retrieval. Its low computational complexity coupled with the ability to approximate associations made by humans (LANDAUER; DUMAIS, 1997; PAPADIMITRIOU et al., 2000) and convenient representation of its results makes the method an excellent candidate. Further discussion on the reasons for this choice can be found in chapter 3.

From the discussion above, we propose an outline for the construction of the semantic parser for Natural Language: a system composed of a preprocessing module, which makes input texts more tractable by applying to them the method of Latent Semantic Analysis, and a module capable of inducing theories about the content that generated the description being processed, based on Solomonoff's Theory of Prediction. There are indications in other studies on how to implement the Theory in its complete form (ROCHA, 2000), but in the interest of restricting the system's computational complexity as a whole and of developing it within a reasonable time frame, it was decided to develop a limited version of Solomonoff's proposal, restricted to the set of Regular Languages.

1.2.1 Linguistic Background

When discussing the analysis of the semantics inherent in a section of text in Natural Language, one must keep in mind the difficulty that human beings, creators and users of this class of languages, meet even when trying to formally define the meaning of isolated words. Attempts to approach concepts by analogy, such as those found in dictionaries and encyclopedias, very seldom – if ever – reach a root meaning. A pigeonhole is a type of house, house is a form of construction, construction is an organization of elements – in the end, there is always something left to be defined.

The English philosopher John Locke wrote, in the seventeenth century, his *An Essay Concerning Human Understanding*, where he considers (LOCKE, 1973, p. 258):

ords having naturally no signification, the idea which each stands for must be learned and retained, by those who would exchange thoughts, and hold intelligible discourse with others, in any language. But this is the hardest to be done where:

First, The ideas they stand for are very complex, and made up of a great number of ideas put together.

Secondly, Where the ideas they stand for have no certain connection in nature; and so no settled standard anywhere in nature existing, to rectify and adjust them by.

Thirdly, When the signification of the word is referred to a standard, which standard is not easy to be known.

Fourthly, Where the signification of the word and the real essence of the thing are not exactly the same.

The apparent lack of semantic axioms in human knowledge is the first problem that one faces in analyzing the meaning of words and in the interpretation of statements within texts in Natural Language. A system capable of performing that task must establish, first, the relationships between words and their meanings – as defined by Ferdinand de Saussure (ASHER, 2005; FALK, 2003), the relationship between signifier and signified.

The second obstacle to overcome in this project is that of examining the structure of texts in Natural Language. Clearly, the phrases "John kicked the stone" and "the stone kicked John" have radically different interpretations, and the string of words "stone John the kicked" doesn't even admit an interpretation. Thus, even if there was a system able to associate each existing word to a meaning, the interpretation of sentences would remain a challenge.

The theory of syntax that is found in traditional grammar books lacks formality. It fails to explain much of the understanding of grammatical constructions by people and, thus, can't be directly used for the solution of the issues met in this proposal (CHOMSKY, 1986). However, some of the concepts of language advocated by researchers working in that area are extremely relevant to this project.

Noam Chomsky, Professor of Linguistic Theory, Syntax, Semantics and Philosophy of Language at the Massachusetts Institute of Technology (MIT), has proposed a new approach to the grammar of Natural Languages: the Generative Grammar (CHOMSKY, 1965). His intention is to describe completely, formally and universally the rules of construction of valid sentences in Natural Languages. It is an ongoing project, with a good deal of controversy on fundamental theoretical points, so that, considering its current state of development, it was deemed imprudent to base this work's concrete implementation in the Generative Grammar. The choice not to apply it, however, does not exempt us from the examination of the work of the linguist – especially since this work shares in its root the base hypothesis that led Chomsky to idealize Generative Grammar.

The initial hypothesis assumed by Chomsky on the Generative Grammar proposal is the existence of a system of rules shared by all human beings, a natural system to all individuals of the species, which enables their communication. His main argument is based on the speed of learning and competence that children, exposed to the most diverse environments, show in learning and using Natural Languages – the so-called "Paradox of the Poverty of Stimulus" or "Plato's Problem". If it is true that the gen-

erating mechanism of all Natural Languages is the same and that the understanding of its rules is innate in humans, it is not necessary to expose an individual to more than a small fraction of the possibilities of a language for him to be able to understand it in its entirety. (HAUSER; CHOMSKY; FITCH, 2002).

In another one of his works, the scholar says (CHOMSKY, 1986, p.27, 33–35):

here does exist what we have called an internalized language and that it is a problem of the natural sciences to discover it. [...] In the sciences, at least, disciplines are regarded as conveniences [...] and their boundaries shift or disappear as knowledge and understanding advance. In this respect, the study of language [...] is like chemistry, biology, solar physics, or the theory of human vision.[...] Linguistics becomes part of psychology...

In this project, syntactic parsing of a language is seen as a problem of inducing rules that produce valid constructs for that language. In choosing the method for establishing these rules, it is essential to keep "Plato's Problem" in mind. It is absolutely necessary to define a set of rules that is finite and simple compared to the infinitude of constructions allowed by Natural Languages. As for the semantic analysis of complete sections of text, excerpts were chosen with concepts and ideas in basic constructions produced by the organization of words. Thus, *a priori*, the methodological proposal of this research is to give semantics a treatment similar to that given to syntax.

1.3 Structure of the thesis

The work reported in this dissertation consisted of an initial study on Machine Learning and Natural Language processing, of the development of tools for evaluating the proposals made and, finally, of tests and the exploration of the results obtained. The structure of the text reflects such evolution.

The first part of this thesis contains a brief compilation of the concepts involved in developing the project – a brief review of the theoretical foundations for the choices made in the proposed research. In it, key issues addressed in the study are explored, namely: machine learning, systems for knowledge representation and the study of Natural Languages. The first two chapters are more focused on the issue of computational systems needed to implement a system capable of analyzing the semantics of texts in Natural Language, while the third contains justification for the chosen approach from the linguistic and cognitive standpoints.

The second part of this work deals with experiments done with computer programs

developed to verify the behavior of the algorithms we propose to use to implement a system capable of semantic analysis of Natural Languages. Firstly we describe our implementation of a simplified version of the Solomonoff's Predictor – the kernel of this work's proposal – along with the tests applied to it and the results obtained. In the following part we discuss our implementation of the Latent Semantic Analysis system proposed in (DEERWESTER et al., 1990), the environment used to verify its performance and its tests results. That second system module is actually the pre-processing module described in section 1.2.

The third part of the dissertation finishes this work with a few final remarks. It presents our conclusions about the functioning of the Solomonoff's Predictor and of the Latent Semantic Analyzer developed for this work, considerations on their integration and also the most promising avenues that are envisioned for future developments of this research.

Part I

Concepts

2 Machine Learning

The term “Machine Learning” is used to denote the operation of computational systems capable of changing their behavior in response to external stimuli or experiences accumulated in the past (ALPAYDIN, 2004). The presentation of such concept raises a pertinent question, which would be the justification for the creation of such a system instead of simply solving the problem (NILSSON, 1996) (choosing to build, for instance, a program which learns to convert speech into writing from examples instead of another one that simply performs such conversion using predefined rules).

There are several situations where the learning behavior might be desirable. The subject which is the focus of this work is an instance where the application of Machine Learning techniques is very justifiable, as the problem of semantic analysis of Natural Languages is extremely difficult to describe – not even human beings, creators and users of hundreds of Natural Languages, are able to lay out in formal rules how that activity can be performed. Natural Languages can produce an infinite number of sentences and texts, each possibly containing an infinite quantity of ideas and concepts. The rules governing its composition, if they exist, are not known. All that makes it impossible to build a complete system for the processing of Natural Language composed exclusively of predefined rules.

The presentation on Knowledge Representation within chapter 3 addresses and provides part of the rationale for the choices for machine learning methods made in this work, but there are reasons other than adequacy of representation that must still be clarified. The first relevant point is the division of tasks set in chapter 4: finding meanings for words should be treated as a separate task to be fulfilled prior to the analysis of the semantics of sentences and texts. Given this project’s focus on text processing, the importance of the module responsible for the analysis of words lies mainly on the adequacy of its results to the module responsible for compound excerpts. The requirements of this latter module restrict the alternatives available to the former so that its choice must be prioritized.

One important attribute differentiating the methods of machine learning is the re-

quired degree of interference by humans. In the semantic analysis of Natural Language, it is very desirable that it be minimized. The semantic analysis of texts is among those human activities considered subjective and it is expected that the characteristics of the operator of the system skew the results. That not only impairs its validation, but also compromises its usefulness as a general purpose device, once training is started. Bearing those requirements in mind, four preeminent machine learning methods were chosen for further analysis: Genetic Algorithms, Neural Networks, the branch of probabilistic methods based on Bayes Theorem and that of statistical methods.

Genetic Algorithms are a machine-learning method that attempts to emulate biological evolutionary theory (MITCHELL, 1996). Several instances of a prototype solution to the problem to be solved are built, each differing in certain predetermined characteristics. All those instances are put to work on the problem, and the n most successful ones, according to a predefined fitness function, are chosen. Those chosen instances are then combined in different ways and suffer new random changes (in a way analogous to *mutations*), resulting in a new generation of instances that is to go through the same cycle as the previous one. If the initial prototype of the solution, the parameters chosen for the mutation mechanism, the algorithm of recombination and, most importantly, the fitness function, compose an adequate system, it converges to a solution. However, when it comes to semantic analysis of Natural Languages, such requirements present a difficult hurdle to overcome. The characteristics of the problem aren't sufficiently well known to propose even a prototype for its solution. Monitoring the results through a fitness function would also require that there was some kind of measure of what is right as a formal description of the knowledge contained in a text passage – a measure that is also not available. For all those reasons, Genetic Algorithms can't be considered an adequate choice for the problem being studied.

Neural Networks, also known as Multilayer Perceptron Networks (RUSSELL; NORVIG, 2003; ALPAYDIN, 2004), are systems inspired by the functioning of the brain. They are composed of several extremely simple elements connected in layers, each element usually able only to transmit or block signals that reach it. Connections between those elements are directed and have weights that determine how much of the signal emitted by one of them reaches those to which it is connected. Connection weights vary according to the outcome of previous communications between the elements involved. The result is that, given an appropriate network, after a period of training – which consists of exposure to a sample of input signals – the system begins to recognize certain patterns. However, the way they are currently constructed, neural networks can't be retrained. This is a very serious obstacle to its implementation in this project since, as explained earlier in this work, semantic analysis requires a pro-

cess of continuous learning. The sets of language constructs and of ideas are infinite. It is necessary that the system is able to modify itself and to continuously contextualize new information. Thus, Neural Networks also do not comprise a suitable alternative for this work.

There are various methods of Machine Learning based on the equation of conditional probabilities of Bayes' Theorem. Most of the events that one might need to describe and of the problems one might need to solve follow patterns, and the way in which the Theorem gradually restricts the universe of possibilities is extremely attractive. It is used in simple devices for natural language processing, such as *N-Grams* (JURAFSKY; MARTIN, 2008), but is also basis for much more complex methods such as Hidden Markov Models (RUSSELL; NORVIG, 2003). A particularly relevant theory based on such probabilistic apparatus is Ray Solomonoff's Theory Prediction. Using elements from Information Theory, Solomonoff proposed a theory and a learning method (SOLOMONOFF, 1964) which are very suitable for the semantic analysis module of this project. In addition to being based in a convincing explanation for the learning process, the method proposed by Solomonoff has guaranteed convergence for generic cases in an impressively small number of steps (section 2.1) and is structured around a continuous process. Because it is a generic method, however, its computational complexity is very high. Except for reports by the author, who actively researched that device from 1964 to 2009, and for very simple didactic implementations, no work demonstrating the device's application and results could be found as reference. If that is not a positive sign for its adoption in this work, one can not say it is negative. The problem of semantic analysis of texts in Natural Language is widely studied but unresolved. There is little chance of significant achievements in the area by simply applying the traditionally used methods. Thus, we opt to use that very well grounded but seldom applied theory, in the belief that any results obtained will be relevant.

Having chosen a method for semantic analysis, there still remains the choice of how to perform the processing of words. Solomonoff's Theory of Prediction works checking hypotheses for their ability of describing sequences of events, in an attempt to providing good extrapolations for future events. Thus, it is important to prioritize the simplicity of the descriptions of events, so that the attributes involved in relations of cause and effect are always exposed to the mechanism. Chapter chapter 3 gives a brief summary on vectorial methods for knowledge representation. Such form of representation is adequate to our needs, and becomes particularly interesting when observing results from other researches on techniques that produce representations of this type conducted in the last decades.

There is one statistical method for Machine Learning, published in 1990, which

stands out among its peers: Latent Semantic Analysis (better known by the acronym LSA) (DEERWESTER et al., 1990). Based on the reduction of a vector space containing a sample of events, the method constructs an orthogonal basis of automatically extracted implicit attributes. The outcome of its application to a *corpus* has no direct meaning to a human being, but by choosing an appropriate number of dimensions for such resulting vector space, the associations that can be extracted from the system are impressive (LANDAUER; DUMAIS, 1997). One mustn't forget that it is a purely mathematical method, and that the number of dimensions suitable for the resulting vector space, for instance, has to be empirically determined, through experimentation. LSA has been adopted in this work for the processing of words for its output's suitability for the next module and because of the extremely consistent results reported in other works. It is important to note that there are recent proposals for improvements to the original model, with the application of more complex statistical devices (HOFMANN, 1999; BLEI; NG; JORDAN, 2003). These proposals, while being shown to improve the practical performance of the method, do not radically alter its operation. As such, we opted in this work to apply the original LSA proposal, as it has been more extensively tested and documented. If the method's performance is later shown to be inadequate, it can be changed without loss to the system's global architecture.

2.1 The Theory of Prediction

In 1964, Ray Solomonoff proposed a device to systematize the inductive process (SOLOMONOFF, 1964). Such device, the Prediction Theory or Solomonoff's Theory of Induction, is based on the Bayes Theorem, interpreting it as a method for testing hypotheses:

$$P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)} \tag{2.1}$$

Meaning: given the occurrence of an event, the probability of a hypothesis being true is given by the likelihood that it attributes to such event, multiplied by the probability of the hypothesis and divided by the global probability of the event. Read as such, the Bayes theorem seems to be a powerful tool for ranking hypotheses. Were it not for the difficulty of formulating new hypotheses and establishing the correctness of the *a priori* probabilities $P(H)$, the problem of finding explanations for events would be essentially solved. The key steps given by Solomonoff were, precisely, the formalization of a mechanism to establish $P(H)$ and the indication of the universe of hypotheses to be explored. The mathematician proposed also methods to search these hypotheses, but these are more complex (both in the computational connotation and in

the usual one) and less well defined. Even in humans, the inductive process appears to be essentially heuristic. In a recent work (SOLOMONOFF, 2003a), Solomonoff points in a direction that seems promising to mitigate the problem of the complexity of generating and testing hypotheses: the division of problems in different domains.

The next subsections will explore a bit further the main topics of Solomonoff's Theory of Prediction, in order to show its applicability in the synthesis of knowledge from data and, therefore, in the semantic analysis of texts in Natural Language.

2.1.1 Kolmogorov Complexity

Algorithmic complexity, or Kolmogorov complexity, is a measure of the amount of information contained in a string of symbols. It is intuitive that there is less information in the string 10^{100} than in the vast majority of the 100-digit-long numbers that can be imagined – it is clear from the fact that it has just been represented using only 5 symbols. An obvious proposal for the measure of complexity is the length of the shortest string representing that same information. However, as it is, such simple method does not provide a good indication of the complexity of the information contained in the string, as it is possible to create infinite languages to describe the same concepts, each using strings of different lengths. How might one ensure that such measure is relative only to the information to which it applies, and not dependent on the language used?

The solution for the dilemma comes with Church's Thesis (apud (LI; VITÁNYI, 1997)). Its proposal: the concept of computability is independent of the method used for computation. According to the Thesis, any effectively calculable procedure can be represented by a universal computer, and all universal computers are equivalent because they can emulate the behavior of each other running a constant number of steps. Assuming Church's Thesis is correct, a complexity definition based on the length of a description of a datum starts to make sense. The $K(x)$ complexity of string x, whose length is $l(x)$ when described using formalism f, capable of universal computation, is given simply by:

$$K(x) = min\{l(p)|f(p) = x\} \tag{2.2}$$

The equation above asserts that the complexity of string x, when described through formalism f, is the length of the shortest string that, when applied to f, generates x. As said, it follows from Church's Thesis that, using a different formalism M that also has universal computational power, it is possible to simulate f with a constant number of steps and obtain for the same output string the same complexity, except for

a value bounded by a constant. Such constant can be arbitrarily large, but that does not invalidate the result. More than that, intuitively it makes a lot of sense. Some systems lead to more natural descriptions for certain specific problems than others, although it might be possible to convert directly between them. A problem does not become more complex due to the longer description, one may create as long a description as he wishes for any event, and it is always possible to explain the other description system before working on such problem.

The equation (2.2) is, however, incomputable. The only way to find out whether an algorithm produces a given string x is by executing its steps. It is impossible to know *a priori* if an algorithm even stops given an input. The best one can realistically (ie in a time frame compatible with human expectations) achieve are approximations, which may be obtained by restricting search time or the space of algorithms to be explored.

2.1.2 Algorithmic Probability

The principle of Ockham's Razor or the Law of Parsimony, attributed to William of Ockham, who lived in the High Middle Ages, says (ENCYCLOPÆDIA..., 2010):

> *"Pluralitas non est ponenda sine necessitate."*
> or *"Plurality should not be assumed without necessity."*

That means roughly that simplicity is a decisive factor for the plausibility of a description. More complex explanations should be deprecated in favor of simpler ones. As soon as there is a formal definition for complexity, it is feasible to verify such principle that seems to pervade science, and that has indeed been done.

Consider a binary alphabet $A = \{0,1\}$. If each symbol of the alphabet has the same probability, it is clear that the probability of a string d of length $|d|$ is given by:

$$\mu(d) = 2^{-|d|} \tag{2.3}$$

That is, the probability of the string *1* is 2^{-1}, that of *11* is 2^{-2}, and so forth. The algorithmic probability, defined for algorithm A, is the probability, given an alphabet, that a device which outputs symbols randomly generates a string representing A. It may be understood as the probability for the spontaneous generation of an algorithm in a chaotic environment. It follows from equation (2.3) that the probability distribution defined by such strings gives more weight to shorter descriptions. When applied to all algorithms that generate string x, equation (2.3) is dominated by the term that defines the Kolmogorov complexity of x. Because of its characteristics, Solomonoff proposes

that the μ semimeasure[1] be used to estimate the *a priori* probability of hypotheses, so that:

$$P(H) = 2^{-|H|} \tag{2.4}$$

It is known, however, that different algorithms may have intersecting sets of results. Not only that, but each single algorithm can be represented by an infinite number of strings, in the same way that the decimal number *8* can be represented in binary as *100*, or *0100*, or *00100*... Given the prefix-free language D formed by all d_i descriptions for the string x (ie algorithms that can generate such string), it can be said that the probabilities defined by equation (2.3) represent a semimeasure for these descriptions for x. Such semimeasure can be normalized to obtain instead a probability[2]: it represents the probability of a description given an output string, the probability that an algorithm is responsible for the output observed. The restriction that D be free prefix exists to eliminate trivially redundant descriptions (eg, reading from the least to most significant digit, all representations given in the example above for number *8* are trivially redundant), which add no information to the set.

2.1.3 Prediction

Solomonoff proposed the application of the concept of algorithmic probability to the prediction of future productions by an unknown generator (SOLOMONOFF, 1964). The basic issue is to create a device capable of answering the following question: given an initial string x, what is the probability that the next symbol be an a? The classical theory of probability tells us that the probability of the occurrence of string xy, given the occurrence of x, obeys the following equation:

$$P(xy|x) = \frac{P(xy)}{P(x)} \tag{2.5}$$

To predict the probability that string x develops into xy, one can profit from the knowledge from the generator's behavior up to x, using the probabilities obtained from the normalization of the semimeasure described in equation (2.3). The function of probabilities P'_M obtained by using those semimeasures follows P_M, that generates the

[1] Semimeasure is a relation associating a real number $\mu \in [0,1]$ to each element of a set, such that $\sum \mu \leq 1$.

[2] Probability is simply a semimeasure in which the total sum equals one.

string with a quadratic error defined by (WILLIS, 1970; SOLOMONOFF, 1978):

$$\sum_{m=1}^{n} (P_M(x_{m+1} = \sigma | x_1 \ldots x_m) - P'_M(x_{m+1} = \sigma | x_1 \ldots x_m))^2 \leq 0.5 \cdot \ln D'_M \tag{2.6}$$

Where:

$\mathbf{m}$ is the index of the symbol being analyzed;

$\mathbf{P_M}(\mathbf{x_{m+1}} = \sigma | \mathbf{x_1} \ldots \mathbf{x_m})$ is the probability that the data generator produces symbol σ following the $x_1 \ldots x_m$ string; and

$\ln \mathbf{D'_M}$ is a constant factor dependent on the device which generates the observed events and is approximately equal to $K \cdot \ln 2$, K being the Kolmogorov complexity of the device.

Since such quadratic error is not dependent on the length of the string, by induction it becomes clear that it decays faster than the inverse of the length of the string. It is a high rate of convergence that is true even for relatively complex devices predictors (those with a greater D'_M), but one that still favors their simpler counterparts.

Solomonoff extended his model to deal with unordered sets of data (SOLOMONOFF, 1999). Later, in 2003, he described a way to use it for an even more generic issue: the mapping of a set of strings to another, in a way that can be seen as locating the best answer to a question (SOLOMONOFF, 2003b).

The application initially proposed for the Theory of Prediction lends itself to the simplest forms of analysis on texts in Natural Language. It might be used, for instance, to create an automated text verifier capable of analyzing the consistency of writing style. The model extended for unordered data sets might be able to synthesize knowledge from texts, creating a compressed description of the information contained therein – enough for extrapolation of implicit information. The third application, if implemented, would provide something truly interesting: a generic expert system able to learn in a way similar to that observed in human beings.

2.1.4 Search for hypotheses

The previous subsection describes a method for validation and refinement of hypotheses to describe a series of events, but it doesn't mention the source of hypotheses. In order to be able to complete at least an approximation of that procedure, there must be a way to find potentially valid hypotheses in a restricted space of functions. In

(SOLOMONOFF, 2003a), it is proposed that such domain be that of partially recursive functions and, in (SOLOMONOFF, 2005), a few candidate methods for the search of hypotheses are enumerated:

a. Lsearch, a universal search method proposed by Levin and extended with the contribution of Solomonoff.

b. PPM (Prediction by partial matching), a statistical technique used for data compression which, quoting Solomonoff, is "a very fast, surprisingly precise method for approximate induction [...] but at the moment there seems to be a very large speed penalty."(SOLOMONOFF, 2005, p. 2).

c. Discovery of stochastic context-free grammars.

d. Genetic programming.

Whichever method is chosen, it is certain that its execution will be responsible for most of the processing time of a device based on the Theory of Prediction.

3 Knowledge representation

The volume of books and articles devoted to finding ways to represent knowledge in the discipline of Artificial Intelligence (MYLOPOULOS, 1981; BRACHMAN; FIKES; LEVESQUE, 1983; BRACHMAN; LEVESQUE, 1985; IWANSKA; SHAPIRO, 2000; LENAT et al., 1990) is a significant indicator of the importance and complexity of the subject. Collection and processing of information on any subject require a representation of the elements pertinent to it.

Models are representations of something one wishes to analyze. What sets apart one representation from another it its adequacy to a certain use. That's the reason its study becomes not only relevant but also important. Different representations privilege different activities, so that the adequacy of the representation to the intentions of the user is often a factor that will determine the reaching of an elegant solution or a resounding failure (DAVIS; SHROBE; SZOLOVITS, 1993).

Much of the current studies on the interpretation and description of knowledge in the field of computing is based on relating different terms. "Cat" is defined as a "mammal," "mammal" is a "vertebrate", "vertebrate" is a "chordate possessing a spinal column" and so forth. Such approach, called a Semantic Network (MYLOPOULOS, 1981), has its merit, as knowledge is extremely dependent on relationships between concepts. Because of its relative ease of implementation and because its organization is perfectly suited to the distribution of information in independent nuclei, this type of representation has gained popularity over the last decade and is being used in proposals such as the Semantic Web (MCGUINNESS; HARMELEN et al., 2004; HENDLER; BERNERS-LEE, 2010). However, even though one such system might lead to very positive results in the analysis of the semantics of a text on, for instance, biology, they can't be said to possess more knowledge about the subject than a typewriter does about grammar. The presence of key semantic elements is not sufficient to characterize knowledge. The inferences that can be extracted from a representation of this type are limited to Propositional Logic, which prevents many forms of extrapolation which are completely ordinary among human beings.

A form of representation with many similarities to Semantic Networks has been very popular on the field of Information Retrieval: the representation in vectorial spaces (MAGNINI; CAVAGLIA, 2000; DEERWESTER et al., 1990; HOFMANN, 1999; BLEI; NG; JORDAN, 2003). Conceptually, it is extremely simple: a vectorial space is created with dimensions analogous to a limited number of conceptual attributes, and the representation of each signifier is made by placing it as vector in that space. Such technique is not traditionally listed as a method for Knowledge Representation. Nevertheless it has had a significant part on the field of inference, mainly due to the results obtained from its application and to the possibility of completely automatizing the building of knowledge bases, providing completely objective results dependent only on the *corpus* used to produce them – a notable advantage over Semantic Networks, which are frequently dependent on the direct interference from specialists.

Another approach, common in automated theorem provers, is the use of assertions in logic form (First Order Logic, Fuzzy Logic or other such form)(FITTING, 1996). That method of knowledge representation gains much in inductive power relative to semantic networks, due to the fact that it allows for better specification of the context for each assertion. For example, it makes possible to create a description for a plane and then make the assertion that any plane without wings falls, without any implication for airplanes with wings or other elements without a wing. In a semantic network, such an attempt would result in undesirable generalizations or the creation of new and overspecified elements (instead of creating only one entity "Plane," one might be forced to specify "WingedPlane" and "WinglessPlane"). An application of the approach with logic assertions that merits special attention was made by the linguist Richard Montague, who used Intentional Logic to represent the semantics of sentences in Natural Language (ASHER, 2005; MONTAGUE, 1973), a work that had great impact in the understanding of semantic analysis of Natural Languages.

A third approach, less popular nowadays, is the representation of knowledge in the form of algorithms. Functional languages like LISP (a family of programming languages, designed by John McCarthy in 1958, which evolved mostly for applications in Artificial Intelligence (RUSSELL; NORVIG, 2003)), that make little distinction between data and procedures, have been widely used for this type of task in expert systems and the like (MYLOPOULOS, 1981). The ease of inferring from algorithmic descriptions and the potential for conciseness of the assertions in this representation certainly exceed those of the other two techniques already exposed in this section. On the other hand, the difficulty of feeding and correcting that type of knowledge base represents an obstacle to its manual maintenance. However, the proposal for an automated inductive system not only provides a way to overcome such difficulties but

also sets, by way of algorithmic representation, a convincing learning theory and an efficient method for inference (SOLOMONOFF, 1964) (more details in section 2.1).

The current project proposes to use algorithms for its internal representation of the knowledge contained in organizations of words because of its advantages and because it is a form of representation adequate to an extremely attractive Machine Learning technique. Assuming, as Saussure (whose theories are briefly presented in chapter 4), that the semantic content of words is arbitrary, it must be acknowledged that obtaining an algorithmic representation of knowledge contained in each of them would probably require human interference and the creation, even if disguised, of a semantic network or other similar taxonomic structure. The proposal to avoid that is to use a vectorial representation for words, obtained from a method for latent semantic extraction – Latent Semantic Analysis (DEERWESTER et al., 1990). Such technique is described under section 3.1.

3.1 Latent Semantic Analysis (LSA)

The automatic inference of semantic associations between words and their contexts is a very intricate problem of Artificial Intelligence. One of the simplest and most successful methods for the indexing of texts for information retrieval (YATES; NETO, 1999) was proposed by Salton in (SALTON; BUCKLEY, 1988). Known as *tf-idf* ("term frequency - inverse document frequency"), the technique consists in keeping score of both global and text-by-text term counts within a *corpus*. That information is then used to establish a vectorial space where each dimension corresponds to a term, and each document is represented as a sum of its components. The modulus of each term is provided by the following equation:

$$|t| = \frac{tf \cdot \log \frac{N}{n}}{\sqrt{\sum W_{t_i}^2}} \tag{3.1}$$

Such that:

t is the term vector;

tf is the frequency of occurrence of the term in the document;

N is the total number of documents in the corpus being examined;

n is the number of documents in the corpus that contain the term, and

$\sqrt{\sum \mathbf{W_{t_i}^2}}$ is a normalization factor which consists in the root of the sum of the squares of the weights of all terms belonging to that document.

Following that same vectorial approach for the representation of texts within *corpora*, Deerwester and a group of researchers proposed in 1990 (DEERWESTER et al., 1990) the Latent Semantic Analysis (LSA) method. The idea behind it is quite simple: reducing the number of dimensions of the vectorial space obtained from models such as *tf-idf* retaining, as much as possible, the original relations between vectors. There is a minimum number of dimensions needed to represent, without loss, the vector space determined by a *corpus*. Dimensional reduction beyond that limit, depending on how it's done, may reduce the sample noise (inappropriate or irrelevant contextual relationships) and elicit strong indirect relations. Results with these characteristics are shown in several studies, in particular (DEERWESTER et al., 1990; LANDAUER; DUMAIS, 1997).

To reduce the number of dimensions in a vector space with the least possible loss of information, the LSA method is based on Singular Value Decomposition (SVD), proposed in 1965 by Golub and Kahan (GOLUB; KAHAN, 1965). Such method decomposes a matrix A in three others:

$$A = U \cdot \Sigma \cdot V^T \tag{3.2}$$

In this decomposition:

U is a unitary orthogonal matrix whose columns are the eigenvectors of $A \cdot A^T$;

V is another orthogonal matrix unit, but its columns are the eigenvectors of $A^T \cdot A$; and

Σ is a diagonal matrix containing the singular values of A – these values are the square roots of the non-negative eigenvalues of $A \cdot A^T$ and $A^T \cdot A$.

Choosing the decomposition in which the values of Σ are in descending order, the result obtained by keeping only the n first rows and columns of Σ (and thus the n first columns of U and n first rows of V) is, for a large number of applications, the best approximation in n dimensions which can be obtained for A. It is that which minimizes the sum of squared errors: $|A - A_n|^2 = \sum_{i,j}(A_{i,j} - C_{i,j})^2$ (DEERWESTER et al., 1990; BERRY; DUMAIS; O'BRIEN, 1995; PAPADIMITRIOU et al., 2000). Such A_n approximation is then given by the following equation:

$$A_n = U_n \cdot \Sigma_n \cdot {V_n}^T \tag{3.3}$$

A common way to compute the distance between vectors in applications of information retrieval, used in the seminal article on LSA, is the cosine of the angle between them (DEERWESTER et al., 1990; LANDAUER; DUMAIS, 1997; YATES; NETO, 1999; WIDDOWS, 2008). Notice, however, that the A_n matrix obtained can not be used for that. After reducing the number of dimensions, it is no longer possible to differentiate between documents and terms. To obtain the vectors corresponding to each document, one must transpose the documents to the new vectorial space. Because of their roles in the reconstruction of A, U represents the array of terms and V the array of documents (BERRY; DUMAIS; O'BRIEN, 1995). The objective in this model is to calculate the distance between documents. Manipulating equation (3.2) to isolate the matrix V, one obtains:

$$V = A^T \cdot U \cdot \Sigma^{-1} \tag{3.4}$$

From equation (3.4) comes the following equation, shown in (BERRY; DUMAIS; O'BRIEN, 1995):

$$d_k = d^T \cdot U_k \cdot \Sigma_k^{-1} \tag{3.5}$$

In equation (3.5), d_k is the representation of the document vector d in the reduced space obtained by truncating the result of the SVD application. The same equation can be used for text search, as queries can be seen simply as short texts expressing a user request. The similarity between a document d and a query q is then given by the cosine of the angle between the vectors representing them. That similarity value can be calculated through a scalar product of vectors, such that:

$$sim(d,k) = \cos(\widehat{dk}) = \frac{d \cdot k}{|d||k|} \tag{3.6}$$

In (LANDAUER; DUMAIS, 1997), results of some experiments directly related to language learning are presented. The LSA method was used to index the *Grolier Academic American Encyclopedia*, then consisting of 4.6 million words organized in 30,473 articles. The *corpus* was mapped in a 300-dimensional space and the result was used to respond to 80 synonymy questions of the TOEFL ("Test of English as a Foreign Language"). Such questions consisted of a model word and four alternatives. The person subject to the test is expected to choose the alternative whose meaning is closest to the model. The result obtained using LSA was a correction rate of 64.4%, compared to an average of 64.5% obtained by humans.

Another relevant study is presented in (PAPADIMITRIOU et al., 2000). In that paper, researchers seek the reasons for the performance of LSA in the association of semantically related terms. Building on some restrictive but quite reasonable assumptions about the normal situation of communication between humans, some theorems that demonstrate the effectiveness of the LSA are proved. Although the initial assumptions are undoubtedly the result of the intuition of the authors, the study gives a good sense of the situations in which one can obtain satisfactory results using LSA and also of those in which the method can go wrong. The study's premise that, intuitively, is the most important, is that documents in the *corpus* should be semantically separable, meaning that each text is relevant to only one topic. The dimensional reduction performed with SVD causes weak contextual relationships to be further weakened and strong relationships, even indirect, to be strengthened. If these contextual relationships don't present well defined semantic meaning, i.e., if each text talking about many, or, at worst, about no particular topic, the relationships between words derived by LSA will not have any semantic value.

It comes as no surprise that it is difficult to understand and, moreover, learn from a text that covers several unrelated subjects. Humans use tools such as punctuation, paragraphing, division into sections and chapters to structure and separate different topics in longer texts. The approach of many works in Natural Language processing, to ignore internal divisions and to consider each text in a *corpus* as an atomic unit, is certainly not the most accurate, but it seems to work quite well for short, oriented texts such as academic papers on (DEERWESTER et al., 1990), encyclopedia articles as those used on (LANDAUER; DUMAIS, 1997) and the definitions from WordNet used by (ESULI; SEBASTIANI, 2005).

From a practical standpoint, for this work, the most important feature of LSA is its ability to map words into a semantic space where the relationships among their meanings are explicit in their coding. It is a method that requires empirical adjustments – the output space's optimum number of dimensions must be determined through testing – and whose quality of results can only be guaranteed in *corpora* composed of coherent and scope-restricted texts. However, the convenience of the representation it produces for texts and the semantic associations that can be extracted from it, remarkably close to those made by humans, leave no doubt about its practical value.

4 The study of Natural Languages

Human communication is matter of studies since antiquity. In his "Letter to Herodotus," the Greek philosopher Epicurus presents a theory about the emergence of Natural Languages (Epicuro, apud FALK, 2003, p.68):

ames ... were not at first deliberately given to things, but men's natures according to their different nationalities had their own peculiar feelings and received their peculiar impressions, and so each in their own way emitted air formed into shape by each of these feelings and impressions, according to the differences made in the different nations by the places of their abode as well.

Such approach to the study of language as a natural science seems to have only been resumed in the Contemporary Age. A famous phrase of St. Isidore of Seville, who lived in the fourth and fifth centuries, defines grammar as "the art of correct expression, the first of Liberal Arts and foundation of them all" and, indeed, from the early Middle Ages up to the nineteenth century, the concept of grammar was essentially linked to textbook rules dictating the correction of constructions (ASHER, 2005). The search for understanding linguistic phenomena, in contradistinction to mere observation and categorization, was only resumed in the nineteenth century by Ferdinand de Saussure.

That leap in time is no accident. In fact, for centuries the pursuit of knowledge was repressed in the western civilization. In this respect, the Italian historian Carlo Ginzburg tells an interesting story. According to him, the Epistle to the Romans 11:20, one biblical text written by the apostle Paul, exhorted the Romans converted to Christianity not to despise the Jews. The text in Greek was translated quite literally into Latin, which resulted in a misunderstanding with serious consequences (GINZBURG; CAROTTI, 1990, p. 95–98).

.. 'sapere' was understood not as a verb with moral meaning ('be wise') but as a verb of intellectual meaning ('knowing'); the adverbial expression 'altum', on the other hand, was understood as a noun meaning 'that which is at the top '. [...] So the moral condemnation of moral pride pronounced by St. Paul became a reproach against intellectual curiosity. [...] We, therefore, find ourselves before not an individual

lapse, but one collective or quasi-collective. The slip of words [...] was certainly favored by linguistic and textual factors [...], the human mind is like a computer that operates in basis of a yes-no, all-or-nothing logic. Even though modern physics may already be sufficiently immune to anthropomorphism not to be bound by such logic, humans continue to behave and think in such way. For them, reality as reflected by language and thus by thinking, is not a continuum but a field covered by discontinuous categories, substantially antithetical.

After the Dark Ages and further accentuating the importance of considering both speaker and listener to understand the language as a means of communication and transmission of knowledge, another lesson worthy of attention is that of the writer, sociologist, literary critic, French philosopher and semiotician Roland Barthes, dated 1978, regarding the role and importance of Saussure's linguistics to the study of (BARTHES, 2003, p.90-92):

aybe someone still remembers (for it is well out of fashion): Saussure formulated with clarity the langue-parole opposition: light and subtle dialectic between the speaking individual and the speaking mass. Since then, Saussure, if not attacked, was at least 'emptied' by different waves of research: Chomsky (competence-performance), Derrida, Lacan (la-langue). I personally believe that something is unwavering in this opposition: the need for two seats, two places in a dialectic relationship: 1) a repository, where the laws of community's language are kept (a sort of tabernacle); 2) a time for updates, choice of subject, collection within the reserve [...] 'mundane' rules (logic, convenience, dialectic about listening to the other, image games, etc.)..

4.1 Structuralism in Linguistics

Ferdinand de Saussure, Swiss scholar, lived between the late eighteenth and early nineteenth century and is regarded, almost unanimously, the father of the study of linguistics as it is now known (FALK, 2003). Many of his proposals haven't yet been overcome by the scientific community and some of them are particularly important to this work.

Saussure argued that the relationship between signifier and signified is arbitrary, as can be seen in one of his writings (SAUSSURE, 1983 apud ASHER, 2005, Vol. X p. 758):

here is no internal connection, for example, between the idea 'sister' and the French sequence of sounds s-ö-r which acts as its signified. The same idea might as well be represented by any other sequence of sounds. This is demonstrated by dif-

ferences between languages, and even by the existence of different languages. The signification ‘ox’ has as its signal b-ö-f on one side of the border [between French and German-speaking regions] but o-k-s (Ochs) on the other side.

The fact that there are compound words and onomatopoeia is not ignored by the author, but the proposition that the relationship between ideas and their representations in Natural Language is essentially arbitrary has important implications. It implies, for example, that the only way to know the meaning of all words of a language is to list the totality of the signifier-signified tuples pertaining to it. It also means that the learning of all such pairs in a language does not have any use in the learning of other languages.

Saussure’s understanding about the study of language opened up new frontiers for researches on the structure of languages themselves, rather than comparative studies between languages or the establishment of grammar rules determining correct writing. The Generative Theory proposed by Chomsky has established particularly important ideas for this current project.

4.2 The Generative Theory

Noam Chomsky laid the foundations of the theory of syntax known as Generative Grammar, still in gestation (CHOMSKY, 1965). On doing so, the intention is to answer three questions considered to be fundamental:

a. What constitutes knowledge of language?

b. How is the knowledge of language acquired?

c. How is the knowledge of language used?

The author argues that these questions are not even approached by Traditional Grammar, as it is focused primarily on listing the uses of language which are considered formally correct. The knowledge of language is accepted as complete and pre-existing, and grammar textbooks are aimed at restricting the use of that which is known to those forms which are defined as correct (CHOMSKY, 1986).

By answering the questions enumerated above, the Generative Theory proposes that the knowledge of language is precisely the knowledge of the Generative Grammar – the set of rules that generates all valid constructs of a language – and that such knowledge is acquired through an apparatus innate to all human beings, a set of rules named the Universal Grammar.

The third question, about how the knowledge of language is used, is initially dealt with by the T-Model (figure 4.1) proposed by Chomsky.

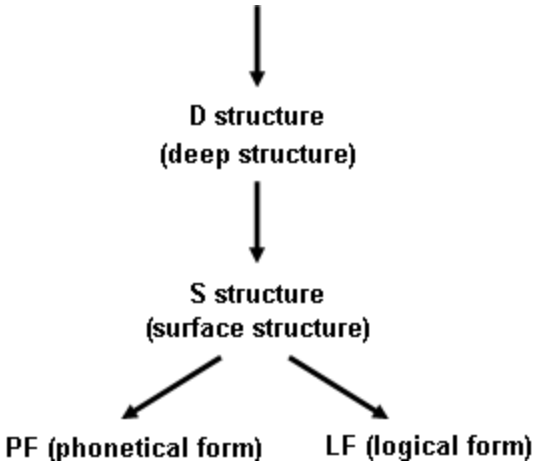

Figure 4.1: T-Model of Language (CHOMSKY, 1986).

According to the T-Model, the process of language use begins at the D-Structure, the organization of abstract ideas that are to be communicated. Through transformational rules, these structures may be mapped to S-Structures, closer to those used by humans in the communication process. Thereafter, these structures can be converted into the phonetic form and logical form of what is to be communicated.

In the 1990s, Chomsky began the Minimalist Program, in which several of these concepts, especially the T-Model, were abandoned, and several others were and are still being revised. Regardless of the specific models generated by the Generative Program being able to answer those questions that set it into motion, its motivation and the assumptions made to explain the problems under study are among the most promising lines for the explanation of the human language phenomenon.

Considering the current state of ongoing development of the theory it is not feasible to apply, in the current work, the grammar models already produced by the Generative Program. However, the hypothesis underlying the Universal Grammar is a very compelling solution to “Plato’s problem”, and is in line with theories of learning developed in other areas of knowledge (section 2.1). The theory of modularity of the human brain and the nativism of certain human behaviors – one of the biggest barriers to the acceptance of Universal Grammar – is gradually being accepted by researchers (FODOR, 1983).

The theories and arguments presented in this section are the basis for the choice, in this work, of assuming that there is a Universal Grammar encoded in the human mind and that the understanding of natural languages requires at least the emulation of the device which processes it.

4.3 Language in the human brain

As in all branches of knowledge, there are critics to the cerebralist conception of mind. For them, the brain is an instantiator of language rules, a mechanism responsible only for the physical processing of something that isn't created there. That, however, has not been the understanding of most scholars (NERO, 2002). Del Nero, psychiatrist, masters in philosophy and Ph.D. in electrical engineering from the University of São Paulo, states (NERO, 2002):

anguage is one of the great architects of mind and culture. Initially a concrete, exclusive department of human brain, it becomes virtual through exposure to the environment. While it carries within the potential ability to recognize the propositional nature of a sequence of symbols, it must further equip itself to handle surface rules of grammar, meanings and discourses.[1]

The knowledge we have about language processing in the human brain has been learned, as is common in neurological research of Man, from the cognitive tests performed in individuals who have suffered brain damage. There are several detailed studies of such cases (APHASIA, 2008; CARAMAZZA; ZURIF, 1976).

One such instance was remarkable in the nineteenth century for having demonstrated, in a pioneering way, the link between brain injury and a specific limitation of rationality. The case of Phineas Gage, in 1848, has been widely recorded. Phineas was a 25-year-old construction worker that, at those times, settled rails for a railroad company in the United States. He was a strong, healthy man, precise in his movements, as well as sociable and friendly. That is, until the explosion of a rock hurt him. An iron bar, about one meter long, having a diameter of nearly three centimeters and weighing six kilograms, pierced his skull entering from his left eye and leaving through the top of his head. Two months later, Gage left the hospital physically recovered and presenting no problems with speech or understanding. However, he became irreverent and began to insult people. The story, rich in detail, is told by Antonio Damasio, a Portuguese neurologist settled in the U.S. He explains (DAMÁSIO, 1999, p.30–31):

hile other neurological injuries that occurred at those same times showed that the brain was the foundation of language, perception and motor functions [...], the history of Gage suggested this astonishing fact: in a sense, there were systems in the human brain more devoted to reasoning than others [...] The changes Gage's in personality

[1] Originally in portuguese: *A linguagem é um dos grandes artífices da mente e da cultura. Inicialmente departamento concreto exclusivo do cérebro humano, torna-se virtual pela exposição ao meio. Se carrega consigo a capacidade potencial de reconhecer a natureza proposicional de uma sequência de símbolos, posteriormente deverá equipar-se para manipular regras superficiais da gramática, significados e discursos.*

were not subtle. He was no longer able to make the right choices...[2]

This section intends to describe the essentials on the subject to clarify the reasons for the proposed work.

According to what is accepted as this text is being written (research in cognition has evolved rapidly in the last decades), language recognition by the human brain occurs mainly in two regions: Broca's Area, located in the frontal lobe of the cortex, and Wernicke's Area, located in the cerebellum, around the auditory cortex (APHASIA, 2008). These two areas communicate through a neural way called Arcuate Fasciculus. Lesions in each of these regions result in three types of aphasia: Broca's Aphasia, Wernicke's Aphasia and Conduction Aphasia, respectively (figure 4.2).

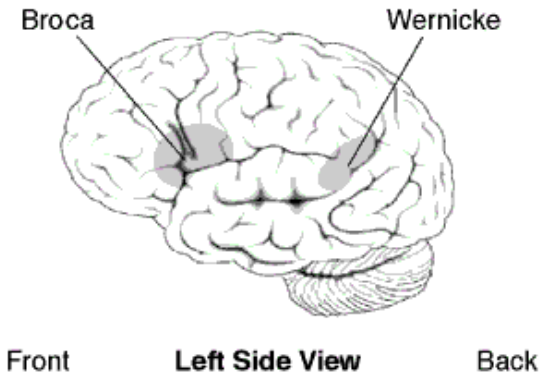

Figure 4.2: Wernicke's and Broca's areas (APHASIA, 2008).

Broca's Aphasia and Conduction Aphasia are very similar, mainly affecting the ability to organize and understand the structure of sentences. People who suffer from these problems typically communicate using very simple and concise sentences, devoid of articles, inflections and verb conjugations (CARAMAZZA; ZURIF, 1976). More detailed tests indicate that, while communicating, these individuals apparently base their understanding on possible semantic contents of each sentence, showing great difficulty to use syntactic information to make choices when faced with ambiguity.

Wernicke's Aphasia is a more debilitating disorder: its victims usually face an immense difficulty communicating. They are capable of constructing structurally complex discourses completely devoid of meaning. It is not uncommon that these individuals make use of indefinite subjects in sentences where none is implicit, as well as using anaphora without a recognizable referent. It seems fairly clear to the specialists

[2]The original, in portuguese: *Enquanto outros casos de lesões neurológicas ocorridas na mesma época revelaram que o cérebro era o alicerce da linguagem, da percepção e das funções motoras [...], a história de Gage sugeriu este fato espantoso: em certo sentido, existiam sistemas no cérebro humano mais dedicados ao raciocínio do que a quaisquer outros [...] As alterações na personalidade de Gage não foram sutis. Ele já não conseguia fazer escolhas acertadas...*

in the subject that the capacity for syntactic constructions and the understanding of the semantic roles of the words is not affected, but instead the comprehension of the meaning of the words themselves is impaired, often preventing communication. There are reports from patients recovered from such ailment and its effects saying that, although they recall having had conversations while suffering from Wernicke's Aphasia, they could understand nothing of what either others or they themselves had said.

There is another curious feature of Wernicke's aphasia deserving of note: it does not affect the ability to sing songs that were familiar to its victim before the occurrence of the injury. However, in general, it is rejected that such phenomenon is linked to understanding (HEBERT et al., 2003).

The differences between the disorders described above lead scholars and authors to the conclusion that the language processing in the brain is done through two complementary approaches. It is speculated, also because of the topological organization tendency that is perceived studying the human brain, that the understanding of the semantics of words and their roles takes place in Wernicke's Area, while syntactic structuring happens in Broca's area. The Arcuate Fasciculus then would be just the connection path between those two areas (HEBERT et al., 2003).

The studies described above corroborate to the hypothesis that the human mind is modular and that it possesses some innate components, in agreement with the propositions by Chomsky and Fodor laid out in section 4.2. The proposal in the current project is to build a structured device to perform the analysis of the meanings of words in a stage apart from the analysis of the structures of sentences, in a manner analogous to an individual with Broca's Aphasia. Deprived of a syntactic interpreter, that device would recognize only the general meaning of clusters of words, and would be very vulnerable facing ambiguity. Without the module that contains the semantic associations of words, it would be completely incapacitated.

Part II

Experiments

5 Implementation of a Solomonoff's Predictor restricted to Regular Languages

One of the most significant contributions of this work is the implementation of a faithful version, albeit restricted, of Solomonoff's Predictor. No other implementation of such device, proposed 46 years ago in (SOLOMONOFF, 1964), could be located, notwithstanding the number of published papers that refer to it (LI; VITÁNYI, 1989; LI; VITÁNYI, 1997; HUTTER, 2005; RISSANEN, 1983; MINSKY, 1967; CHAITIN, 1997) – there is an informal attempt, called *Solomonoff-lite Evaluator*, that can only be found in a private Internet site (HAY, 2007). However, a quick inspection of the source code of that implementation shows that its methods for generating theories are not consistent with Solomonoff's proposal.

In this section, we describe the implementation of Solomonoff's Predictor that was made for the current work, where the device has been restricted to the induction of theories with the computing power of Regular Grammars. Subsequently, results of tests of such implementation are presented, along with measures of processing time for input strings of different lengths: an empirical indicator of the computational complexity of the program.

It is important to keep in mind that Solomonoff's Predictor, as proposed in (SOLOMONOFF, 1964), is complete, but incomputable (SOLOMONOFF, 1975). It originally operates over recursively enumerable languages. The reduction of the model to compute only hypotheses belonging to the universe of Regular Languages makes it computable, even though its asymptotic computational complexity is $\Omega(e^n)$. Solomonoff demonstrates (SOLOMONOFF, 1986) that these two characteristics are mutually exclusive: being complete and computable. The model implemented in this work is, therefore, necessarily and intentionally incomplete. That does not mean that it can't produce useful results for a large set of problems that are very common in practice. Its interest as a universal predictor, however, is intentionally compromised to guarantee its computability.

5.1 Description of the program

As has been stated in section 2.1, Solomonoff's Theory of Prediction uncovers a way to calculate the probability of each algorithm able to explain a set of events, and in doing so delineates a predictor device. In its simplest form, such device blindly scans the space of algorithms, verifying the *a priori* probability of each element and the set of valid outputs it produces. The *a priori* probability of each hypothesis is given by equation (2.4), reproduced below:

$$P(H) = 2^{-|H|} \tag{2.4}$$

The probability of a hypothesis being the one to have generated an event can be calculated through Bayes' Theory, represented by equation (2.1) and reproduced below:

$$P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)} \tag{2.1}$$

The device derived from Solomonoff's Theory of Prediction combines these two equations to the space of outputs produced by each hypothesis (ie, the universe provided by it), obtaining $P(H|E)$: the probability that an hypothesis correctly describes an unknown generator given the observed events such generator has output.

The program that was developed for the current work implements the simplest form of the predictor device exposed by Solomonoff. The space of generator-describing algorithms is restricted to that of codes processable by a machine with computing power equivalent to that of regular grammars, reducing the quality of the theories obtained, but greatly simplifying the processing.

The choice for the space of Regular Languages as a restriction was based on the simplicity of that class of formal languages in relation to the outer levels of Chomsky's Hierarchy. There are a number of qualities due to such simplicity. The most important ones are the existence of low cost algorithms for obtaining the optimal Deterministic Finite State Automaton (DFA) that produces a Regular Language, the unicity of such optimal device and the linear computational complexity of the process for accepting a string (HOPCROFT; MOTWANI; ULLMAN, 2000). All these features were exploited to reduce both computational and implementation complexity. The general structure of the implemented device can be seen in algorithm 5.1.

Algorithm 5.1: Solomonoff's Predictor – General structure

```
for h in hypotheses
  fda = create_finite_deterministic_automaton(h)
  P(E|h) = fda.probability(E)
  for event in E
    if P(event|h) = 0
      remove_hypothesis(h)
      P(h|E) = 0
      break
    else
      P(h|E) += P(event|h)*P(h)/P(event)
```

A detailed description of the implemented predictor's core is shown in algorithm 5.2. It takes the form of a class, *Predictor*, containing as attributes the alphabet of terminal symbols with which to work, the number of occurrences of each observed event, the probabilities of all the theories already formulated by the device and length of the longest among them. All of the complexity of this class is in the *observe* method, which is responsible for changing the internal set of theories of *Predictor* as new events are presented.

Each call to the *observe* method of an instance of *Predictor* alters the object's internal state. The probabilities of theories compatible with the observed events are recalculated to account for the new event. If it is possible to obtain a better (more probable) theory that is longer than the ones that have already been explored, the *observe* method continues its exploration of the space of regular expressions. At the point in which the length of the theories could only reduce their probabilities, or would exceed the limit imposed during the method call, the search for hypotheses is interrupted.

Algorithm 5.2: Solomonoff's Predictor - *observe* method

```
Predictor::observe(new_event, maximum_hypothesis_length)
  for theory in this.theory_probabilities
    # Create the minimum deterministic finite state automaton (DFA) which
    # recognizes the code being processed
    dfa = minimum_deterministic_finite_automaton(this.alphabet, theory)
    valid_outputs = 0
    total_outputs = 0

    # Exercise the DFA - verify how many strings it recognizes before reaching
    # the length of the longest input string (total_outputs), and, among them,
    # how many are equivalent to each of the input strings (valid_outputs)
    dfa.compute_string_counts(new_event, valid_outputs, total_outputs)

    if valid_outputs == 0
      theory_probabilities.remove(theory)
    else
      theory_probability = (1/length(alphabet))^length(theory)
      this.theory_probabilities[theory] = ((this.theory_probabilities[theory] *
```

```
            this.observed_event_count) + (theory_probability * valid_outputs /
            total_outputs)) / (this.observed_event_count+1)
19
20        if this.theory_probabilities[theory] > best_theory_probability
21          best_theory_probability = this.theory_probabilities[theory]
22          best_theory = theory
23
24    if this.event_ocurrences[new_event] > 0
25      this.event_ocurrences[new_event]++
26      maximum_length = this.longest_theory_length
27    else
28      this.event_ocurrences[new_event] = 1
29      if this.observed_event_count == 0
30        maximum_length = length(new_event)
31      else
32        maximum_length = this.compute_maximum_length(this.longest_theory_length+1,
              this.longest_theory_length + length(new_event) + 1,
              probability_best_theory)
33      maximum_length = min(maximum_length, maximum_hypothesis_length)
34
35    this.observed_event_count++;
36    generator = regular_expression_generator(this.alphabet,
          this.longest_theory_length, maximum_length)
37    while generator.next_valid(theory, keys(this.theory_probabilities)) &&
          length(theory)<=maximum_length
38      theory_probability = (1/length(this.alphabet))^length(theory)
39      dfa = minimum_deterministic_finite_automaton(this.alphabet, theory)
40
41      valid_outputs = []
42      total_outputs = 0
43      dfa.compute_string_count(keys(this.event_occurrences), valid_outputs,
            total_outputs)
44      probability_hypothesis_given_events = 0.0
45      for event in this.event_occurrences
46        if valid_outputs[event] == 0
47          probability_hypothesis_given_events = 0.0
48          break
49        else
50          probability_hypothesis_given_events += (probability_theory *
                valid_outputs[event] * this.event_occurrences[event]) /
                (total_outputs * this.observed_event_count)
51      if probability_hypothesis_given_events > 0.0
52        this.theory_probabilities[theory] = probability_hypothesis_given_events
53
54        if length(theory) > this.longest_theory_length
55          this.longest_theory_length = length(theory)
56
57        # if we find a theory which is better than all previous ones, it is
              possible
58        # that we might set a new, smaller limit for the length of the hypotheses
59        if probability_hypothesis_given_events > best_theory_probability
60          best_theory = theory
61          probability_best_theory = probability_hypothesis_given_events
62          maximum_length = this.compute_maximum_length(length(theory),
                maximum_length, best_theory_probability)
```

The Predictor's *observe* method processing begins at line 2, where there is a loop

responsible for recalculating the probabilities of all the theories already computed considering the new observed event. The minimum Deterministic Finite State Automaton equivalent to the regular expression representing the theory being tested is created in line 5. That automaton is exercised in line 12, from which are obtained the total number of outputs computed by the automaton and how many of them had the observed event as its prefix. If the theory being tested is not capable to explain the new event (none of the outputs had the event as its prefix), it is discarded. Otherwise, the theory's global probability relative to all observed events is recalculated to take into account the new event using equation (2.1) at line 18. Between lines 24 and 33 an upper limit for the length of the hypotheses is established in a way that is described further ahead in this report. That limit is used for restricting the exploration of the space of hypotheses that hasn't yet been visited to the minimum area that can possibly yield improvements. In the loop on line 37, new hypotheses are generated by applying the Kleene star to the symbols of language. Next, we calculate the probability of the hypothesis given all the known events in a manner analogous to what is done in the loop of line 2, and, if the hypothesis is able to describe all the observed events, it is added to the list of valid theories along with the probability $P(H|E)$ that's been assigned to it (line 52). If the new hypothesis to be the best found thus far, the limit for the length of hypotheses to be visited is updated accordingly.

The initial limit set for the length of the hypotheses to be investigated is calculated between lines 24 and 33 of algorithm 5.2. If the event had already been observed, because of the way the algorithm is constructed, all hypotheses with a chance of besting all known candidates will already have been visited. In that way the space to be investigated is limited according to the length the best hypothesis tested thus far, effectively interrupting the exploration of the infinite space of hypotheses. If the event is new, there are two possibilities: either the new event is the first to be observed or not. If the event is the first to be observed by the predictor, the greatest useful length of description is the length of the event itself: it is a trivial hypothesis that will invariably maximize the term $P(E|H)$. Any longer hypothesis will present lesser *a posteriori* probability. If the event is not the first one, the *compute_maximum_length* is invoked (algorithm 5.3), setting as lower limit the length of the best theory that's been explored incremented by one (after all, hypotheses that have not yet been explored are to be processed) and as upper limit the length of the $best_hypothesis_computed|new_event$ string, which maximizes $P(E|H)$. The algorithm 5.3 has a rather simple internal logic: for each length between the given lower and upper limit, it checks whether there might be a hypothesis whose *a posteriori* probability exceeds the best theory found thus far. If the method finds a limit to the improvement of the hypotheses below the upper limit

passed as a parameter, it is returned. Otherwise, the original upper limit parameter itself is returned.

Solomonoff proposed in (SOLOMONOFF, 1989) that the predictor should be implemented to be incremental and to accept external interference to its execution time, making it possible to obtain the best response found up to a given moment – something very interesting, given it is impossible to predict the time required to find the absolute best answer. Such proposal was incorporated into the algorithm 5.2 through the *maximum_hypothesis_length* argument that is passed to the *observe* method. The interruption of the predictor's processing can not thus be done arbitrarily, but still there is a direct way to interfere with the system's processing time. In the common case, most of the processing time is spent generating and testing new hypotheses against all events already observed. Interrupting the exploration of new hypotheses, the time required for adding new events tends to grow much more slowly.

Algorithm 5.3: Solomonoff's Predictor - *compute_maximum_length*

```
1 Predictor::compute_maximum_length(lower_limit, upper_limit,
    best_theory_probability)
2   for theory in this.theory_probabilities
3     if lower_limit >= upper_limit
4       break
5     maximum_probability = (1/length(this.alphabet))^lower_limit
6     if maximum_probability < best_theory_probability
7       return lower_limit - 1
8   return upper_limit
```

The behavior of algorithm 5.3 explains why, at the end of the loop on line 37 of algorithm 5.2, the *obtem_comprimento_maximo* method is invoked to update the maximum length of hypotheses in case a theory has been found that overmatches all previous ones. It is possible that the probability of the new hypothesis exceeds the limit of what one can find in the space that is to be explored. The cost of exploring larger portions of the hypotheses space is very high, justifying the attempts to reduce it.

There are three critical points in algorithm 5.2: the generation of theories, in line 37; the creation of a device capable of executing the theory, in line 5; and the exercising of each theory, in line 12. These three points are the main reason for the computational complexity and difficulty of implementing a predictor as proposed by Solomonoff. They are also the main motivators for choosing Regular Languages for the presented implementation.

The generation of theories can be made at first as a simple Kleene star closure over

the language of description of these theories. The implementation is apparently simple, but the result set is infinite. It is necessary, for the feasibility of the system, to restrict the universe of theories to be searched. In the implementation shown, such restriction was made by limiting the maximum length of the theories, as described above on the exposition about the operation of the algorithm.

The second and third critical points from algorithm 5.2 cited above – the generation of theories and exercising of them in a device that can extract their every possibly interesting output – complicate the implementation of the system for similar reasons. It is necessary, first of all, to ensure that the device that will be used for exercising hypotheses will terminate processing in a finite period of time. Is not feasible to start a process that may fall prey of Turing Machine's Halting Problem (LI; VITÁNYI, 1997). There are techniques that could be used to deal with that issue, such as alternating the processing of all known theories (SOLOMONOFF, 1989), yet the cost of processing tends to the impractical. The Deterministic Finite State Automaton doesn't suffer from these problems. It is able to recognize any string that belongs to the language that defines it in linear time proportionally to the length of that input string (HOPCROFT; MOTWANI; ULLMAN, 2000). Moreover, obtaining such device, although computationally expensive (the computational complexity of converting a non-deterministic FA, which is the natural translation of a regular expression, in a deterministic FA, is of exponential order (HOPCROFT, 1971)), is easily implemented.

5.2 Results

The implementation done in this work of a Solomonoff's Predictor restricted to Regular Languages was validated in a series of tests that, although simple, should prove its effectiveness. It is possible to verify divisibility within the realm of integer numbers using Finite State Automata. The set of integers divisible by two, for example, is defined by a regular language. What was done to test the program was to feed it with sequences of numbers written in binary form, divisible by a certain integer. The intention is to verify the speed of convergence to the correct regular expression, both in relation to the number of events and to the actual time of execution.

5.2.1 Processing time

The processing time of the program depends primarily on the number of tests that must be done to validate the hypotheses against the observed events. The asymptotic computational complexity of the transformation of each regular expression in the minimum

equivalent Deterministic Finite Automaton (DFA) is $O(e^n)$, but in practice, its asymptotic behavior is not the main factor that contributes to the processing time. $|\Sigma|^n$ strings are generated for each length n of hypothesis, and they all need to be transformed into minimal DFA for testing all possible outcomes. Thus, it is observed that the main responsible for the processing time, in the common case where there are few events compared to the number of possible hypotheses, is the number of hypotheses that need to be checked.

To verify the evolution of the processing time, the program was run with strings that would force the generation of hypotheses up to its own length. The intent of these tests was to obtain empirical data to base the choice of tests possible within the available resources, so that they were executed only once each, with manually chosen inputs. The graph of execution time according to the length of the input (and thus the maximum hypothesis length explored) is shown in figure 5.1.

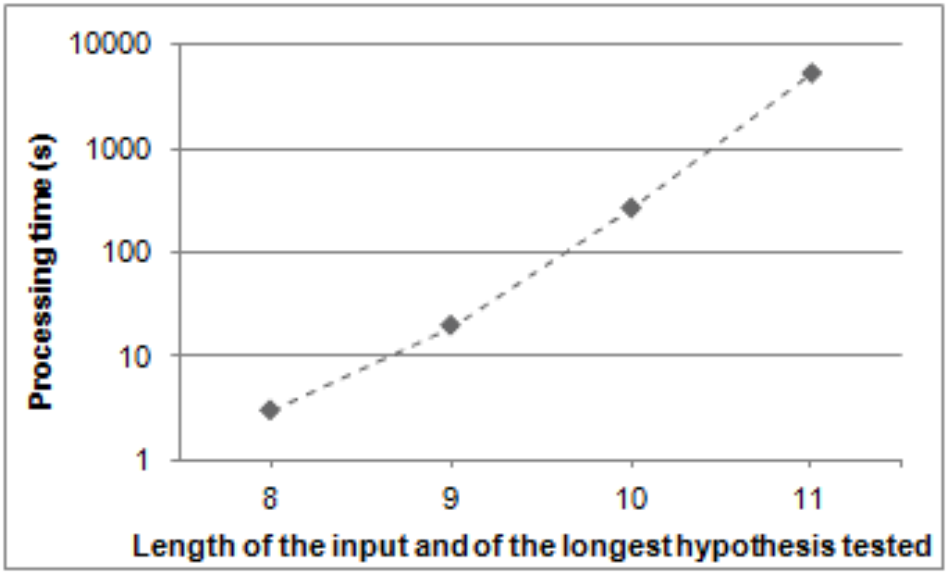

Figure 5.1: Chart of the processing time of the predictor against the length of the longest generated hypothesis.

Processing times of less than one second were omitted from the chart in figure 5.1 to allow for visualization in logarithmic scale. The logarithmic scale, in its turn, was chosen to emphasize the similarity between the curve of progression of the processing time obtained empirically and expected behavior for an algorithm with asymptotic computational complexity $O(e^n)$. The practical conclusion of these tests is that, given the resources available for this work, it is not appropriate to plan functional tests that require the generation of hypotheses with more than 11 symbols.

5.2.2 Test with a synthetic language

The computational complexity of Solomonoff's Predictor greatly limits the universe of feasible tests. With that in mind, we decided to check, initially, the behavior of the prototype device with a synthetic language without semantic interest. The proposal is to check how many examples of the $(01)*$ language are needed for the device to reach such a regular expression.

Initially, string "01" was presented to the predictor, to which it output as its sole hypothesis "01". Then the string "0101" was fed as input to the predictor input, in response to which it output the analogous string "0101". Feeding the device with both strings, the description for the proposed language was already output as the first among the results, as shown in table 5.1.

Hypothesis	**Probability**
$(01)*$	0.0078125
$\|(01)*$	0.00390625
$01\|0101$	0.00390625
$0101\|01$	0.00390625
$01(\|01)$	0.00390625
$01(01\|)$	0.00390625
$0(\|10)1$	0.00390625
$0(10\|)1$	0.00390625
$(\|01)01$	0.00390625
$(01\|)01$	0.00390625

Table 5.1: Top Ten hypotheses proposed by the predictor to the sequence of strings "01 0101".

The result presented in table 5.1 required a processing time of $0.43s$. It is certainly a very simplified example that can not be extended to all situations. However, the performance of the predictor to find a hypothesis capable of describing the observed events can not be ignored. Even if the original generator of the events exposed to the predictor were substantially more complex, it is undeniable that the theories considered most likely by the device are consistent and elegant in their simplicity.

5.2.3 Theories generated for testing divisibility by two

Once the initial operation of the implemented predictor had been verified, it was decided to test a language defined by a simple mathematical problem: divisibility, within the set of Natural Numbers, by two. Keeping the alphabet used in the previous sections, numbers will be represented in binary form.

As discussed in section 2.1, it is expected that Solomonoff's Predictor induces the

algorithm which generates a sequence of events by observing a relatively small number of them. The proposed test thus consists in feeding the predictor with a sequence of binary numbers divisible by two. One of the shortest regular expressions that define such regular language is "$(1|0)*0$" – it includes all strings consisting of zeros and ones that end with a zero.

The input to the predictor now consisted of all even numbers in the $[0,126]$ interval, and a limit of 7 was imposed to the maximum length of hypothesis – that is the length of the regular expression that is known to recognize all such events. Each value was repeated with all string lengths up to 7 symbols (length of the number 126 in binary), being each repetition padded with a prefix of non-significant zeros. As such, the information that leading zeros aren't significant was implicit in the sample. The strings were fed to the program in groups of increasing size in lexicographic order. As a result, the first set of events input in the program was $\{0\}$, then $\{0,00\}$, $\{0,00,10\}$ and so on. A chart showing the evolution of the position of the correct hypothesis for the generator of such data ("$(1|0)*0$") is shown in figure 5.2.

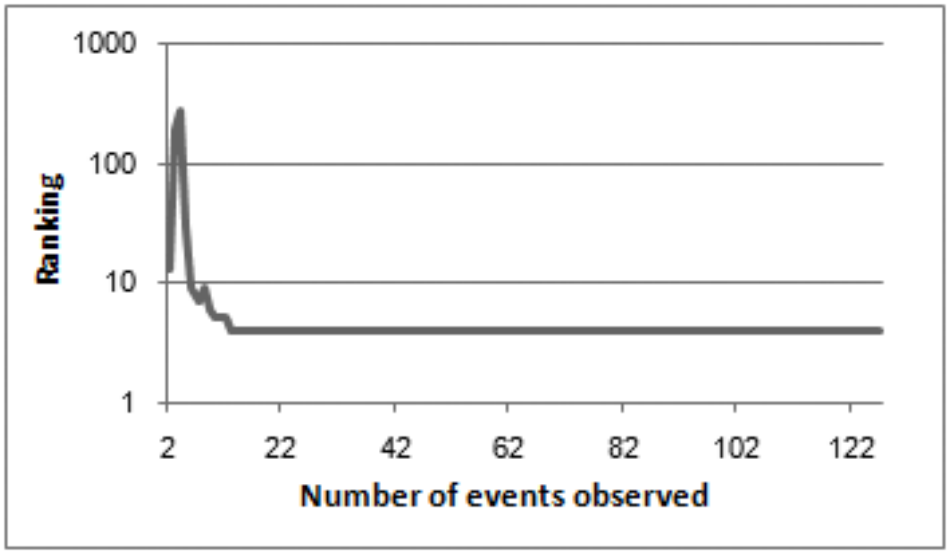

Figure 5.2: Chart of the evolution of the hypothesis "$(1|0)*0$" of the predictor as the number of observed events is increased.

A very interesting fact can be observed in the results: the hypothesis that actually generated the sequence of events has not reached the top position among the alternatives proposed by the predictor, even after its exposition to 126 events. To understand the phenomenon, one must see the list of the major contending hypotheses, shown in table 5.2.

The difference between the language accepted by the regular expression "'$(1*0)*$" and that which is accepted by expression "$(1|0)*0$" (or the equivalent "$(0|1)*0$") is the empty string. It is remarkable that a hypothesis so close to the correct one tops those

Hypothesis	Probability
$(1*0)*$	0.000143537
$(1\|0)*$	0.000111827
$(0\|1)*$	0.000111827
$(1\|0)*0$	$7.56032e-05$
$(0\|1)*0$	$7.56032e-05$
$\|(1*0)*$	$7.17684e-05$
$\|(1\|0)*$	$5.59133e-05$
$\|(0\|1)*$	$5.59133e-05$
$(\|1\|0)*0$	$3.78016e-05$
$(\|0\|1)*0$	$3.78016e-05$

Table 5.2: Top Ten hypotheses proposed by the predictor to the problem of binary numbers divisible by two, given 126 samples.

proposed by the predictor after it having been exposed to only 12 events. However, it is undesirable that shorter, inaccurate hypotheses dominate the list for so long. It is the negative side effect of choosing a language as broad as that of integers divisible by two – half of the existing binary numbers is divisible by two, so that a regular expression that recognizes absolutely all binary numbers, "$(1|0)*$", has a probability of only 50% of recognizing a string that is not divisible by two. The length of the hypothesis in this case makes it so that more general theories are preferred to correct ones even after the observation of a relatively large number of events. The tests were continued feeding up to 254 samples to the program without changing the ordering of the best ranking hypotheses shown in table 5.2.

The result obtained is unexpected as it evidences a class of problems in which the performance of the predictor is inferior to that expected. However, the predictor proposed very consistent hypotheses for the observed events. The one favored by the predictor after the observation of 12 events is particularly interesting because, while extremely simple, it describes very accurately the proposed Regular Language. Even in a test that unveils a case of slow convergence, it is difficult to ignore the qualities of the device.

6 Implementation of an Information Retrieval System Based on Latent Semantic Analysis

As described in section 3.1, LSA is a method used to map a *corpus* into a vector space with a pre-defined, restricted number of dimensions. To create such representation, it is necessary, in first place, to process the documents in the *corpus*, storing data about the words that occur in each text and consolidating these data into a matrix. Such matrix serves as input to the SVD method, whose result, truncated, provides the desired approximation of the vector space determined by the *corpus*. The result should consist of a series of matrices that can be used for query processing.

Because of the decoupling between the initial processing of a *corpus* and query processing, the implementation of information retrieval systems traditionally consists of two separate modules (YATES; NETO, 1999). The first module, called *indexer*, is responsible for processing the *corpus* itself and generating an index for searching. The second module, the *query processor*, expects as input a string of terms and returns an ordered list of documents without need of direct access to the *corpus*, only to the index generated by the *indexer*. These two modules, represented in figure 6.1, will be individually described in the following sections.

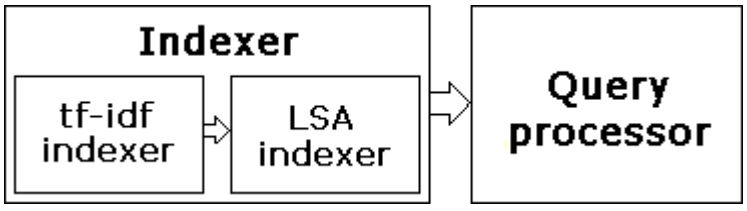

Figure 6.1: Information retrieval system based on LSA.

Once both indexer and query processor had been implemented, a series of tests were applied to them to verify the performance of indexing using LSA. To that end, we used one of Reuters' *corpus* of journalistic texts in English, the *Reuters Corpus*

Volume 1 (**??**), which contains texts published by the news agency between August 1996 and March 1997. This section will be completed with the description of the test methodology and the results obtained.

A copy of the consolidated results of the tests – a few thousands of graphics – will be kept online in a server accessible through the Internet address http://iuri.chaer.org/master-thesis/. Working instances of a term comparator and a search processor will also be kept online in the same server for as long as possible (a timeframe restricted by the consumption of computational resources by the tools) so that others can observe the results of indexing using LSA. The original texts of the Reuters *corpus*, however, can't be made available due to copyright restrictions. If needed, it must be requested directly to the NIST ("National Institute of Science and Technology") United States' agency.

6.1 Description of the indexer

The first step in indexing the *corpus* is simply to generate a sparse matrix relating terms to documents, the normal result of the *tf-idf* indexing method (SALTON; BUCKLEY, 1988). To that matrix can then be applied the method of dimensional reduction of LSA, to produce indexes with any lesser number of dimensions. To that end, a pre-indexer was created, responsible solely for the separation of terms, morphological extraction of their stems and calculation of the *tf-idf* that relates each term to each document. The processed data is stored in structured files substantially smaller than the original *corpus*, reducing the time required for each execution of the LSA indexer, which uses them as input to generate indexes with different numbers of dimensions.

The second phase of indexing consists in applying the LSA method itself. The program responsible for that part basically performs singular value decomposition over the matrix obtained by the first stage, truncates the matrix of singular values keeping only the desired number of dimensions and generates the matrix of terms and documents necessary for query processing. The results are stored in compressed files, so as to ease their loading by the query processor.

6.1.1 The *tf-idf* indexer

To reduce the time spent in disk read operations, the *tf-idf* indexer is built to directly access the compressed archives of the Reuters *corpus*, in the same format as they are distributed. Indexing is done in two passes. In the first one, all terms are counted, keeping tab of both global and local counts. From these counts, it is possible to pre-allocate the space needed for the sparse matrix generated by the *tf-idf* method, as well

as calculate the values assigned to each term in each document. As the test *corpus* contains a significant number of spelling errors, considering a total number of 470 thousand documents, all terms with fewer than 6 occurrences were removed – a rather simplistic heuristic, but one that could be empirically verified to yield good results taking into account the complexity of its implementation. As a result of this module's work, three files are generated:

Lexicon : Index of all terms' stems occurring in the *corpus*.

Document index : An index listing all documents that make up the *corpus*.

Inverted index : A sparse matrix whose rows correspond to terms and columns to documents. The values from this matrix are computed by equation (3.1), reproduced below.

$$|t| = \frac{tf \cdot \log \frac{N}{n}}{\sqrt{\sum W_{t_i}^2}} \tag{3.1}$$

Most of the time consumed by the *tf-idf* indexer is spent on reading and writing operations on the disk. As noted above, the *corpus* is read in compressed format to reduce the impact of reading. For writing, instead of compressing, the choice was simply to use as compact a binary format as possible without reprocessing the data. All reads and writes are done using the file memory mapping mechanism from the operating system.

6.1.2 The LSA indexer

The LSA indexer needs only access to the inverted index produced by the *tf-idf* indexer to complete its work. The most costly part of the process is the extraction of the matrix's singular values. For that task the SVDLIBC library was used, because it is well written and widely used for applications in latent semantic analysis (GLIOZZO; STRAPPARAVA, 2005; TURNEY, 2005). It reimplements the LAS2 method from the SVDPACKC library, and much of the analysis of the algorithm that can be found in the documentation of SVDPACKC (BERRY, 1992) also applies to SVDLIBC.

Solving equation (3.2), the SVDLIBC library provides, given the matrix containing the inverted index, the singular values that are the basis of latent semantic analysis.

$$A = U \cdot \Sigma \cdot V^T \tag{3.2}$$

Once the U and Σ matrices are known, they are truncated (a process described in section 3.1) and their $\cdot U \cdot \Sigma^{-1}$ product is stored in the query processing files, as it is needed to transpose new documents to the query vector space. That same result is also used for the transposition of the *corpus* documents, d_k, following equation (3.5).

$$d_k = d^T \cdot U_k \cdot \Sigma_k^{-1} \tag{3.5}$$

These two matrices are stored in two files. Those files are the core of the LSA-based search engine developed for the current work and, along with the vocabulary and document index produced by the *tf-idf* indexer, form the interface between the indexer module and the query processor.

6.2 Description of the query processor

The query processor module was developed as a client-server application. The client is trivial, consisting only of a visual interface for sending queries to the server and displaying the responses. The server, therefore, is left responsible for the entire complexity of query processing, from the preparation of the string of terms that composes the query itself to the evaluation and organization of the results.

Each string of terms sent by the server is treated as a new document to be compared with others of the *corpus*. The program initially loads into memory the data files containing the lexicon L (terms index), the reduced document matrix D_{LSA}, and the matrix M_{LSA}, used to map new documents to the vector space in which the search will be done using latent semantic analysis. Upon receiving a query, it separates the terms of the query string and applies the Porter Algorithm to obtain the stem of each term. From the lexicon L output by the indexer, it creates a sparse vector q that represents the query in the same way that a document would be represented in the *tf-idf* model. Multiplying the q vector by the M_{LSA} matrix, the vector q_{LSA} is obtained, representing, within the new vector space, a document consisting solely on the input query. The program then calculates the cosine of the angle between the query vector q_{LSA} and the LSA vectors of each document pertaining to the *corpus*. The greater the cosine of that angle, the greater the similarity between the query and the document within the LSA model. Documents are ordered by similarity and the result is returned to the user.

The implementation of the query processor was made in two programming languages. The client module was written in PHP to simplify the development of a remotely accessible, rich visual interface. The server module was developed in C++,

taking advantage of those libraries that had already been written for the indexer's common tasks. The communication between those two parts is done using POSIX sockets. The server development was done keeping optimization in mind. All explicit disk reads are made only once: when the program starts. Still, the $O(n \cdot \log n)$ asymptotic computational complexity of each query (dominated by the sorting made after the end of the calculation of similarity between each document and the query string) results, in the environment used for testing, in a response time around seven tenths of a second for a *corpus* of 473,876 documents. The results of tests will be explored in the next section.

6.3 Results

All tests were performed in a single computer, with the following characteristics:

- **Processor:** Intel E6420, a 64-bit processor equipped with 4MB of level 1 cache memory and a clock frequency of 2.13GHz.
- **Random Access Memory:** 2GB of DDR2 memory with a nominal operating frequency of 800MHz.
- **Storage:** Two 8000 rpm hard drives, with 8MB of internal cache space and 500GB of total storage space organized in a RAID 1 hardware-managed array.
- **Operating System:** Linux, with core version 2.6.24, compiled to take advantage of the processor's 64-bit architecture and configured to use a raw 8GB partition as swap space for its virtual memory system.

The input for the tests made was the first disk of journalistic texts in English of the *Reuters Corpus Volume 1*. It consists of 224 compressed files in *ZIP* format, totaling 577MB of compressed data. Uncompressed, that means 473,876 files totaling 2.3 GB of text encoded using the ISO-8859-1 character set (one which encodes each character with a single byte). Each uncompressed file contains a single journalistic text, including HTML coding for formatting and metadata in XML.

6.3.1 Indexing and objective performance measures

Each text of the *Reuters Corpus Volume 1* is classified by subject and by the economic industries to which it is related. Originally, there are 128 different subject classifications and a tree of 872 industry categorizations. These classifications, however, contain

redundant and meaningless codes, identified by the mapping provided by Reuters itself as *dummy codes*. To perform the tests, the tree of economic industries was converted to a one-level list and the redundant classifications were merged and corrected manually, resulting in 121 subject classifications and 367 industries. Excerpts of the listings of these two classifications can be seen in Tables 6.1 and 6.2.

accounts/earnings
advertising
advertising/promotion
annual results
arts, culture, entertainment
asset transfers
balance of payments
biographies, personalities, people
bond markets
bonds/debt issues
brands
business news
capacity utilization
capacity/facilities
comment/forecasts
...

Table 6.1: Excerpt from the list of classifications in subjects from the test *corpus*.

The first step in testing the system was the indexing of the first disc of the *Reuters Corpus Volume 1*. To observe the effect of the number of dimensions retained in the vector space represented by the LSA matrices, indices were generated for the 60 vector spaces that have dimensions from 10 to 600 in increments of 10, for the 14 indices from 620 to 880 dimensions in increments of 20 and, increasing the sampling resolution in a sector which was found to be interesting, 4 indices were made with 804 to 816 dimensions, another four between 817 and 820 and another 3, in increments of 20 dimensions, between 840 and 880 dimensions. The process of generating those indices involves a single execution of the *tf-idf* module, but 81 runs of the LSA module. The resulting *tf-idf* index has 425MB of data and its generation requires approximately 10 minutes. Sizes and processing times for LSA indices increase apparently linearly as the number of dimensions is made larger, at least for the volume of data processed, as can be seen in figures 6.2 and 6.3[1]. Accordingly, in (BERRY, 1992) it is demonstrated that the computational cost of algorithm used by the SVDLIBC library for extracting singular values of a matrix increases linearly with the increase in the number of dimensions. The computational complexity of the products made between arrays after

[1]The data for the processing of indexes above 600 dimensions were omitted, as they forced the operating system to enter the swap space of the virtual memory system, distorting runtime statistics with an increase in processing time which is not due to the algorithm.

applying SVD, similarly, increases linearly with the number of dimensions, so that the expected asymptotic computational complexity of the process as a whole would be $O(n)$, in agreement with the progression observed empirically for the runtimes.

abrasive products
accountancy and auditing
adhesives
advertising agencies
aero-engines
aerospace
agricultural equipment hire
agricultural machinery
agriculture
agriculture and horticulture
agriculture, forestry and fishing
air transport
aircraft components, not electrical
aircraft hiring and leasing
aircraft maintenance
...

Table 6.2: Excerpt from the list of classifications in economic industries of the test *corpus*.

With the LSA indices ready, tests were ran to verify how the number of dimensions influences the observed quality of results. The classifications in subjects and economic industries made by the Reuters journalists were treated as goals by the system. It is assumed that journalists chose good semantic associations, so that the system can be considered good as it is capable of reproducing the results of those manual choices.

Firstly, measurements were made of the cosines of the angles within the LSA vector spaces obtained for all classifications used by Reuters to characterize the *corpus*. These measures were centered on the average and normalized to be between -1 and 1, so that the type of statistical distribution generated by the method could be assessed. That normalized measure of the cosines was called *distance* – although not immediately intuitive, in polar coordinates the use of the term is justified. Also for the study of the observed statistical distributions, the skewness and kurtosis of all curves were calculated. Those standard measures are widely used to assess statistical distributions (JOANES; GILL, 1998).

It was hoped that it would be possible to find a classic distribution of well-known behavior that fit the obtained data. However, that was not the case. For the classification in subjects, the asymmetry of the distributions ranged from 0.335, for 10 dimensions, to 2.039, for 600 dimensions. The kurtosis also increased with the number of dimensions, ranging from -0.450 and 12.059. Some of the histograms are shown

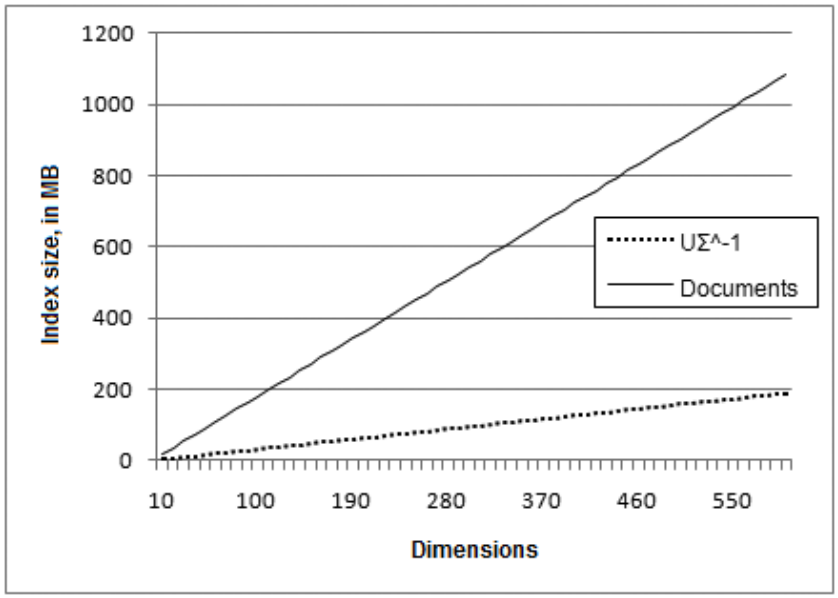

Figure 6.2: Graph of the progression of the indices' size with the number of dimensions.

in figure 6.4. The classification into industries behaved similarly, with the asymmetry varying from 0.172 to 2.423 and kurtosis between -0.526 and 16.356. Histograms similar to those shown for classifications on subjects are displayed in figure 6.5

Although the distributions obtained do not fit into any classical model, their analysis provides some interesting information about the method in relation to the classifications used by Reuters. The first interesting fact is that the distributions of subject classifications and industries are very similar and show very similar behavior as the number of dimensions of the indices is increased, both in relation to the aspect of graphics and on the measures of skewness and kurtosis. Another interesting observation is how LSA tends to measure the distance between texts and the classifications used changes with the dimensionality of the indices. In spaces with a lower number of dimensions, distances are distributed more evenly, while in areas with a higher number of dimensions there is a well marked accumulation around the average (it is important to note that the histograms shown in figures 6.4 and 6.5 are in logarithmic scale, so that the disparity is much greater than it might seem at first sight).

One possible explanation for the distributions seen in the histograms is in the analogy between the dimensions of the spaces constructed using LSA and semantic classifications. A space with fewer dimensions provides fewer possibilities of semantic tags to the texts. A 10-dimensional index gives the possibility of classifying the *corpus* in vectors composed of different grades of intensity for 10 independent concepts. One possible instance of such idea would be to characterize all the terms contained in an edition of the newspaper at different intensities of a set of words such as *{red, politics,*

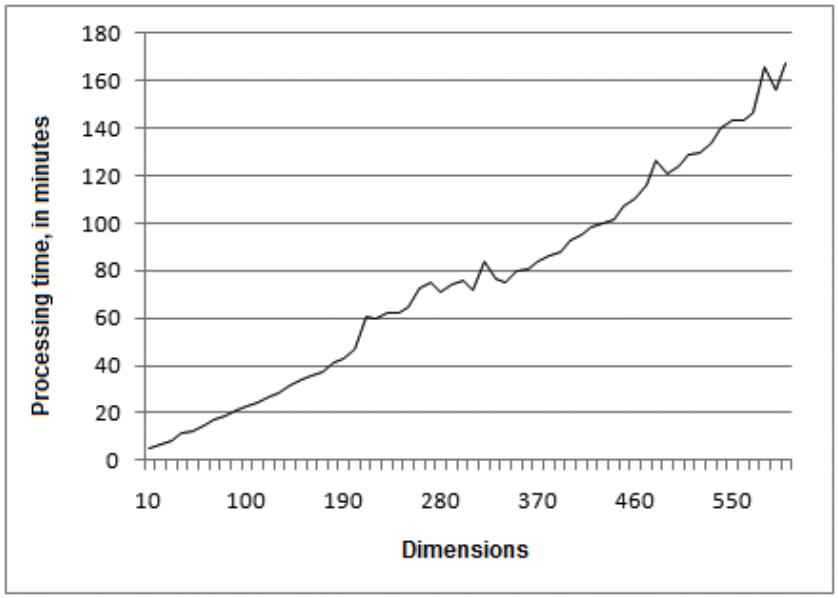

Figure 6.3: Graph of the progression of LSA indexing time with the number of dimensions.

child, car, chair, violence, television, economy, war, climate}. From that point of view, it would seem reasonable that a small number of dimensions resulted in a balanced distribution of distances – after all, none of the dimensions does a really good job describing most of the terms. Results presented in (LANDAUER; DUMAIS, 1997) corroborate with such conjecture.

The search for a classical distribution able to describe the data produced by the method was motivated, primarily, by the testing procedure phase that will be described followingly. Two extremely common measures for evaluating Information Retrieval systems are precision and recall (YATES; NETO, 1999).

Precision, in Information Retrieval systems, is given by equation (6.1) and is the proportion of documents selected by the system that are actually relevant to the query fed to it. Precision is a measure of false positives resulting from the method used for selecting documents.

$$precision = \frac{|\{relevant\ documents\} \cap \{automatically\ selected\ documents\}|}{|\{automatically\ selected\ documents\}|} \tag{6.1}$$

Recall is a measure very much related to accuracy, essentially different in that it refers to the whole *corpus*, and not just by those documents retrieved by the system. It is given by equation (6.2), and can be seen as a measure of the proportion of false

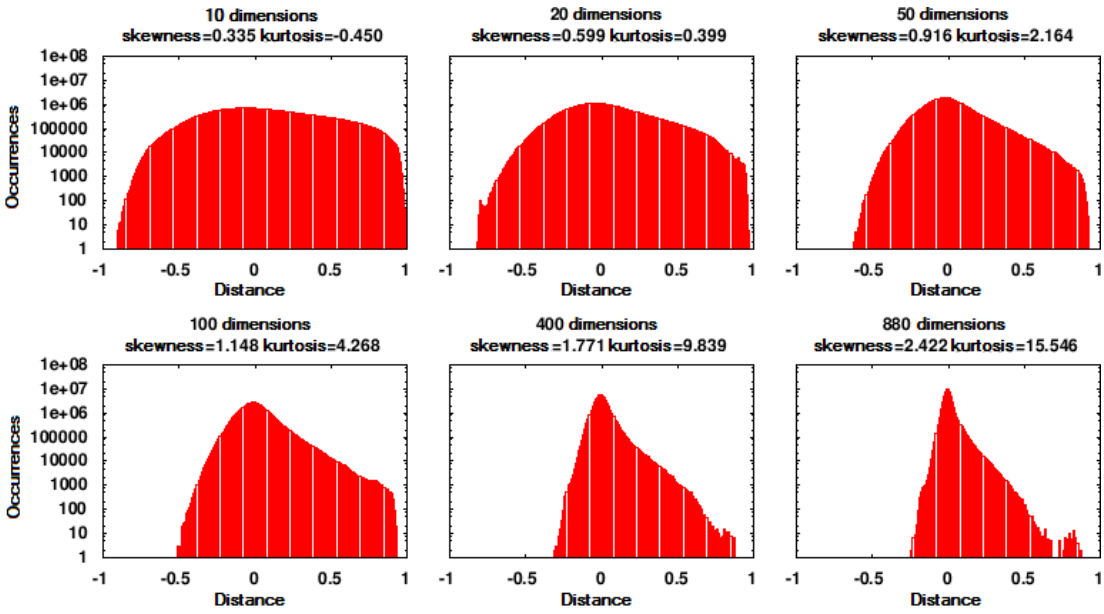

Figure 6.4: Distribution of normalized cosines of the angles between texts and subject classifications.

negatives obtained by the system.

$$recall = \frac{|\{relevant\ documents\} \cap \{automatically\ selected\ documents\}|}{|\{relevant\ documents\}|} \tag{6.2}$$

To determine what is the level of similarity found by the system that will distinguish a positive from a negative response as to if a classification can be applied to a document, one must choose numeric values.

As it was not possible to find a classical distribution able to describe the behavior of the LSA, experimentation was made with arbitrary thresholds for the automatic classification system. The system's performance was verified using thresholds of 1 to 10 times the standard deviation of the distribution for each index, in increments of one standard deviation. The progression of the average precision and recall rates for all LSA indices in relation to the chosen thresholds can be observed in figure 6.6. In order to have a single decision parameter, the geometric mean between these two measures was also placed on the graphs. Unlike the arithmetic mean, the geometric mean values the balance between its factors and not only their modules. As the two measures being used are complementary, it was felt that such balance is crucial. For the classification in subjects, the highest value obtained for the geometric average was 0.233 with a threshold of 2σ. For industries, the peak value was reached at 3σ, with a geometric mean of 0.143 (very close to the value obtained for 2σ, 0.141).

Just as was done for precision and recall in relation to the chosen threshold, data

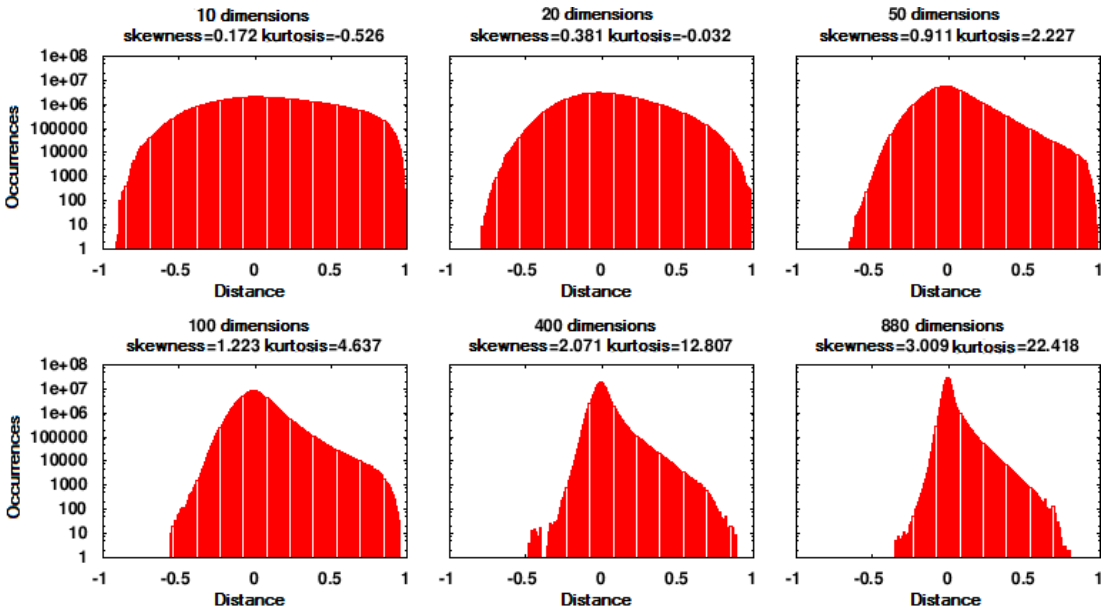

Figure 6.5: Normalized distributions of the cosines of the angles between texts and classifications in industries.

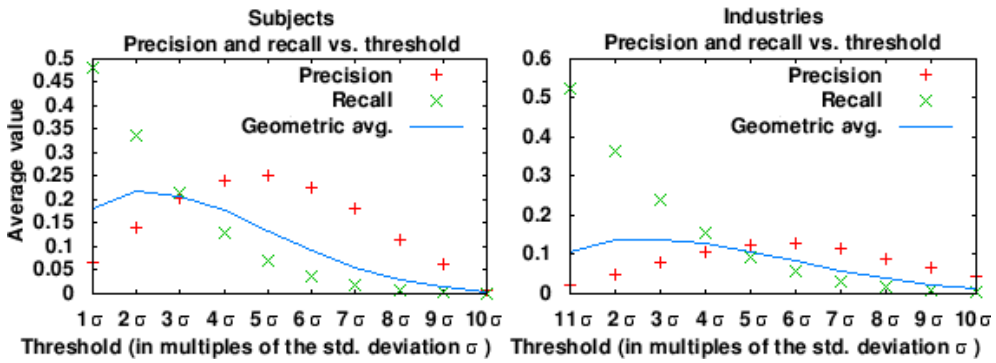

Figure 6.6: Graphs of average precision and recall versus the decision threshold of the system (in multiples of the standard deviation σ).

was collected for the generation of graphs of the progression of the average precision and recall while varying the number of index dimensions shown in figure 6.7.

The graph shown in figure 6.6 is particularly interesting. The steep performance drop observed exactly at the index of 820 dimensions might at first appear to be a spurious result. In fact, it shows a curious behavior with which previous works on LSA have come across: it is not rare for small changes in the number of LSA dimensions of an index to cause big changes in the system's ability to make relevant semantic associations. A graph presented in (LANDAUER; DUMAIS, 1997), reproduced in this paper in (LANDAUER; DUMAIS, 1997), illustrates a representative case.

In an attempt to obtain more detailed data on the variation in performance, the histograms of the best results of geometric mean between precision and recall for each number of dimensions were chosen to be analyzed. They are presented in figures 6.9

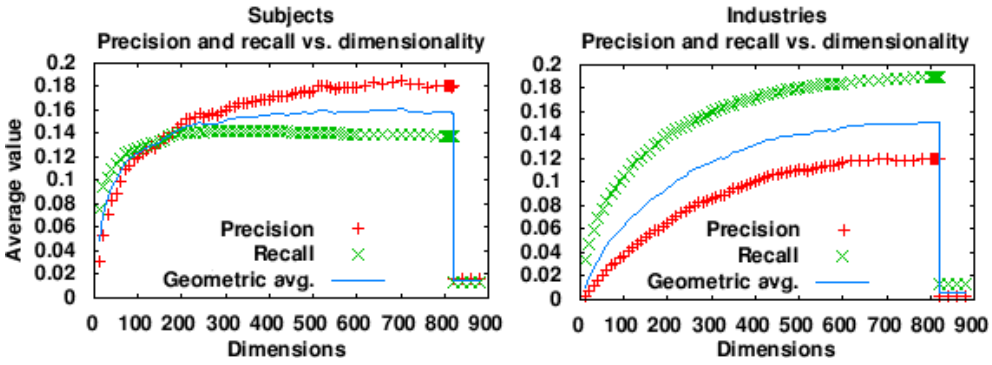

Figure 6.7: Graphs of the average precision and recall versus the number of dimensions of the indices.

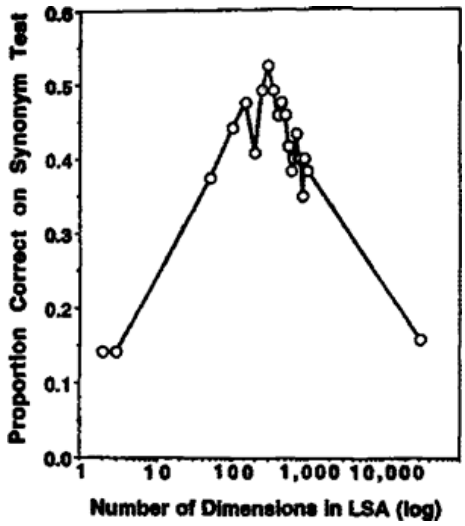

Figure 6.8: Graph of system performance of LSA developed in (LANDAUER; DUMAIS, 1997) (extract from (LANDAUER; DUMAIS, 1997)).

and 6.10

The first fact of note in the histograms of figures 6.9 and 6.10 is the number of dimensions at which top performance is reached. Latent semantic analysis is based on an algebraic method that generates a vector space in which, ideally, each dimension represents a semantic concept contained in the *corpus*, each made explicit by the co-occurrence of terms over the documents. In that context, to the extent that the concepts implicit in the classifications used in the tests can be said orthogonal, it is reasonable that there is correlation between the number of classifications and the number of dimensions of space that best describes these classifications. As mentioned earlier in this section, the number of subject classifications used was 121, and that of economic industries, 367. The best result in tests with subject classifications was obtained with an 80-dimensional index, and the best for industries with 490 dimensions. Results are consistent with predictions from the hypothesis that there is correlation between the vector space produced by LSA indexing and the underlying semantic organization of

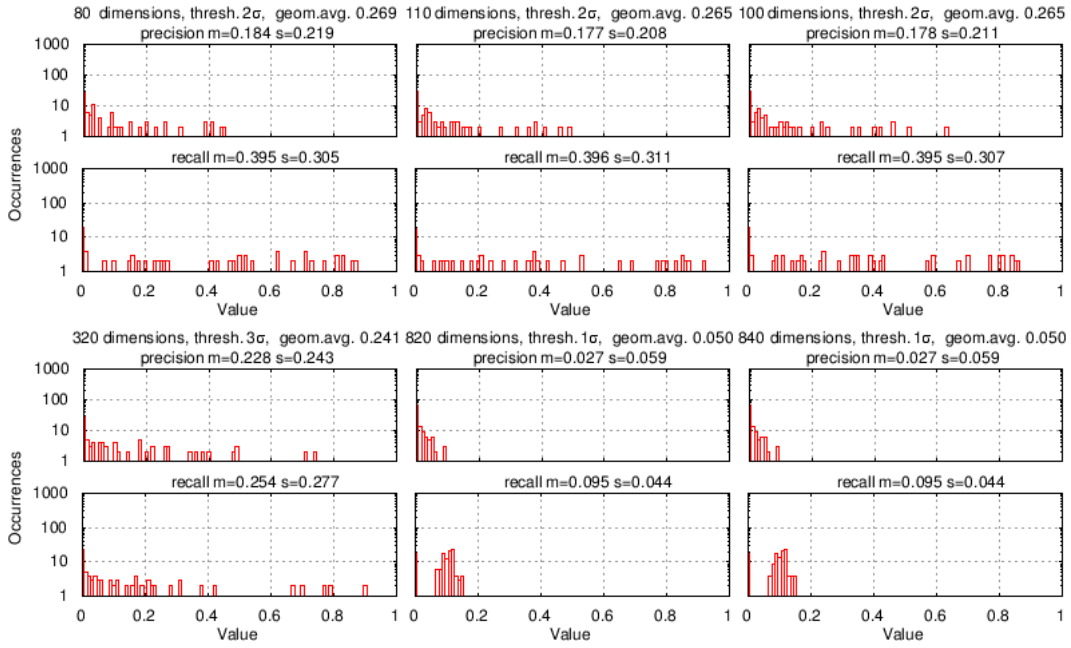

Figure 6.9: Histograms of a few representative best results in descending order, varying the choice of threshold levels for LSA. Classification by subject.

the *corpus*. Conversely, assuming that that hypothesis is correct, the result indicates that there is a superposition of 51.2% in the subject classifications defined by Reuters, while, following the same line, it would take more than 123 classifications of sectors of economy to describe all the possibilities covered by the documents.

The second interesting fact is observed in the histograms of figures 6.9 and 6.10 is the non-monotonic progression of the results, something that is more pronounced in the classifications by industry than in those by subject. In the subject classification tests, the best results occur with about 80 dimensions, falling gradually as one distances oneself from that value, but with a few spikes like the performance improvement observed when reducing the number of dimensions from 110 to 100. In the classification by industry, those spikes are even greater, with the 440-dimensional index being the best with less than 490 dimensions and that of 510, the third best among those with more than 500 dimensions.

The third and last result that was found worth of special note during the experiments is related to the influence of the automatic decision threshold of the system on its performance. The effect of the progression of that value seems at first to have a rather regular behavior, as shown in table 6.3. Between 30 and 300 dimensions, the best threshold experimented was twice the standard deviation of the distribution; from 310 to 819 dimensions, it was 3σ; and for the extremes, the lower threshold of 1σ.

In the industry classification tests, the best threshold for automatic sorting follows the pattern shown in table 6.4. The result is very interesting as it explicitates that the

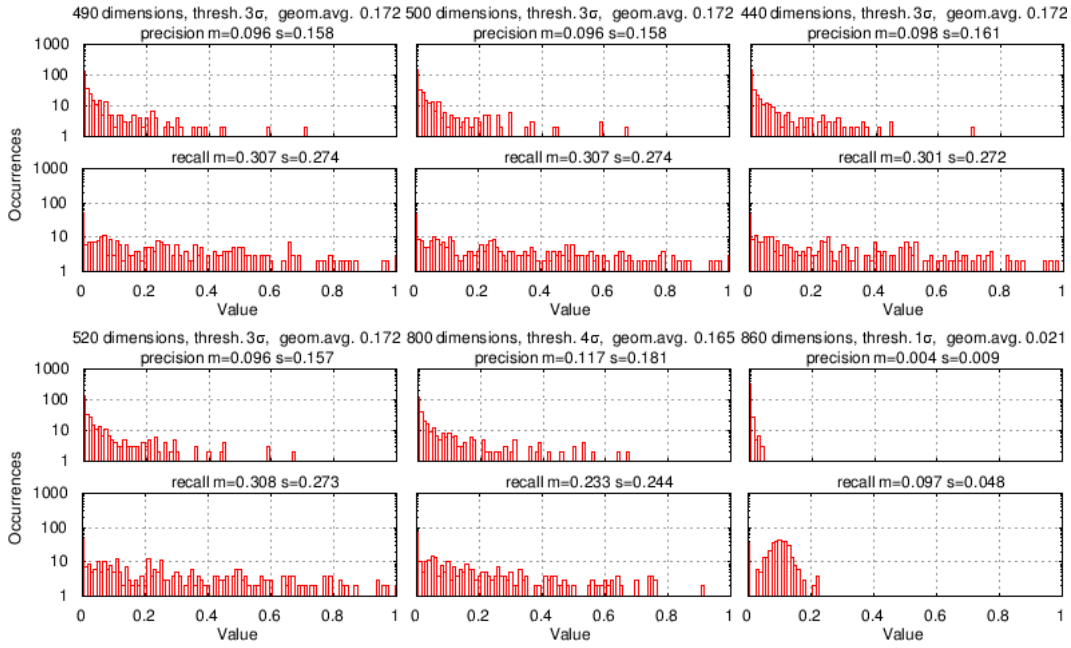

Figure 6.10: Histograms of a few representative best results in descending order, varying the choice of threshold levels for LSA. Classification by industry.

Threshold	Interval
1σ	$[10,20] \cup [820,880]$
2σ	$[30,300]$
3σ	$[310,819]$

Table 6.3: Best thresholds observed for subject ratings for each range of number of dimensions in LSA indices.

sudden performance drop shown in figure 6.7 is correlated with a slightly less abrupt decline in the value of the best threshold for choice. For indices with dimensionality between 760 and 818, the best threshold tested was that of 4σ. For the index of 819 dimensions, immediately prior to the sharp drop in performance, that figure falls to 3σ. In tests with subsequent indexes, with 820 to 880 dimensions, the optimal choice threshold drops to 1σ, and stabilizes at that lower level.

Threshold	Interval
1σ	$[10,40] \cup [820,880]$
2σ	$[50,230]$
3σ	$[240,740] \cup 819$
4σ	$[760,818]$

Table 6.4: Best thresholds observed for ratings by industry sector for each range of number of dimensions in LSA indices.

The results presented corroborate with the theory that the drop in performance observed in figure 6.7 is caused by changes in the distribution of distances assigned by LSA indices in relation to the classifications provided by Reuters as the dimensionality

of the vector space is modified. The reason for such a steep drop remains unclear, but it seems likely that tests with smoother changes to the automatic choice threshold might show that the performance decrease is not as sharp, and that the choice of the standard deviation as reference for the threshold might have been inadequate.

6.3.2 A few handpicked query results

A very interesting result, representative of the capacity of association of terms and documents using LSA, is obtained in the search for the name "Bush" in the document index. As mentioned earlier in this chapter, the *corpus* indexed for the tests of the current work consists of newspaper articles published in English by Reuters between August 1996 and March 1997. At that time, Bill Clinton was the president of the United States of America. He had won the 1992 presidential election competing with George H. W. Bush, and that of 1996 against Bob Dole. The next presidential election, when George W. Bush became president in his place, would only occur years later, in the year 2000. Still, for the search term "Bush" (notwithstanding the botanical meaning of the word), the first result is a text about the choice of the moderator of the debate between presidential candidates Bill Clinton and Bob Dole. In that 1447-characters long text, the term "Bush" does not occur even once. It is not possible to verify if the reason for this association is then ex-president George H. W. Bush and the then governor of Texas, or the future U.S. President George W. Bush.

Other curious results following that same line of politics-related subjects in the U.S. in the 1996-1997 period are the calculation of the distance between 'Bush" and "Republican", 0.38; "Bush" and "Democrat", 0.05, and, leaving aside the political context, between the terms "Madonna" and "Evita" (a movie in which the artist starred in 1996) the cosine is 0.90. However, the associations made by the algorithm are not always so anecdotal. One must take into account the fact that the period covered by the documents of the *corpus* is very short and that newspaper articles tend to be very concise and focused on the context of the moment. For some reason, the cosine between "Clinton" and "Democrat" is much less than between the then-president's surname and the name of the opposing party, "Republican". Perhaps that simply results from the number of time the main competing party in the U.S. was mentioned during that election year, but the reason might be something much more complex.

Part III

Final considerations

7 Contributions

For the current study, a version of the Solomonoff's Predictor restricted to producing hypotheses belonging to the set of Regular Languages was implemented, as well as an indexing and searching system based on Latent Semantic Analysis. Those two systems were tested in a series of situations, generating new data and interpretations. The issue of semantic analysis of Natural Language by machines remains open, but the results of this study contribute to understanding both the class of statistical algorithms as the set of analytical approaches to the problem.

In the sections that follow the results obtained in the course of this work are listed, and a brief discussion of what is reported in part II of this dissertation is presented.

7.1 Solomonoff's Predictor

As already mentioned in the introduction of chapter 5, there is no precedent in the literature for the implementation of Solomonoff's Predictor as was done in this work. That alone is not a quality and brings the difficulty of not being able of comparing results. Those that were presented in subsection 5.2.3 of this work, during the tests of divisibility of integers, have no published precedent. In an initial analysis, the device's observed behavior could even be interpreted as a serious defect. However, from a pragmatic point of view, one might arrive to different conclusions.

The preferential hypothesis proposed by predictor tests in subsection 5.2.3 is extremely accurate. It correctly predicts the outcome of the generator used in all the endless strings it produces, but also includes a single incorrect one. It took more than two centuries for generations of scientists to correct the Newtonian equation that describes the kinetic energy of a body with a term that, in routine situations, is around $10^{-8}J$. An error as the one made by the predictor device presented in this paper, literally infinitesimal, would probably never be noticed by people who could only observe the events produced by the original generator.

Regardless of stance on the convergence rate of the prototype of the simplified

Solomonoff's Predictor that was built for this work, the answers given by it appeal to the sense of the researcher. Those are clearly reasonable conjectures and their evolution follows the logical path that one might expect from an intelligent being.

Solomonoff wrote several works on the incomputability of his predictor (incomputability that is solved in the current work with the restriction made to the universe of Regular Languages) asserting that it is a desirable feature of the system, not a defect[1] (SOLOMONOFF, 2003a). The argumentation is that a comprehensive model of learning must necessarily be incomputable. The fact is that in a real environment, incomputability – and even exponential computational complexity in relation to input – greatly limits the applicability of a system. Solomonoff's proposal to mitigate the problem – interrupting the incremental processing of the predictor and accepting a suboptimal response whenever the delay is excessive – is not sufficient even for relatively simple problems in today's physical computers.

Many of the Artificial Intelligence methods that are used today rely on a device very natural for human beings: measures of utility. Those responses that can be shown to be more useful are prioritized in relation to those that are less useful, even when the less useful ones are more likely. A good doctor does not diagnose a patient as having an incurable disease before checking all less common curable diseases. The answer that a disease is incurable is marginally useful, while any diagnosis that may lead to treatment is interesting.

At this point this is just a conjecture, but it is possible that Solomonoff's Predictor could have a much improved practical performance if a measure of usefulness was attached to each event. In real situations, it isn't enough to have the best response that could be achieved in terms of balance between complexity and ability to explain the universe.

7.2 Representing semantics using LSA

In previous works on Latent Semantic Analysis of (DEERWESTER et al., 1990; LANDAUER; DUMAIS, 1997; BERRY; DUMAIS; O'BRIEN, 1995; PAPADIMITRIOU et al., 2000; ESULI; SEBASTIANI, 2005) as well as in the current one, one can observe that the vector representation given to Natural Language terms by the LSA method is related to the way humans interpret semantic relations. Applying such method to Information Retrieval systems is not in the scope of this work, but its performance

[1] Solomonoff's exact words are: "While these features are all very beautiful, there seemed at first to be a quite serious problem – that universal probability was incomputable. Surprisingly enough, this turned out to be not a *Bug* but a *Feature*!" (SOLOMONOFF, 2003a, p. 3)

in responding to requests made by humans is a parameter for the verification of its operation. However, the *corpus* used in the tests presented over the current study is substantially larger and more diverse than those used in such classic works on LSA and in such situation some interesting features of the method came to light.

An interesting fact that was observed in the present tests is that, unlike what is commonly seen in the literature for other methods, in tests made in this work precision and recall improved together in a large number of instances. That is strong indication that the performance increase observed with changes in the LSA system's choice threshold and the number of dimensions of vector space to represent the *corpus* is not casual. The system presented an evolution oriented in the sense of aligning its results with the expectations of the people who made the manual classification of texts.

The performance shown in tests using LSA classification in this study was considerably worse than what can be found in works like citelandauer:97 and (SCHONE; JURAFSKY, 2000). In those works, there are measures of precision and recall above 80%, while in this work at no time more than 23% precision and 40% recall were obtained. That difference, however, is largely dependent of the choice of *corpus*. The TOEFL Test of English as a Foreign Language used in (LANDAUER; DUMAIS, 1997) and the CELEX *corpus* used in (SCHONE; JURAFSKY, 2000), organized and annotated by linguists, are orders of magnitude smaller than the Reuters RCV1 *corpus* used for this work. The data set used in this work was built upon the work of hundreds of journalists whose primary commitment was to provide information that interested the reader. If on one hand the narrow focus of each text and common terminology consistency reporters are advantages to tests as those made in this work, on the other hand, the inevitable lack of consistency in the use of classifications in the *corpus* certainly negatively affected the results.

It is important to remember that the goal of this work, regarding LSA, is not coming up with new theories or advance knowledge about the characteristics of the method. The motivation for the functional testing of Latent Semantic Analysis made in this work was to verify the validity of its application as a module for the pre-processing of texts in Natural Language. In that context, what has been noted is that the characteristics of the method are adequate and, more importantly, that its performance parameters and processing time are extremely attractive to the intended application.

8 Conclusion

In this work, a prototype of the Predictor Solomonoff (SOLOMONOFF, 1964) restricted to the production of hypotheses belonging to the set of Regular Languages has been implemented. Tests were made with the program and it was determined that its computational complexity and cost of each of the tasks that the program must execute result in an excessively high processing time for most practical applications. Some interesting performance characteristics were observed in it, but, most of all, its efficiency was verified in terms of the number of computational steps to provide good hypotheses about an unknown generator of a sequence of events.

In a second phase an Information Retrieval system was implemented and tested, a system for indexing and searching text *corpora* using the Latent Semantic Analysis method proposed in (DEERWESTER et al., 1990). That system was exercised over the course of several days to build 81 indices of different dimensionalities, producing a total of 71.5GB of data. Those indices were then used to measure the cosine of the angle between the vectors of the classifications assigned manually by Reuters' journalists and the documents on the first disk in the RCV1 *corpus*. From these measurements, graphs were consolidated for the analysis of various features of the LSA method. The final conclusion from the tests is that the system has the potential to work as proposed at the beginning of this work, that is, as a module for the pre-processing of texts in Natural Language within a system of semantic analysis.

The results presented in chapter 6 indicate that the representation resulting from LSA indexing can be very advantageous for a mechanism that consumes texts in Natural Language as strings of events. In the vector space resulting from the application of the method, at least part of the semantic relations between words is exposed on their morphology. The computational cost involved in indexing a large *corpus* is relatively low and even if the contents of such *corpus* does not address explicitly every concept involved in its content, if the number of dimensions of vector space created is not too low for the recovery of the original terms (but not documents, as that would defeat the application of the method), there can be no loss of content. Everything points to likely gains in processing texts in natural language using Solomonoff's Predictor.

The computational complexity of Solomonoff's Predictor, despite all the qualities of the model, is still too difficult an obstacle to transpose. In this work it wasn't possible to experiment with the implemented predictor in the really interesting cases. Even the processing of a single average-length word from the English language – for instance, the word "world" – would take too long. The tests presented in chapter 5 were performed using an alphabet of two symbols and a string with 11 symbols was enough to consume just under an hour and a half of processing time. Considering that the device tests, in a case where there are no obvious regularities, $|\Sigma|^{input\_length}$ hypotheses (with $|\Sigma|$ being the number of symbols of the alphabet and $input\_length$ the length of the program's input), the average time consumed by hypothesis is around $2.5s$. Ignoring capitalization, the alphabet of English currently has 26 symbols. Thus, a prediction greatly simplified the processing of the word "world" would require around 275 days using computational resources equivalent to those available in the current work.

Despite the excellent intermediate results, as expected, the computational complexity of Solomonoff's Predictor prevented the integration of the modules. Other techniques, such as heuristic proposed in section 7.1, or significant advances in processing speed of computer systems will be necessary for a device such as the one implemented can be used on the scale needed for Natural Language processing.

The results of the current work will remain accessible on the Internet at http://iuri.chaer.org/master-thesis/. The source code of the software used will also be made available at the same address for free consultation and future developments of this work after acceptance for publication. It is expected that in this way the contributions of this research are strengthened.

9 Future work

At the end of this report, there seems more to be done than when it was began. The most straightforward continuation to it is to find ways to overcome or work around the obstacle represented by the high computational complexity involved in the execution of Solomonoff's Predictor, so as to enable its application to Natural Language processing. The independent results from the two modules built for the proposed system are very encouraging. Their integration will no doubt require a giant leap, but everything suggests promising results .

The proposal from section 7.1, of applying a usefulness measure to guide the suboptimal responses of Solomonoff's Predictor, is one possible path to apply the predictor to real problems of larger size. It is quite a difficult proposal to generalize because it involves a concept as fluid as "useful" and that would probably hurt the context independence of the Solomonoff's model. Still, it is possible that some balance between generality and complexity might result in a more easily applicable method.

Regarding the use of Latent Semantic Analysis with the goal of reducing the convergence time of Natural Language text processing by Solomonoff's Predictor, there is an incompatibility problem of representation that has not been discussed thus far, but that will become important once methods are found that can mitigate the problem of the predictor's computational complexity: the outcome of the application of LSA is a continuous vector space. The usual representation for such kind of data currently adopted in discrete computational systems is too costly in terms of space, which has a direct impact on processing time by the predictor. An assessment of the impact of the reduced accuracy of the representation of continuous values resulting from the LSA in its performance would be very valuable.

Finally, in a line that appears to be almost as difficult to implement as promising, it would be extremely interesting that a device implementing the full version of Solomonoff's Predictor were implemented, one capable of processing hypotheses that could only be described using the full power of Level 0 Languages. In (ROCHA, 2000), the application of Adaptive Formalism for that type of task is proposed. In (CHAER; ROCHA, 2009), a method of semantic search within such formalism is pro-

posed, something directly applicable to a problem-solver able to sort your entries in different groups of problems, as proposed in (SOLOMONOFF, 1986). In addition, it would probably be necessary to add the execution time of programs to their Kolmogorov complexity, following the proposal made by Schmidhuber in his Optimal Ordered Problem Solver (SCHMIDHUBER, 2004).

Part IV

References

References

ALPAYDIN, E. *Introduction To Machine Learning*. Cambridge, MA, USA: MIT Press, 2004.

APHASIA. Bethesda, Maryland, USA: National Instutute of Health (NIH), 2008. Disponível em: <http://www.nidcd.nih.gov/health/voice/aphasia.asp>. Acesso em: Maio de 2008.

ASHER, R. E. (Ed.). *The Encyclopedia of Language and Linguistics*. 2. ed. Amsterdã, Holanda: Elsevier, 2005.

BARTHES, R. *O Neutro*. São Paulo, SP, Brasil: Martins Fontes, 2003.

BERRY, M. W. Large-scale sparse singular value computations. *International Journal of Supercomputer Applications*, MIT Press, Cambridge, MA, USA, v. 6, n. 1, p. 13–49, 1992.

BERRY, M. W.; DUMAIS, S. T.; O'BRIEN, G. W. Using linear algebra for intelligent information retrieval. *SIAM review*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, v. 37, n. 4, p. 573–595, 1995.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *The Journal of Machine Learning Research*, MIT Press, Cambridge, MA, USA, v. 3, p. 993–1022, 2003.

BRACHMAN, R.; FIKES, R.; LEVESQUE, H. Krypton: A Functional Approach to Knowledge Representation. *Computer*, IEEE Computer Society, Washington, DC, USA, v. 16, n. 10, p. 67–73, 1983.

BRACHMAN, R. J.; LEVESQUE, H. J. *Readings in Knowledge Representation*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1985.

CARAMAZZA, A.; ZURIF, E. B. Dissociation of algorithmic and heuristic processes in language comprehension: Evidence from aphasia. *Brain and Language*, Elsevier, Amsterdã, Holanda, v. 3, n. 4, p. 572–582, 1976.

CHAER, I.; ROCHA, R. L. A. Um estudo sobre a Ação Elementar de Consulta no Formalismo Adaptativo. In: *Memórias do WTA 2009 – Terceiro Workshop de Tecnologia Adaptativa*. São Paulo, SP, Brasil: Escola Politécnica da USP, 2009. p. 129–134. ISBN 978-85-86686-51-1.

CHAITIN, G. J. Algorithmic information theory. Cambridge University Press, Cambridge, MA, USA, 1997.

CHOMSKY, N. *Aspects of the Theory of Syntax*. Cambridge, MA, USA: MIT Press, 1965.

CHOMSKY, N. *Knowledge of Language: Its Nature, Origin, and Use*. Westport, CT, USA: Praeger/Greenwood, 1986.

CRAIG, E. (Ed.). *Routledge Encyclopedia of Philosophy*. London, UK: Routledge, 1998.

DAMÁSIO, A. O. *O Erro de Descartes: emoção, razão eo cérebro humano*. Trad. Dora Vicente e Georgina Segurado. São Paulo, SP, Brasil: Companhia das Letras, 1999.

DAVIS, R.; SHROBE, H.; SZOLOVITS, P. What Is a Knowledge Representation? *AI Magazine*, AAAI, Menlo Park, CA, USA, v. 14, n. 1, p. 17–33, 1993.

DEERWESTER, S. et al. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, New York, NY, USA, v. 41, n. 6, p. 391–407, 1990.

ENCYCLOPÆDIA Britannica Online. Chicago, IL, USA: Encyclopædia Britannica, Inc., 2010. Disponível em: <http://www.britannica.com/EBchecked/topic/424706-/Ockhams-razor>. Acesso em: Janeiro de 2010.

ESULI, A.; SEBASTIANI, F. Determining the semantic orientation of terms through gloss classification. In: ACM. *Proceedings of the 14th ACM international conference on Information and knowledge management*. New York, NY, USA: ACM Press, 2005. p. 624.

FALK, J. S. *Saussure and American linguistics. Cambridge companion to Saussure, ed. by Carol Sanders*. Cambrige, UK: Cambridge University Press, 2003.

FITTING, M. *First-Order Logic and Automated Theorem Proving*. Berlin/Heidelberg, Alemanha: Springer, 1996.

FODOR, J. A. *The modularity of mind*. Cambridge, MA, USA: MIT Press, 1983.

GINZBURG, C.; CAROTTI, F. *Mitos, emblemas, sinais: morfologia e história*. São Paulo, SP, Brasil: Companhia das Letras, 1990.

GLIOZZO, A.; STRAPPARAVA, C. Domain kernels for text categorization. In: ACL. *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL)*. New Brunswick, NJ, USA: ACL, 2005. p. 56–63.

GOLUB, G.; KAHAN, W. Calculating the Singular Values and Pseudo-Inverse of a Matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*, Society for Industrial and Applied Mathematics, v. 2, n. 2, p. 205–224, 1965.

HAUSER, M. D.; CHOMSKY, N.; FITCH, W. The Faculty of Language: What Is It, Who Has It, and How Did It Evolve? *Science*, AAAS, Washington, DC, USA, v. 298, n. 5598, p. 1569–1579, 2002.

HAY, N. Solomonoff-lite evaluator. 2007. Disponível em: <http://www.singinst-.org/blog/2007/07/09/solomonoff-lite-evaluator/>. Acesso em: Outubro de 2009.

HEBERT, S. et al. Revisiting the dissociation between singing and speaking in expressive aphasia. *Brain*, Oxford University Press, Oxford, UK, v. 126, n. 8, p. 1838, 2003.

HENDLER, J.; BERNERS-LEE, T. From the Semantic Web to social machines: A research challenge for AI on the World Wide Web. *Artificial Intelligence*, Elsevier, Amsterdã, Holanda, n. 174, p. 156–161, 2010.

HOFMANN, T. Probabilistic latent semantic indexing. In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. New York, NY, USA: ACM Press, 1999. p. 50–57.

HOPCROFT, J. E. An n log n algorithm for minimizing the states in a finite automaton. Stanford University, Stanford, CA, USA, 1971.

HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Introduction to automata theory, languages, and computation*. 2. ed. Boston, MA, USA: Addison-Wesley, 2000.

HUTTER, M. *Universal artificial intelligence: Sequential decisions based on algorithmic probability*. Berlin/Heidelberg, Alemanha: Springer, 2005.

INFORMATION overload causes stress. *Reuters Magazine*, Reuters, London, UK, 1997. Disponível em: <http://library.humboldt.edu/˜ccm/fingertips/ioverloadstats-.html>. Acesso em: Janeiro de 2009.

IWANSKA, L.; SHAPIRO, S. C. *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language*. Cambridge, MA, USA: MIT Press, 2000.

JOANES, D. N.; GILL, C. A. Comparing measures of sample skewness and kurtosis. *The Statistician*, Blackwell Publishers, Hoboken, NJ, USA, v. 47, n. 1, p. 183–189, 1998.

JURAFSKY, D.; MARTIN, J. H. *Speech And Language Processing*. Englewood Cliffs, NJ, USA: Prentice Hall, 2008.

LANDAUER, T. K.; DUMAIS, S. T. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge. *Psychological Review*, APA – American Psychological Association, New York, NY, USA, v. 104, p. 211–240, 1997.

LENAT, D. B. et al. Cyc: toward programs with common sense. *Communications of the ACM*, ACM Press, New York, NY, USA, v. 33, n. 8, p. 30–49, 1990.

LI, M.; VITÁNYI, P. M. B. Inductive reasoning and kolmogorov complexity. *Proceedings of the Fourth Annual Conference on Structure in Complexity Theory*, IEEE Computer Society, Washington, DC, USA, p. 165–185, 1989.

LI, M.; VITÁNYI, P. M. B. *An Introduction to Kolmogorov Complexity and Its Applications*. Berlin/Heidelberg, Alemanha: Springer, 1997.

LOCKE, J. *Ensaio acerca do entendimento humano. (Col. Os Pensadores)*. São Paulo, SP, Brasil: Ed. Abril, 1973.

MAGNINI, B.; CAVAGLIA, G. Integrating subject field codes into wordnet. In: *Proceedings of LREC-2000, Second International Conference on Language Resources and Evaluation*. Atenas, Grécia: LREC-2000, 2000. p. 1413–1418.

MCGUINNESS, D. L.; HARMELEN, F. van et al. OWL Web Ontology Language Overview. *W3C Recommendation*, W3C, Cambridge, MA, USA, v. 10, p. 2004–03, 2004.

MINSKY, M. L. *Computation: finite and infinite machines*. Englewood Cliffs, NJ, USA: Prentice Hall, 1967.

MITCHELL, M. *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: Bradford Books, 1996.

MONTAGUE, R. The proper treatment of quantification in ordinary English. *Proceedings of the 1970 Stanford Workshop on Grammar and Semantics*, D. Reidel, Dordrecht, Holanda, v. 49, p. 221–242, 1973.

MYLOPOULOS, J. An overview of Knowledge Representation. In: *Proceedings of the workshop on Data abstraction, databases and conceptual modelling*. New York, NY, USA: ACM, 1981. v. 11, n. 2, p. 5–12.

NERO, H. S. D. *O sítio da mente pensamento, emoção e vontade no cérebro humano*. São Paulo, SP, Brasil: Collegium Cognitio, 2002.

NILSSON, N. J. Introduction to Machine Learning. Unpublished Draft – Department of Computer Science, Stanford University, Redwood City, CA, USA, 1996. Disponível em: <http://robotics.stanford.edu/people/nilsson/mlbook.html>. Acesso em: Janeiro de 2010.

PAPADIMITRIOU, C. H. et al. Latent semantic indexing: A probabilistic analysis. *Journal of Computer and System Sciences*, Elsevier, Amsterdã, Holanda, v. 61, n. 2, p. 217–235, 2000.

RISSANEN, J. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, Institute of Mathematical Statistics, Beachwood, OH, USA, v. 11, n. 2, p. 416–431, 1983.

ROCHA, R. L. *Um método de escolha automática de soluções usando tecnologia adaptativa*. Tese (Doutorado) — Escola Politécnica da USP, São Paulo, SP, Brasil, 2000.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence – A Modern Approach*. 2. ed. Englewood Cliffs, NJ, USA: Prentice Hall, 2003.

SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management: an International Journal*, Pergamon Press, Inc., Tarrytown, NY, USA, v. 24, n. 5, p. 513–523, 1988.

SAUSSURE, F. *Course in General Linguistics*. Trad. Roy Harris. La Salle, IL, USA: Open Court, 1983.

SCHMIDHUBER, J. Optimal Ordered Problem Solver. *Machine Learning*, Kluwer Academic Publishers, Amsterdã, Holanda, n. 54, p. 211–254, 2004.

SCHONE, P.; JURAFSKY, D. Knowledge-free induction of morphology using latent semantic analysis. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning*. Morristown, NJ, USA, 2000. v. 7, p. 67–72.

SOLOMONOFF, R. J. A formal theory of inductive inference. parts I and II. *Information and Control*, Academic Press Inc., Boston, MA, USA, v. 7, n. 2, p. 224–254, 1964. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. Inductive inference theory - A unified approach to problems in pattern recognition and artificial intelligence. *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, Tbilisi, Georgia, U.S.S.R., p. 274–280, 1975. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. Complexity-based induction systems: Comparisons and convergence theorems. *Information Theory, IEEE Transactions on*, IEEE, New York, NY, USA, v. 24, n. 4, p. 422–432, 1978. Disponível em: <http://world.std.com/˜rjs-/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. The application of algorithmic probability to problems in artificial intelligence. *Proceedings of the Second Annual Conference on Uncertainty in Artificial Intelligence*, Elsevier, Amsterdã, Holanda, p. 473–491, 1986. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. A system for incremental learning based on algorithmic probability. *Proceedings of the Sixth Israeli Conference on Artificial Intelligence, Computer Vision and Pattern Recognition*, Tel Aviv, Israel, p. 515–527, 1989. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. Two Kinds of Probabilistic Induction. *The Computer Journal*, British Computer Society, London, UK, v. 42, n. 4, p. 256–259, 1999. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. The Universal Distribution and Machine Learning. *The Computer Journal*, British Computer Society, London, UK, v. 46, n. 6, p. 598–601, 2003. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. Three Kinds of Probabilistic Induction: Universal Distributions and Convergence Theorems. *The Computer Journal*, British Computer Society, London, UK, 2003. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

SOLOMONOFF, R. J. *Lecture 1: Algorithmic Probability*. 2005. Disponível em: <http://world.std.com/˜rjs/pubs.html>. Acesso em: Janeiro de 2009.

TURNEY, P. D. Measuring semantic similarity by latent relational analysis. *International Joint Conference on Artificial Intelligence*, Lawrence Erlbaum Associates Ltd., Philadelphia, PA, USA, v. 19, p. 1136–1141, 2005.

WIDDOWS, D. Semantic vector products: Some initial investigations. In: *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*. UK: College Publications, 2008.

WILLIS, D. G. Computational Complexity and Probability Constructions. *Journal of the ACM (JACM)*, ACM Press, New York, NY, USA, v. 17, n. 2, p. 241–259, 1970.

WURMAN, R. S. *Information Anxiety*. New York, NY, USA: Doubleday, 1989.

WURMAN, R. S. *Information anxiety 2*. Indianapolis, IN, USA: Hayden/Que, 2001.

YATES, R. B.; NETO, B. R. *Modern Information Retrieval*. Boston, MA, USA: Addison-Wesley, 1999.