THEORETICAL RESULTS ON A WEIGHTLESS NEURAL CLASSIFIER AND APPLICATION TO COMPUTATIONAL LINGUISTICS

Hugo Cesar de Castro Carneiro

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Felipe Maia Galvão França
Priscila Machado Vieira Lima
Carlos Eduardo Pedreira

Rio de Janeiro
Junho de 2017

THEORETICAL RESULTS ON A WEIGHTLESS NEURAL CLASSIFIER AND APPLICATION TO COMPUTATIONAL LINGUISTICS

Hugo Cesar de Castro Carneiro

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Felipe Maia Galvão França, Ph.D.

Prof. Priscila Machado Vieira Lima, Ph.D.

Prof. Carlos Eduardo Pedreira, Ph.D.

Prof. Valmir Carneiro Barbosa, Ph.D.

Prof. Daniel Sadoc Menasché, Ph.D.

Prof. Aluizio Fausto Ribeiro Araújo, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
JUNHO DE 2017

Carneiro, Hugo Cesar de Castro

Theoretical Results on a Weightless Neural Classifier and Application to Computational Linguistics/Hugo Cesar de Castro Carneiro. – Rio de Janeiro: UFRJ/COPPE, 2017.

XVIII, 97 p.: il.; 29, 7cm.

Orientadores: Felipe Maia Galvão França
Priscila Machado Vieira Lima
Carlos Eduardo Pedreira

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2017.

Bibliografia: p. 73 – 84.

1. WiSARD. 2. Bleaching technique. 3. n-tuple classifier. 4. Part-of-speech tagging. 5. Zipf's law. 6. Statistical learning theory. 7. Vapnik-Chervonenkis dimension. I. França, Felipe Maia Galvão *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

To my wife.
You are my inspiration.

Acknowledgments

I would like to thank my wife, Adriana Mussap Cabral, for being very supportive during all the time I spent in my D.Sc. research. She proved very patient and helpful to me, by motivating me to keep on my research and aiding me in focusing on my tasks, leading this work to its completion.

I would also like to thank my mother for teaching me how to have discipline and responsibility, so I could be successful in my academic and professional career. Thanks to my father for being the first person who led me aspire for the academic career. My grandmother and my aunt also deserve to be remembered, for always being with me in many important moments of my life.

I am very thankful to my advisors, Prof. Felipe Maia Galvão França and Prof. Priscila Machado Vieira Lima, for offering me the opportunity to carry out this research. Their insightful ideas and thought-provoking questions were key contributions to the production of this thesis. Thanks to Prof. Carlos Eduardo Pedreira for accepting to become my third advisor, and whose knowledge on statistical learning theory was vital for all theoretical research done in this publication.

I would like to thank Profs. Valmir Carneiro Barbosa, Daniel Sadoc Menasché and Aluizio Fausto Ribeiro Araújo for their careful reviews and the relevant comments they provided. Thanks to my lab coworkers for all instigating brainstorms and all the good humor, which made my life as a D.Sc. student much more pleasurable. And thanks to Nicholas P. Bradshaw, whose pioneering research on the VC dimension of the n-tuple classifier gave me pretty good insights, helping me achieve the results presented in this thesis.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

RESULTADOS TEÓRICOS SOBRE UM CLASSIFICADOR NEURAL SEM PESO E APLICAÇÃO À LINGUÍSTICA COMPUTACIONAL

Hugo Cesar de Castro Carneiro

Junho/2017

Orientadores: Felipe Maia Galvão França
Priscila Machado Vieira Lima
Carlos Eduardo Pedreira

Programa: Engenharia de Sistemas e Computação

WiSARD é um classificador n-upla, historicamente usado em tarefas de reconhecimento de padrões em imagens em preto e branco. Infelizmente, não era comum que este fosse usado em outras tarefas, devido à sua incapacidade de arcar com grandes volumes de dados por ser sensível ao conteúdo aprendido. Recentemente, a técnica de *bleaching* foi concebida como uma melhoria à arquitetura do classificador n-upla, como um meio de coibir a sensibilidade da WiSARD. Desde então, houve um aumento na gama de aplicações construídas com este sistema de aprendizado.

Pelo uso frequente de *corpora* bastante grandes, a etiquetação gramatical multilíngue encaixa-se neste grupo de aplicações. Esta tese aprimora o mWANN-Tagger, um etiquetador gramatical sem peso proposto em 2012. Este texto mostra que a pesquisa em etiquetação multilíngue com WiSARD foi intensificada através do uso de linguística quantitativa e que uma configuração de parâmetros universal foi encontrada para o mWANN-Tagger. Análises e experimentos com as bases da Universal Dependencies (UD) mostram que o mWANN-Tagger tem potencial para superar os etiquetadores do estado da arte dada uma melhor representação de palavra.

Esta tese também almeja avaliar as vantagens do *bleaching* em relação ao modelo tradicional através do arcabouço teórico da teoria VC. As dimensões VC destes foram calculadas, atestando-se que um classificador n-upla, seja WiSARD ou com *bleaching*, que possua N memórias endereçadas por n-uplas binárias tem uma dimensão VC de exatamente $N\left(2^n - 1\right) + 1$. Um paralelo foi então estabelecido entre ambos os modelos, onde deduziu-se que a técnica de *bleaching* é uma melhoria ao método n-upla que não causa prejuízos à sua capacidade de aprendizado.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

THEORETICAL RESULTS ON A WEIGHTLESS NEURAL CLASSIFIER AND APPLICATION TO COMPUTATIONAL LINGUISTICS

Hugo Cesar de Castro Carneiro

June/2017

Advisors: Felipe Maia Galvão França
Priscila Machado Vieira Lima
Carlos Eduardo Pedreira

Department: Systems Engineering and Computer Science

WiSARD is an n-tuple classifier, historically employed in pattern recognition tasks involving black-and-white images, like recognition handwritten characters. Unfortunately, it was not commonly used in other tasks, for its inability to handle large loads of data, as it was sensitive to the learned content. Recently, the bleaching technique was conceived as an enhancement to the n-tuple classifier architecture as a means to curb WiSARD sensitiveness. Since then, there has been an increase in the range of applications built with this learning system.

Multilingual part-of-speech (POS) tagging is one of such applications, given its frequent use of large corpora. This thesis improves mWANN-Tagger, a multilingual weightless neural network POS-tagger proposed in 2012. The text herein presented shows that the research on multilingual POS-tagging with WiSARD was intensified through the use of quantitative linguistics and that a universal parameter configuration was found for mWANN-Tagger. Further analyses and experiments with Universal Dependencies (UD) treebanks show that mWANN-Tagger has potential to outperform state-of-the-art POS-taggers given a better word representation.

This thesis also aims to assess the advantages of bleaching towards the traditional model through the theoretical framework of VC theory. The VC dimensions of both architectures were calculated, attesting that an n-tuple classifier, WiSARD or bleaching alike, which has N memory nodes addressed by binary n-tuples has VC dimension of exactly $N\,(2^n - 1) + 1$. A parallel was then drawn between both models, where it was deduced that the bleaching technique is an enhancement to the n-tuple method that does little to no harm to its learning capacity.

Contents

List of Figures **x**

List of Tables **xii**

List of Symbols **xiv**

List of Abbreviations **xviii**

1 Introduction **1**

1.1 Motivation . . . 2
1.2 Related Works . . . 3
1.3 Goals and Contributions . . . 4
1.4 Thesis Structure . . . 6

2 The WiSARD Model **7**

2.1 The Traditional WiSARD n-tuple Classifier . . . 7
2.1.1 System Architecture . . . 8
2.1.2 Training Procedure . . . 8
2.1.3 Recognition Procedure . . . 9
2.1.4 Mathematical Formulation . . . 9
2.2 The Bleaching Technique . . . 11
2.2.1 Training Procedure . . . 11
2.2.2 Recognition Procedure . . . 12
2.2.3 Mathematical Formulation . . . 14

3 mWANN-Tagger **16**

3.1 Basic Concepts: Quantitative Linguistics . . . 16
3.2 Basic Concepts: Statistics of Word Distribution . . . 18
3.3 mWANN-Tagger Architecture . . . 21
3.3.1 Tagset . . . 23
3.3.2 mWANN-Tagger Design Parameters . . . 23
3.3.3 Input Construction . . . 26

3.4 Experiments . . . 27
3.4.1 The Zipfian Nature of Languages . . . 27
3.4.2 The Optimal Set of Parameters . . . 32
3.4.3 Comparison with the State of the Art . . . 38
3.5 Discussion . . . 42

4 VC Dimension of n-tuple Learning Models 44
4.1 Basic Concepts: Introduction to Learning Theory . . . 44
4.1.1 Risk Minimization . . . 45
4.1.2 Bounding the Risk . . . 45
4.2 Determination of the VC Dimension of WiSARD n-tuple Classifier . . . 48
4.2.1 Preliminaries . . . 49
4.2.2 The Lower Bound . . . 51
4.2.3 The Upper Bound . . . 57
4.3 Determination of the VC dimension of bleaching n-tuple classifier . . . 60
4.3.1 Preliminaries . . . 60
4.3.2 The Proof Itself . . . 62
4.4 Discussion . . . 64

5 Conclusion 68
5.1 Summary of the Thesis . . . 68
5.1.1 Chapter 1 . . . 68
5.1.2 Chapter 2 . . . 68
5.1.3 Chapter 3 . . . 69
5.1.4 Chapter 4 . . . 70
5.1.5 Chapter 5 . . . 71
5.2 Final Remarks . . . 71
5.3 Future Works . . . 72

Bibliography 73

A Publications related to the thesis 85

B Further publications as a D.Sc. student 89

List of Figures

2.1 Traditional WiSARD classifier 8
2.2 Bleaching classifier. 12

3.1 Word rank per word frequency graphs depicting the effect of α_1 on the steepness of the Zipf curve. 19
3.2 Word rank per word frequency graphs depicting the effect of penalizing parameter α_2 on the ZM curve. 20
3.3 Zipf curves for different languages. The curves are fitted according to the dots, which express the observed data. The horizontal and vertical axes represent a same word rank and frequency, respectively. 20
3.4 mWANN-Tagger modes 22
3.5 mWANN-Tagger procedure of encoding sentences into arrays of probabilities 24
3.6 Discretization of a relative frequency into a string of bits 26
3.7 Correlation between mWANN-Tagger and Zipf-Mandelbrot parameters of corpora of Table 3.3. 31
3.8 Sensitivity analysis for the tagger parameters. Graphs depict the accuracy variation caused by fluctuations on parameter values. 33
3.9 Sensitivity analysis for the new corpora. Graphs depict the accuracy variation caused by fluctuations on parameter values. 38

4.1 A linear separator shatters a set with three points 46
4.2 A linear separator does not shatter a set with four points 47
4.3 Proof flowchart for determination of the VC dimension of traditional WiSARD n-tuple classifier. 50
4.4 WiSARD memory position fillings 54
4.5 Classification of 01:00 54
4.6 Classification of 10:00 55
4.7 Classification of 00:01 55
4.8 Classification of 00:10 55
4.9 Classification of 00:00 55

4.10 Proof flowchart for determination of the VC dimension of min-max bleaching n-tuple classifier. 61

List of Tables

2.1 Ambiguous scenario, where an input pattern is not clearly recognized by any single discriminator. 13

3.1 Comparisons of properties of isolating and synthetic languages 18
3.2 Tagset of mWANN-Tagger 23
3.3 Original corpora 28
3.4 Zipf-Mandelbrot parameters of the original corpora 30
3.5 Empirically-obtained mWANN-Tagger parameter configurations for the corpora of Table 3.3. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure. 31
3.6 Cross-linguistic analysis. Rows represent the corpora being tested and columns the corresponding tagger configurations. Table cells show the accuracy (%) of a tagger in this particular scenario. Bolded cells represent the greatest accuracy achieved for a given corpus. 32
3.7 mWANN-Tagger parameter configurations of Table 3.5 and the derived universal configuration 34
3.8 Accuracies for the empirical and universal configurations. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure. 34
3.9 New corpora 35
3.10 Zipf-Mandelbrot parameters and accuracy for the new corpora. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure. 37

3.11 Comparison between part-of-speech accuracies (%) on UD treebanks. Columns mWANN – Suffix and mWANN – Ext. offer the accuracies of (suffix-based) mWANN-Tagger universal configuration and its extension, respectively. The remaining ones, labeled MM, RDR, YAP, bi-LSTM and MM, hold the results of MarMoT [1], RDRPOSTagger [2], YAP [3] and bi-LSTM [4]. Highlighted cells represent cases where no statistically significant difference was attested between the performances of mWANN-Tagger extension and MarMoT. 39
3.12 Comparison between part-of-speech accuracies (%) on UD treebanks. Columns mWANN – Suffix and mWANN – Ext. offer the accuracies of (suffix-based) mWANN-Tagger universal configuration and its extension, respectively. The remaining ones, labeled MM, RDR, YAP, bi-LSTM and MM, hold the results of MarMoT [1], RDRPOSTagger [2], YAP [3] and bi-LSTM [4]. Highlighted cells represent cases where no statistically significant difference was attested between the performances of mWANN-Tagger extension and MarMoT. (cont.) . . 40

4.1 Notation for traditional n-tuple classifier. 52
4.2 Dichotomies of $X_{2,2}$ 53
4.3 Notation for min-max bleaching n-tuple classifier. 62

List of Symbols

0^n	All-zero n-tuple, p. 51
1^n	All-one n-tuple, p. 51
$A(\cdot)$	Addressing function, p. 9
$B_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots)$	Bleaching N-node n-tuple classifier with memory matrices $\mathbf{M}_a$, $\mathbf{M}_b$, and others, p. 14
C	A set of classes, p. 9
D	Training dataset, p. 45
L	Amount of characters, p. 24
$L_{\max}$	Size of largest ending, p. 24
$L_{\min}$	Size of smallest ending, p. 24
L_{root}	Size of smallest root, p. 25
N	Number of memory nodes, p. 8
$P(\mathbf{x})$	Probability distribution, p. 45
$Q(w_i, t_j)$	Absolute frequency of word w_i tagged as t_j, p. 24
$R(\theta)$	Risk functional, p. 45
T	Number of tags, p. 23
U	Number of unique words, p. 25
$V(y, h(\mathbf{x}, \theta))$	Loss function, p. 45
$W_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots)$	WiSARD N-node n-tuple classifier with memory matrices $\mathbf{M}_a$, $\mathbf{M}_b$, and others, p. 10
X	Ordered set of input data points, p. 9

Z	Ordered set of feature matrices, p. 51
$\Delta_\Theta(X)$	Set of dichotomies realizable on X, p. 46
$\Pi_\Theta(\ell)$	Growth function of a learning machine $\mathcal{L}$ on samples of size ℓ, p. 46
Σ_c	Summation device of discriminator $\mathcal{D}_c$, p. 9
Θ	Set of parameters of $\mathcal{L}$, p. 45
α_1	Slope parameter of Zipf's law and Zipf-Mandelbrot distribution, p. 19
α_2	Offset parameter of Zipf-Mandelbrot distribution, p. 19
β	Bleaching threshold, p. 12
δ	Probability discretization degree, p. 25
ℓ	Amount of input data points, dataset length, p. 45
η	Tolerance level associated to the difference between the risk functional and its empirical variant, p. 47
$\hat{\alpha_1}$	Maximum likelihood estimator of α_1, p. 21
$\hat{\alpha_2}$	Maximum likelihood estimator of α_2, p. 21
$\lambda_{c,i,j}$	Element at i-th row and j-th column of $\mathbf{\Lambda}_c$, p. 63
$\mathbf{M}_c$	Memory matrix associated to class c, p. 10
$\mathbf{Z}_k$	A feature matrix, k-th element of Z, p. 9
$\mathbf{\Lambda}_c$	Modified memory matrix of class c, p. 63
$\mathbf{s}$	Bleaching-independent score vector, p. 11
$\mathbf{s}^{(\beta)}$	Score vector given threshold β, p. 15
$\mathbf{x}_k$	k-th element of X, p. 9
$\mathbf{x}_{k,1} : \ldots : \mathbf{x}_{k,N}$	Addressing representation of $\mathbf{x}_k$, p. 10
$\mathbf{x}_{k,i}$	i-th n-tuple of $\mathbf{x}_k$, which addresses $d_{c,i}$, p. 10
$\mathbf{y}$	Vector of classes, one for each element of Z, p. 51
$\mathbf{z}_{k,i}$	i-th row of $\mathbf{Z}_k$, p. 51

$\mathcal{B}_{N,n,C}$	Multiclass bleaching N-node n-tuple classifying model, p. 14
$\mathcal{B}_{N,n}$	Biclass min-max bleaching N-node n-tuple classifying model, p. 62
$\mathcal{D}_c$	Discriminator for class c, p. 8
$\mathcal{G}$	Generator, p. 45
$\mathcal{H}$	Set of functions implemented by $\mathcal{L}$, p. 45
$\mathcal{L}$	Learning machine, p. 45
$\mathcal{S}$	Supervisor, p. 45
$\mathcal{W}_{N,n,C}$	Multiclass WiSARD N-node n-tuple classifying model, p. 10
$\mathcal{W}_{N,n}$	Biclass WiSARD N-node n-tuple classifying model, p. 50
$\mathcal{X}$	input space, p. 45
$\mathcal{Y}$	Output space, p. 45
μ_k	Slack variable, p. 58
ν_k	Difference between scores $s_1(\mathbf{Z}_k)$ and $s_{-1}(\mathbf{Z}_k)$, p. 58
ω_c	Number of observations of a training dataset D that are associated to class c, p. 65
π_i	A prefix, p. 41
σ_i	A suffix or ending, p. 41
$\mathbf{w}$	A sentence or sequence of words, p. 41
θ	Parameter configuration of $\mathcal{L}$, p. 45
θ^*	Parameter configuration that minimizes the expected value of loss function $V(y, h(\mathbf{x}, \theta))$, p. 45
θ^*_ℓ	Parameter configuration that minimizes the empirical risk functional, p. 45
ε	A given error, p. 47
a	Number of words after, p. 25
b	Number of words before, p. 25

c	A generic class, p. 10
d_{VC}	VC dimension, p. 47
$d_{c,i}$	i-th memory node of $\mathcal{D}_c$, p. 11
f	Relative frequency, p. 23
$g(\mathbf{x};\theta)$	A function produced by learning machine $\mathcal{L}$, p. 46
$h(\mathbf{x};\theta)$	Function implemented by $\mathcal{L}$, p. 45
$m_{c,i,j}$	Element at i-th row and j-th column of $\mathbf{M}_c$, p. 10
n	Addressing tuple length, p. 8
p_i	Fixed probability value, p. 25
r	Word rank, p. 19
$s_c^{(\beta)}$	Score of $\mathcal{D}_c$ given threshold β, p. 15
s_c	Bleaching-independent score of $\mathcal{D}_c$, p. 11
t_j	A tag, p. 23
$v_{c,i}$	Value stored in the accessed position of i-th node of $\mathcal{D}_c$, p. 13
w_i	A word, p. 24
y_k	k-th element of $\mathbf{y}$, expected class of $\mathbf{Z}_k$, p. 51
$z_{k,i,j}$	Element at i-th row and j-th column of $\mathbf{Z}_k$, p. 11
$\{0,1\}^n$	Generic binary n-tuple, p. 51

List of Abbreviations

CRF	Conditional random field, p. 39
ELM	Extreme learning machine, p. 65
ERM	Empirical risk minimization, p. 45
IR	Information retrieval, p. 4
ML	Maximum likelihood, p. 21
MSEA	Mainland Southeast Asia, p. 36
NLP	Natural language processing, p. 4
OOV	Out of vocabulary, p. 2
POS	Part of speech, p. 1
RAM	Random access memory, p. 7
RDRPOSTagger	Ripple down rules part-of-speech tagger, p. 38
SVM	Support vector machine, p. 65
UD	Universal Dependencies, p. 38
VC	Vapnik-Chervonenkis, p. 3
WANN	Weightless artificial neural network, p. 1
WiSARD	Wilkie Stonham and Aleksander Recognition Device, p. 1
YAP	Yet another parser, p. 38
ZM	Zipf-Mandelbrot, p. 4
bi-LSTM	Bidirectional long short-term memory, p. 38
mWANN-Tagger	Multilingual weightless artificial neural network part-of-speech tagger, p. 1

Chapter 1

Introduction

The **Wi**lkie, **S**tonham and **A**leksander **R**ecognition **D**evice (**WiSARD**) [5, 6] is a versatile weightless artificial neural network (WANN). Its origins date back to the n-tuple classifier, which was initially proposed in [7] and then formally defined in [8]. WiSARD has a very modular architecture and trains unseen patterns in a single pass, making it an efficient learning machine. However, little attention was given to this model because it could not handle large loads of data. The bleaching technique [9], a recent enhancement developed for WiSARD, solved this issue. Applications built employing this technique indicated their ability to handle large loads of data [10–13]. When compared with state-of-the-art systems, these applications showed competitive results, sometimes outperforming them.

Part-of-speech (POS) tagging[1] is one kind of application that requires learning machines capable of handling a large load of data, given that some corpora (and treebanks) can have from tens of thousands to some million tokens to be learned. The **m**ultilingual **w**eightless **a**rtificial **n**eural **n**etwork part-of-speech **tagger** (**mWANN-Tagger**) [10–12] is a part-of-speech tagger which employs a bleaching n-tuple classifier to acquire knowledge on the tokens of a given textual base. This tagger had a group of parameters to determine the kind of knowledge it should get from a sentence in order to learn how to accurately tag others. However, fine-tuning a parameter configuration can be very time-consuming, and for a multilingual part-of-speech tagger this procedure needs to be done for several documents with diverse linguistic structures. This thesis describes the research made towards a heuristic that could provide the most suited parameter configuration for a given document according to its lexical diversity.

The capabilities brought by the bleaching technique led to the assembling of

[1] **Part-of-speech tagging** is the task of assigning a part-of-speech tag to each word in a given text. A **part of speech** is a category of lexical items which have similar grammatical properties. Common listed English parts of speech are noun, verb, adjective, adverb, pronoun, preposition, conjunction, interjection, and sometimes numeral, article or determiner.

accurate and precise applications [11–16]. A theoretical research was conducted to unveil the reasons responsible for the improvement in WiSARD performance. Bleaching is a recent technology and no theoretical background has been derived up to this point. This thesis provides a mathematical foundation for the bleaching n-tuple classifier, by analyzing the generalization capacity of both traditional and bleaching recognition schemes.

1.1 Motivation

In the last decade, given the availability of corpora in several distinct languages, research on multilingual part-of-speech tagging started to grow [1–4, 10–12, 17–24]. Part-of-speech tagging, as the basis for several other natural language processing tasks, needs to be as accurate as possible, so to avoid that potential mistakes propagate in cascade to the following tasks. There are two major issues that prevent part-of-speech tagging from being a fully straightforward process, they are homonymy and out-of-vocabulary (OOV) words. The former consists of a word having at least two different meanings (or parts of speech) while the latter are words that need to be tagged but were not seen during the training phase. Those issues can be much of a trouble in multilingual part-of-speech tagging, since they arise from the basic nature of languages and how these encode information into words. A same heuristic may work for one language and not for the other because they treat information in different ways.

Some languages have a small lexicon, mostly with words of atomic meaning, and build sentences using many of them in a very fixed order. Other languages prefer to have a freer word order, but their words tend to have more complex meanings and new ones can always be coined by appending affixes to other existing words. The languages of the former group are known as isolating and the ones of the latter as synthetic [25].[2] A same language can encode distinct pieces of information in very different manners. This way, languages can be classified in a syntheticity spectrum that range from fully isolating languages at one end to fully synthetic at the other. It is important to know where a language lies in this spectrum, so that one might determine if the tagger must be more concerned about homonymy or OOV words.

Part-of-speech taggers may use some additional information to disambiguate words that can be classified by more than a single part of speech. The context of

[2]Languages can be classified according to a morpheme-to-word ratio. A language is said to be **isolating** if this ratio is on average close to 1. This kind of languages contain very little inflection, instead relying on features like word order and auxiliary words to convey meaning. **Synthetic** languages have a higher morpheme-to-word ratio, as they prefer to form words by affixing a given number of dependent morphemes to a root morpheme. A language is said to be **polysynthetic** is the morpheme-to-word ratio is high enough, so that some words in these languages are equivalent to whole sentences in others.

a word, i.e., its adjacent words, could provide enough information to disambiguate homonymy in sentences of languages with a more fixed word order. Besides, suffixes can be used to avoid the problem of OOV words, since even if a given word does not show up during the training phase of the tagger, it is quite probable that some part of it would do show up. This thesis presents mWANN-Tagger, a part-of-speech tagger that makes use of context and word endings to mitigate misclassification due to homonymy and OOV words, respectively. It also introduces a heuristic to determine how much knowledge mWANN-Tagger should obtain from the words of a corpus and from their adjacencies. It was intended to devise a heuristic based on the lexical diversity of a corpus to build an accurate part-of-speech tagger that works well with any language.

The successful performance of mWANN-Tagger [10–12] and other applications [13–16], and the absence of a theoretical background for the bleaching n-tuple classifier were the main motivations behind the rendering of a mathematical foundation for that learning machine. The framework of statistical learning theory was chosen in a means of exploring the theoretical characteristics concerning the generalization capacity of the bleaching n-tuple classifier. Furthermore, a previous study on the search of the Vapnik-Chervonenkis (VC) dimension [26, 27] of the traditional WiSARD n-tuple classifier was devised, but no exact value was found, letting its VC dimension defined by a lower and an upper bound [28, 29]. This thesis intends to deliver exact values for the VC dimensions of both variations, which should provide future insights on how these learning machines can be enhanced and how they relate to other weighted learning systems.

1.2 Related Works

Once the amount of annotated corpora in very distinct languages started to grow, research on multilingual part-of-speech tagging appeared as an almost immediate consequence. Universal tagsets [20], large inventories of multilingual corpora [22, 23], use of translated parallel sentences to help in tag disambiguation [17–19, 21, 24] and language-agnostic part-of-speech taggers [1, 2, 4, 10–12] are some examples of research done on multilingual part-of-speech tagging.

Part-of-speech taggers often make use of features, such as context, affixes, word shapes, characters and others. They tend to have relevant information about a word being trained (or tagged). The use of language-independent features for part-of-speech tagging may be traced back to RATNAPARKHI [30]. Seven years later, TOUTANOVA *et al.* [31] also proposed a feature-rich part-of-speech tagger, the Stanford Tagger. Nowadays, those features are more commonly employed in neural taggers [4, 11, 32–36], but are also found in other architectures, like stochastic

ones [1].

Zipf's law [37, 38] (or Zipf distribution) and its generalization, the Zipf-Mandelbrot (ZM) distribution [39] are empirical laws that describe many types of data, including how words are distributed in a corpus, i.e., its lexical diversity. These probability distributions have been extensively studied since they were proposed in the first half of the previous century. Studies regarding parts of speech and power law distributions are considerably common in the literature [40–49]. The study of the Zipf's law was also employed to help part-of-speech taggers classify OOV words [50]. Those power laws also prove useful for other fields that are closely related to natural language processing (NLP). For instance, in information retrieval (IR) tasks, the Zipf's law was applied in the construction of sets of stopwords [51] and for heuristics in document retrieval [52].

Theoretical researches and architecture improvements were performed on the traditional WiSARD n-tuple classifier [9, 28, 29, 53–69]. One of the main issues raised by those studies was how to find a means to mitigate memory saturation. As WiSARD was fed with (especially if noisy) data, it started filling up every position of its memory nodes, deteriorating its capability of discriminating patterns.

Among the studies intended to lessen the effects of saturation, it is worth mentioning the work of BRADSHAW [28], where lower and upper bounds for the VC dimension of WiSARD n-tuple classifier were calculated. No exact value for this measure was found. The research made by BRADSHAW [28] provided a fertile ground for further analyses on this field. Unfortunately, the solution to the saturation problem there proposed was not that successful, as it relied on convergence and had no guarantee that it would actually happen whatsoever.

In 2010 [9] an actual way of mitigating saturation was devised, called the bleaching technique. It allowed the network to be exposed to loads of data (noisy and unnoisy) and yet keep the fidedignity of its pattern discrimination capability. This improvement considerably enhanced both accuracy and precision of traditional WiSARD n-tuple classifier applications with no performance harm on their training procedures and just a small bit on the classification step [11–16]. Despite its empirical results, no theoretical background was provided for the bleaching variation of the n-tuple classifier up to this publication.

1.3 Goals and Contributions

This thesis aims to present the potential of the bleaching n-tuple classifier for natural language processing, with a special focus in part-of-speech tagging. Its ability to handle large loads of data and still train in an agile fashion makes it a promising learning architecture for such tasks. It was also intended to determine through this

thesis the role quantitative linguistics measures might have in aiding part-of-speech taggers disambiguate the words in a corpus. Multilingual part-of-speech tagging appears as an adequate scenario for this, given that a greater variety of lexical diversity can be found when employing corpora of very diverse languages.

It is also a goal of this thesis to introduce a mathematical foundation for the bleaching n-tuple classifier through the study of its generalization capacity. Because BRADSHAW [28] did not find an exact value for the VC dimension of the traditional WiSARD n-tuple classifier, this thesis intends to search for an alternative way to calculate it in order to attain its exact value.

This thesis fulfills its objectives concerning mWANN-Tagger by finding a universal parameter configuration. In other words, a balance was met between how much knowledge mWANN-Tagger should extract from the word itself and how much from its adjacencies. This way, a single mWANN-Tagger instance was assembled, such that it produces a high level of accuracy for any annotated textual base. This contribution puts an end to the need for fine-tuning and makes mWANN-Tagger ready to be employed in any corpus regardless of its lexical diversity or language.

The theoretical research on the generalization capacity of WiSARD and bleaching n-tuple classifiers also delivers what it was intended to. A mathematical formulation is given to the bleaching n-tuple classifier. Also, a same exact value is calculated for the VC dimensions of these learning machines. Further discussions concerning this result provide insights on how to achieve a lower VC dimension for the bleaching n-tuple classifier and its impacts on the classification capabilities of that learning machine. This work may contribute as a fertile ground for future researches concerning the bleaching n-tuple classifier, such as the development of new tie-breaking policies.

In summary, the main contributions of this thesis are:

- Test the capabilities of the bleaching n-tuple classifier through the exposure to large loads of data;
- Verify the potential of the bleaching n-tuple classifier for natural language processing, with a special focus in multilingual part-of-speech tagging;
- Study the role quantitative linguistics measures might have in aiding part-of-speech taggers disambiguate words in a corpus;
- Provide a universal parameter configuration for mWANN-Tagger, so it can be applied to new corpora with no need of fine-tuning its design parameters;
- Introduce a mathematical foundation for the bleaching n-tuple classifier;

- Determine the exact value of the VC dimension of the traditional WiSARD n-tuple classifier;
- Determine the VC dimension of the bleaching n-tuple classifier;
- Discuss what the VC dimensions of both variants tell about the improvement provided by the bleaching technique;
- Discuss how the value calculated for the VC dimension relates to the presented practical application, and how one can improve the bleaching n-tuple classifier.

1.4 Thesis Structure

Chapter 2 contains the background in n-tuple classifiers. It presents WiSARD and bleaching architectures and their training and recognition procedures. A mathematical formulation is provided for both models. One similar to that of STECK [8] as used for the WiSARD model, and an analogous version was devised for its bleaching variant. This chapter also opens a discussion about tie-breaking policies for the bleaching n-tuple classifier, as it proves necessary for the assembling of a formal mathematical formulation for that learning machine. As a consequence, a subclass of bleaching n-tuple classifier is defined for that purpose.

Chapter 3 revisits mWANN-Tagger, initially proposed in [10]. It gives insights on how lexical diversity may contribute in multilingual part-of-speech tagging and provides background on Zipf's law and its extension, the Zipf-Mandelbrot distribution. The chapter then describes a series of experiments and analyses meant to detect a relation between the parameters of these probability distributions and those of mWANN-Tagger. Through those analyses, a universal parameter configuration is found and its impact in mWANN-Tagger performance is tested through a comparison with other state-of-the-art multilingual part-of-speech taggers.

Chapter 4 presents the calculations of the VC dimensions of WiSARD and bleaching n-tuple classifiers. At first, the chapter offers a background on statistical learning theory needed to understand the following content. Through this background, the importance of discovering the VC dimension of the bleaching n-tuple classifier is made clear. A finite VC dimension needed be found, so to guarantee that the model is consistent with the empirical risk minimization principle [27]. Exact (finite) values are met for both learning machines. The proofs that lead to these results are rendered in this chapter.

Finally, Chapter 5 contains a summary of the thesis, along with conclusions that can be drawn from the work here presented. It also offers suggestions for how this research may be extended in the future.

Chapter 2

The WiSARD Model

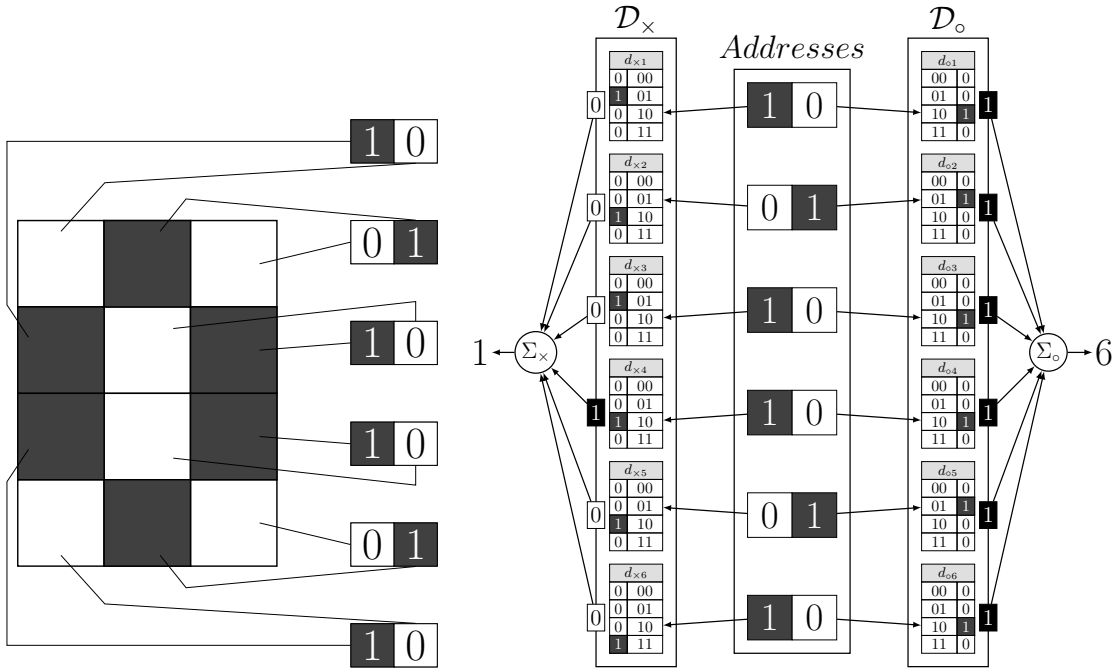
WiSARD is an n-tuple classifier, and as such, it has two key features. The splitting of its input into various n-tuples and the storage of its learned knowledge in memory nodes, which are addressed by those n-tuples. This chapter contemplates the traditional WiSARD model and its enhanced version produced by the bleaching technique.

The learning of new patterns is made through simple changes in the content of the memory nodes of the system. There is no need for convergence-dependable processes or pretty complex calculations. In other words, n-tuple classifiers have efficient training procedures and so, they are adequate for empirical studies where there is a large amount of hyperparameters to be tuned and massive loads of data to be learned.

2.1 The Traditional WiSARD n-tuple Classifier

The n-tuple method was initially proposed in [7] and formally defined in [8]. Its most widely known implementation, WiSARD, was made in 1982 [5, 6]. It showed that it was actually possible to assemble the n-tuple classifier.

The WiSARD n-tuple classifier is considered a weightless neural network for the resemblance of its memory nodes to actual neurons. The weightless paradigm is characterized by the network learned knowledge being stored inside its neuronal nodes whereas weighted models store it in their synaptic connections. WiSARD is also known as a RAM-based neural network since the implementation of its memory nodes was effected with actual random access memories (RAMs). Due to this, the network memory nodes are also known as RAM-nodes or simply RAMs.



(a) Pseudorandom mapping. (b) Canonical classifier.

Figure 2.1: Traditional WiSARD classifier

2.1.1 System Architecture

The traditional WiSARD n-tuple classifier is a Boolean neural network. It receives bit strings as inputs and assign to them classes. This is made through the use of structures called discriminators. Each discriminator is associated to a single class.

The discriminators have a very simple structure. They consist of a pseudorandom mapping that shuffles the bits of the network input and split them into tuples, which address memory nodes. Figure 2.1a portrays a picture mapped into the series of tuples $10 : 01 : 10 : 10 : 01 : 10$. An n-tuple classifier is defined according to its tuple length n and the amount of memory nodes in each discriminator N.

In a canonical n-tuple classifier a same pseudorandom mapping can be used for every discriminator. This way, N n-tuples address all memory nodes of each discriminator. Figure 2.1b depicts a canonical n-tuple classifier, where the same series of tuples produced in Figure 2.1a addresses the memory nodes of both discriminators, $\mathcal{D}_\circ$ and $\mathcal{D}_\times$.

2.1.2 Training Procedure

Training starts by the network setup, where all memory positions are initialized with 0. Every time a training observation and its corresponding class are sent to the network, the model selects the discriminator of that class and marks every memory position addressed by the observation. Positions marked this way have their stored

values set to 1. At the end of the training procedure, every memory position should store a 1 if they were addressed at least once, and 0 otherwise.

2.1.3 Recognition Procedure

If a new input pattern is presented to the network, every discriminator responds with a similarity score, representing how close this new pattern is to the learned knowledge stored in the discriminator. The similarity score is defined as the amount of memory nodes whose addressed positions store 1, i.e. those positions which were accessed at least once during the training step. This score is represented by summation devices $\Sigma_\circ$ and $\Sigma_\times$ in Figure 2.1b.

The network opts for the discriminator whose score is the highest and assigns its class to the input pattern. If there are at least two discriminators that share the highest response, then the network outputs that a tie happened. In the latter case, the classification system may apply some policy to choose one class or use a draw procedure.

The discriminator similarity score is the number of addressed memory positions. It implies that the number of nodes N (or the size of the addressing tuples n) plays an important role in defining the generalization capability of the WiSARD classifier. High values of N (or small ones of n) reduces the chance of the similarity score getting too low if a newly presented pattern is slightly different from what was learned by the network by only a few bits. For example, given a network that is presented to a 30-bit input pattern and only one of its RAMs does not recognize it, then (*i*) if there are 30 RAMs addressed by 1-bit tuples, the similarity score of the classifier will be 29/30; (*ii*) if there are 15 RAMs addressed by 2-bit tuples, the similarity lowers to 14/15; and (*iii*) if there is only one RAM addressed by the whole network input, then the similarity will be 0. So, the greater is N the more generalizing is the WiSARD system.

2.1.4 Mathematical Formulation

An n-tuple classifier is defined according to N, its amount of memory nodes, n, their address length, and $C = \{c_a, c_b, \ldots\}$, the set of classes it can discriminate. Let X be an ordered set of input data points, $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$. An input $\mathbf{x}_k \in X$ must be such that $\mathbf{x}_k \in \{0,1\}^{Nn}$ for it to be read by the n-tuple classifier. The example of Figure 2.1b employs two discriminators, $\mathcal{D}_\circ$ and $\mathcal{D}_\times$, each one assigned to a particular class ($\circ$ and $\times$, respectively). There are $N = 6$ memory nodes in each discriminator, which are addressed by n-tuples of length $n = 2$.

Any input $\mathbf{x}_k \in X$ is firstly transformed into a feature matrix $\mathbf{Z}_k \in \{0,1\}^{N\times 2^n}$ via an addressing function $A : \{0,1\}^{Nn} \to \{0,1\}^{N\times 2^n}$, which is pseudorandomly

generated at the network setup. The feature matrix $\mathbf{Z}_k$ produced this way has a single element per row equal to 1 and the remaining ones to 0. This is a direct consequence of the fact that each n-tuple produced by $A(\mathbf{x}_k)$ addresses one and only one memory position. For instance, input $\mathbf{x}_k = 10:01:10:10:01:10$ given in Figure 2.1b, is converted into feature matrix

$$\mathbf{Z}_k = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.1}$$

By assigning the feature matrix first column to address 00, the second one to 01, the third to 10 and the last to 11, one can verify that every n-tuple 10 (the first, third, fourth and last) is represented by matrix row (0 0 1 0). The remaining rows of $\mathbf{Z}_k$ represent n-tuples 01.

Let $\mathcal{W}_{N,n,C}$ be the family of functions that denotes every n-tuple classifier defined by parameters N, n and C ($\mathcal{W}$ stands for WiSARD). Let also $W_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots) \in \mathcal{W}_{N,n,C}$ be a function representing an n-tuple classifier instance defined as above, whose learned knowledge on any class c is characterized by a memory matrix $\mathbf{M}_c \in \{0,1\}^{N \times 2^n}$. Every network instance $W_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots)$ has card(C) discriminators, denoted $\mathcal{D}_a, \mathcal{D}_b, \ldots$. They are represented by $\mathbf{M}_a, \mathbf{M}_b, \ldots$, the memory matrices which store their learned knowledge. $\mathbf{M}_c$ is defined as $\mathbf{M}_c = [m_{c,i,j}]_{N \times 2^n}$, where $m_{c,i,j} \in \{0,1\}$. The example of Figure 2.1b has two discriminators, $\mathcal{D}_\circ$ and $\mathcal{D}_\times$, which are mathematically represented by memory matrices

$$\mathbf{M}_\circ = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.2}$$

and

$$\mathbf{M}_\times = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2.3}$$

respectively. Each row of these matrices corresponds to the content of their respective RAM node. For instance, the first row of $\mathbf{M}_\circ$ is (0 0 1 0), which is exactly the same sequence of values in the memory positions of $d_{\circ 1}$.

For any feature matrix $\mathbf{Z}_k$ mapped from an input $\mathbf{x}_k \in X$, $W_{N,n,C}(\mathbf{Z}_k; \mathbf{M}_a, \mathbf{M}_b, \ldots)$ produces a score vector $\mathbf{s} = [s_a, s_b, \ldots]^{\mathrm{T}}$, where $s_c \in \{0, 1, \ldots, N\}$ is the score of $\mathcal{D}_c$, defined as

$$s_c = \sum_{i=1}^{N} \sum_{j=1}^{2^n} m_{c,i,j} z_{k,i,j}, \tag{2.4}$$

where $z_{k,i,j}$ is the element at i-th row and j-th column of $\mathbf{Z}_k$. Given the score vector $\mathbf{s}$, function $W_{N,n,C}(\mathbf{Z}_k; \mathbf{M}_a, \mathbf{M}_b, \ldots)$ returns the corresponding class as

$$W_{N,n,C}(\mathbf{Z}_k; \mathbf{M}_a, \mathbf{M}_b, \ldots) = \underset{c \in C}{\operatorname{argmax}}\, s_c. \tag{2.5}$$

The n-tuple classifier of Figure 2.1b produces scores $s_\circ = 6$ and $s_\times = 1$, which are the outputs of $\mathcal{D}_\circ$ and $\mathcal{D}_\times$. Because the chosen class is that whose score is the highest, then that n-tuple classifier opts for $\circ$ as the class to be applied to input $\mathbf{x}_k = 10 : 01 : 10 : 10 : 01 : 10$.

2.2 The Bleaching Technique

Traditional n-tuple classifiers tend to experience misclassification problems when trained with large amounts of data. This occurs because many memory positions should be set to 1 even if they were accessed only once due to a slightly noisy observation. This problem is known as **saturation** and is considered one of the major disadvantages of the traditional n-tuple classifier since its conception [7].

Some attempts were performed in order to overcome this drawback [28, 53, 56, 58, 68]. A solution was devised in 2010, the bleaching technique [9]. Its contribution as a major upgrade to the traditional architecture allowed the production of very accurate and precise applications [11–15].

2.2.1 Training Procedure

Bleaching n-tuple classifier training procedure differs from the traditional one by the storage of any integer value in the memory positions, instead of only 0 and 1. Every element of the memory nodes initialize as 0 during the network setup, just like in the traditional process. When a position is addressed at training, its stored value is incremented by 1, whereas it would only be set to 1 in the traditional procedure, independently on its original value. Summarizing, at the end of the training step,

each memory position stores how many times it was accessed.

2.2.2 Recognition Procedure

The integer values stored in the memory nodes give the network a better representation of the trained data. However, another structure must be added to the traditional architecture for a proper classification. The bleaching technique introduces a threshold β (known as the bleaching threshold). It is responsible for defining which memory positions should contribute to the similarity score of the class.

Threshold β is a non-negative integer number. The canonical bleaching n-tuple classifier employs a single threshold for the whole network. If a memory position is addressed during the classification step, then it should be further subjected to β. A node fires 1 if its addressed position stores a value greater than β, otherwise it fires 0. So, the similarity score of a discriminator of a fixed-threshold bleaching n-tuple classifier is the amount of its accessed positions whose stored value is greater than β. Figure 2.2 portrays the classification of an input pattern by a bleaching n-tuple classifier with $\beta = 2$.

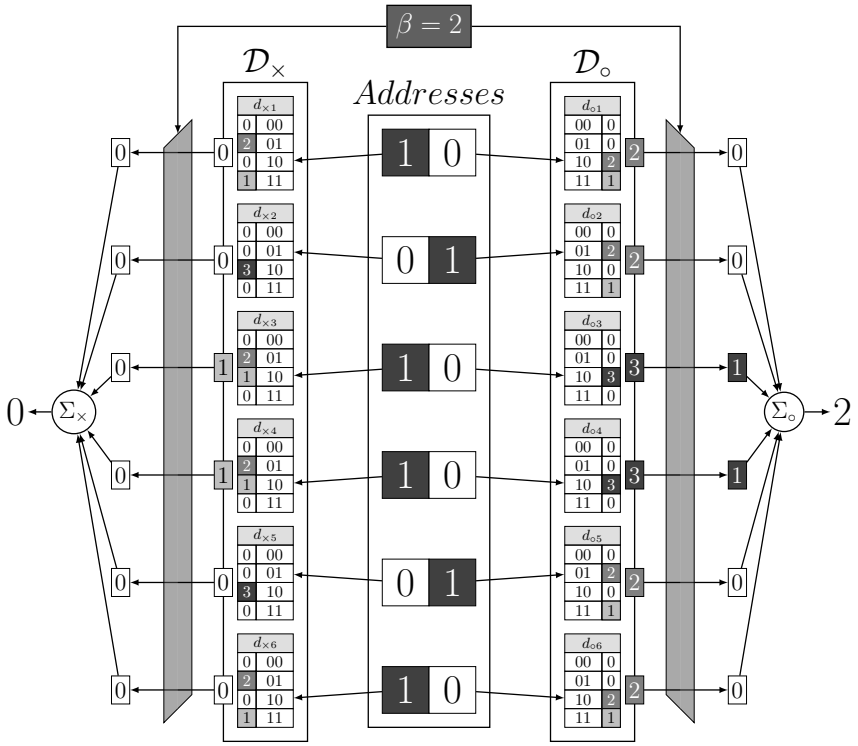


Figure 2.2: Bleaching classifier.

In practical applications, however, a dynamic threshold is employed instead. The threshold is initialized as $\beta = 0$ at the network setup. Thus, at first glance the dynamic-threshold bleaching n-tuple classifier works like the traditional classifier, where the score of a discriminator is defined as the number of accessed positions

whose stored value is greater than 0. If a selection criterion for a given class is not satisfied at the classification of a pattern, β is incremented by 1 and a new score is calculated. The iterative procedure keeps going until that criterion is satisfied.

Tie-breaking Policies – Min-max Bleaching n-tuple Classifier

There are several possible criteria to select a winning discriminator. As a direct extension to the traditional n-tuple method, the most straightforward criterion consists of electing the discriminator with the single highest score. If two or more discriminators share the same highest response, then there is a tie and the bleaching threshold β must be increased. This may produce some anomalous cases for discrimination among three or more classes. For instance, in the case given in Table 2.1 there are always at least two discriminators whose output is the highest.

Table 2.1: Ambiguous scenario, where an input pattern is not clearly recognized by any single discriminator.

Accessed Contents		β			
		0	1	2	3
$\mathcal{D}_1$	$v_{1,1}:2$ $v_{1,2}:2$ $v_{1,3}:1$	3	2	0	0
$\mathcal{D}_2$	$v_{2,1}:3$ $v_{2,2}:1$ $v_{2,3}:1$	3	1	1	0
$\mathcal{D}_3$	$v_{3,1}:3$ $v_{3,2}:2$ $v_{3,3}:0$	2	2	1	0

A tie-breaking policy less prone to those anomalous cases was employed for mWANN-Tagger. There all discriminators begin as valid candidates a priori and the bleaching threshold is set up to $\beta = 0$. For an ever-increasing β, the scores of the candidate discriminators are compared. A discriminator ceases to be candidate if its score is not the highest being compared. Once there is only a single candidate discriminator, then its class is assigned to the network input. If there are at least two candidate discriminators at the end of the iterative process, then the network outputs that a tie happened. Similarly to the traditional model, in such case, the classification system may apply some policy to choose one class or use a draw procedure.

In the scenario displayed in Table 2.1, the chosen class would be that of $\mathcal{D}_1$, since it would be the only remaining candidate discriminator after two iterations. At $\beta = 0$, $\mathcal{D}_3$ would be discarded because its score, 2, is lower than the highest attained, 3. Next, at $\beta = 1$, $\mathcal{D}_2$ ceases to be considered a valid candidate, for $\mathcal{D}_1$ has a highest score.

For the determination of the VC dimension, this tie-breaking policy is degenerated to a simpler case, for the VC theory paradigm relies on biclass learning systems. The chosen class is the one whose similarity score is the greatest when β is the lowest one for which there is no tie between the scores. This learning system is henceforth named minimum-threshold maximum-score bleaching n-tuple classifier, or **min-max bleaching n-tuple classifier** for short. Every further mention in this thesis on bleaching n-tuple classifiers refers to this particular family of machine learning schemes.

2.2.3 Mathematical Formulation

Let $\mathcal{B}_{N,n,C}$ be the family of functions that denotes every min-max bleaching n-tuple classifier defined by parameters N, n and C defined in Section 2.1.4 ($\mathcal{B}$ stands for bleaching). Let also $B_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots) \in \mathcal{B}_{N,n,C}$ be a function representing a min-max bleaching n-tuple classifier instance defined as above, whose learned knowledge on any class c is characterized by memory matrix $\mathbf{M}_c$. Let $B_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots)$ be defined in an analogous way to $W_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots)$ (see Section 2.1.4). The only difference lies on the memory matrices $\mathbf{M}_c$, which are now defined as $\mathbf{M}_c = [m_{c,i,j}]_{N \times 2^n}$, where $m_{c,i,j} \in \mathbb{N}$. Figure 2.2 presents a bleaching n-tuple classifier with two discriminators $\mathcal{D}_\circ$ and $\mathcal{D}_\times$. They are mathematically represented by memory matrices

$$\mathbf{M}_\circ = \begin{pmatrix} 0 & 0 & 2 & 1 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 2 & 0 & 1 \\ 0 & 0 & 2 & 1 \end{pmatrix} \tag{2.6}$$

and

$$\mathbf{M}_\times = \begin{pmatrix} 0 & 2 & 0 & 1 \\ 0 & 0 & 3 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 2 & 0 & 1 \end{pmatrix}, \tag{2.7}$$

respectively. One can note that the same relation between memory matrix rows and RAM node contents mentioned in Subsection 2.1.4 applies to the bleaching variants of $\mathbf{M}_\circ$ and $\mathbf{M}_\times$.

Let $\mathbf{s}^{(\beta)} = \left[s_a^{(\beta)}, s_b^{(\beta)}, \ldots\right]^{\mathrm{T}}$ be the score vector of the bleaching n-tuple classifier for a given fixed threshold β. The scores $s_c^{(\beta)} \in \{0, 1, \ldots, N\}$ represent the number of nodes of discriminator $\mathcal{D}_c$ whose addressed positions store values greater than β. So, they can be defined as

$$s_c^{(\beta)} = \sum_{i=1}^{N} \sum_{j=1}^{2^n} \mathbf{1}_{(\beta,\infty)}(m_{c,i,j} z_{k,i,j}) \,, \tag{2.8}$$

where $\mathbf{1}_S(x)$ is the indicator function that returns 1 if $x \in S$ and 0 otherwise. $m_{c,i,j}$ and $z_{k,i,j}$ are respectively elements of matrices $\mathbf{M}_c$ and $\mathbf{Z}_k$. Figure 2.2 shows the scores produced by a bleaching n-tuple classifier when subjected to a bleaching threshold $\beta = 2$. In this case, the learning machine produces scores $s_\circ^{(2)} = 2$ and $s_\times^{(2)} = 0$.

At last, two mathematical formulations for the output of the min-max bleaching n-tuple classifier are given, one for the multiclass model and a simpler one for its biclass variant. The definition of the latter is relevant for the determination of the VC dimension of this learning system.

Let $B_{N,n,C}(\cdot; \mathbf{M}_a, \mathbf{M}_b, \ldots) \in \mathcal{B}_{N,n,C}$ be a function representing a multiclass min-max bleaching n-tuple classifier, whose learned knowledge is stored in memory matrices $\mathbf{M}_a, \mathbf{M}_b, \ldots$. For a feature matrix $\mathbf{Z}_k$ mapped from an input $\mathbf{x}_k$, $B_{N,n,C}$ returns

$$B_{N,n,C}(\mathbf{Z}_k; \mathbf{M}_a, \mathbf{M}_b, \ldots) = c, \tag{2.9}$$

such that for every $c' \neq c \in C$, $\underset{\beta'}{\operatorname{argmin}}\, s_c^{(\beta')} > s_{c'}^{(\beta')} < \underset{\beta'}{\operatorname{argmin}}\, s_c^{(\beta')} < s_{c'}^{(\beta')}$. This procedure is exemplified in Subsection 2.2.2, where a classification scenario is offered in Table 2.1 and further explained in the nearby text.

In a biclass variant, with a class set $C = \{c_a, c_b\}$, this procedure can be written in a simplified form. In such case, for a feature matrix $\mathbf{Z}_k$, $B_{N,n,C}$ returns the corresponding class as

$$B_{N,n,C}(\mathbf{Z}_k; \mathbf{M}_a, \mathbf{M}_b) = \underset{c \in \{c_a, c_b\}}{\operatorname{argmax}}\, s_c^{(\beta)}, \tag{2.10}$$

where $\beta = \underset{\beta'}{\operatorname{argmin}}\, s_a^{(\beta')} \neq s_b^{(\beta')}$. This part of the mathematical formulation refers to the explanation given in the last paragraph of Subsection 2.2.2. There it is written that the chosen class is the one whose score is the highest, given a bleaching threshold β that is the lowest one for which there is no tie between the discriminators.

Chapter 3

mWANN-Tagger

mWANN-Tagger [11] constitutes a multilingual language-independent part-of-speech tagger. It is a solution to the training performance problem that arises when the number of languages used by a POS-tagger increases. It is intended to profit from the efficiency of weightless artificial neural networks in classification tasks and a method to straightforwardly predict the optimal values for some classification parameters. It is an evolution of WANN-Tagger [70], a part-of-speech tagger whose architecture is based on the WiSARD model.

mWANN-Tagger is intended to be used for very diverse languages. The tagger has distinct forms to treat each language. Some require information to disambiguate a word that can be attained through context, whereas others get information from parts of word itself. mWANN-Tagger needs to balance the importance of context and of the form of the word itself. This chapter describes how mWANN-Tagger should do this balance and how it is related to the syntheticity of a language.

3.1 Basic Concepts: Quantitative Linguistics

There are two major issues that prevent part-of-speech tagging from being a fully straightforward process: homonymy and OOV words. The former consists of a word having at least two different meanings (or parts of speech) while the latter are words that need to be tagged but were not seen during the training phase.

These issues arise from the basic nature of languages and how they encode information into words. Some languages have a small lexicon, mostly with words of atomic meaning, and build sentences using many of them in a very fixed order. Other languages prefer to have a freer word order, but their words tend to have more complex meanings and new ones can always be coined by appending affixes to other existing words.

The languages of the former group are known as isolating and the ones of the latter as synthetic [25]. A same language can encode distinct pieces of information in

very different ways. This way, languages can be classified in a syntheticity spectrum that range from fully isolating languages at one end to fully synthetic at the other. It is important to know where a language lies in this spectrum, so that one might determine if the tagger must be more concerned about homonymy or OOV words.

GREENBERG [71] proposed the index of synthesis, a straightforward form to quantify the syntheticity of a language. It is simply the average number of morphemes per word in a given corpus. However, this metric requires every word of a corpus to be morphemically divided and there is no automatic way of doing it. Besides, a same word may be morphemically divided in different ways depending on what can be considered a morpheme.

An alternative to this metric can be achieved through an analysis of the lexical diversity of a corpus. Isolating languages tend to have small sets of words, as most of its words represent atomic meanings. On the other hand, synthetic languages tend to have a large set of words, which can always be increased by appending affixes to existing ones. Those distinct natures of encoding information reflect in the word distribution of texts in those languages.

Some languages prefer to apply more synthetic strategies in some kind of information and more isolating ones in some other. This way, the classification of languages according to its degree of synthesis can be seen as the task of finding its position on a syntheticity spectrum. The fully isolating languages lie on one end of the spectrum and the fully synthetic on the other. Languages that are neither fully isolating nor fully synthetic should lie somewhere in between those extrema.

For the sake of comparison, translations of the 9th article of the Universal Declaration of Human Rights in languages of very distinct degrees of synthesis are given below. Firstly, the English version of the article is presented, then the same article appears in some languages, ordered according to their syntheticity. It can be seen that languages encode a same information in very different ways. Vietnamese, an isolating language employs many small words (most of them monosyllabic) whereas Greenlandic, a polysynthetic language, prefers fewer larger words.

- **English**: No one shall be subjected to arbitrary arrest, detention or exile;
- **Vietnamese**: Không ai bị bắt, giam giữ hay đày đi nơi khác một cách độc đoán;
- **Finnish**: Ketään ei saa mielivaltaisesti pidättää, vangita tai ajaa maanpakoon;
- **Greenlandic**: Kinaluunniit namminissarsiortumik tigusarineqassanngliq, tigummigallagassanngortitaassanani imaluunniit nunagisamit peersitaassanani.

The way languages encode information into words affects their lexical diversity. Isolating languages have a fairly small set of words, which may be used for diverse meanings (sometimes with different parts of speech). Synthetic words are less prone to homonymy. Tagging sentences on those languages may yet be a challenge, since new words can be created through affixation and there is a high probability of finding words not seen during the training phase. In other words, there is an intuitive relation between isolating languages and homonymy, and between synthetic languages and OOV words. A comparison between properties of isolating and synthetic languages is offered in Table 3.1.

Table 3.1: Comparisons of properties of isolating and synthetic languages

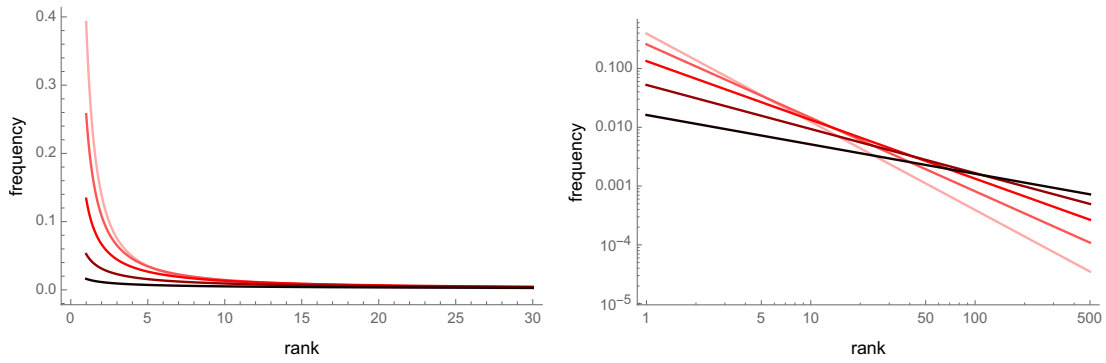
Property	**Language**	
	Isolating	**Synthetic**
Word information	Atomic	Complex
Word order	Fixed	Free
Lexical diversity	Low	High
Problem in POS-tagging	Homonymy	OOV words

Heuristics may be used to resolve ambiguities that arise from homonymy and OOV words. Single words that can belong to a vast variety of parts of speech may have this ambiguity mitigated by the use of the contexts where those words appear. That is, adjacent terms may help the part-of-speech tagging of a word by restricting the amount of possible parts of speech it may assume. The issue of OOV words can also become less troublesome if the endings of the words are also considered during the training phase. Most of the languages prefer suffixing to prefixing [72], so those endings could help with hints on the part of speech of an OOV word. Even if the word does not show up in the training corpus it is highly probable that another word which shares its endmost affix does show up. Hence, one may determine which heuristics a part-of-speech tagger should use by knowing how synthetic is the language of a corpus. How much focus is given to each of those heuristics depends on knowing how synthetic is the language.

3.2 Basic Concepts: Statistics of Word Distribution

Words in a text are distributed according to a power law [37, 38]. Such behavior was extensively observed in languages, even in extinct and yet-untranslated ones [73]. It was empirically verified that the frequency of every word is proportional to the frequency of the most common one by a power of its rank. That is, if the words of a text are ranked according to its frequency (from the most common to the rarest),

then the r-th most frequent word is proportional to the most frequent word by a factor of $r^{-\alpha_1}$, where α_1 is a positive real number. This specific kind of power law is known as the Zipf's law. The effect of different values of α_1 for the Zipf distribution can be checked in Figure 3.1, where curves are plotted for five different values of α_1, 0.5, 0.75, 1.0, 1.25 and 1.5. Values of α_1 closer to 0 are represented by darker colors.



(a) Linear-linear plot (b) Log-log plot

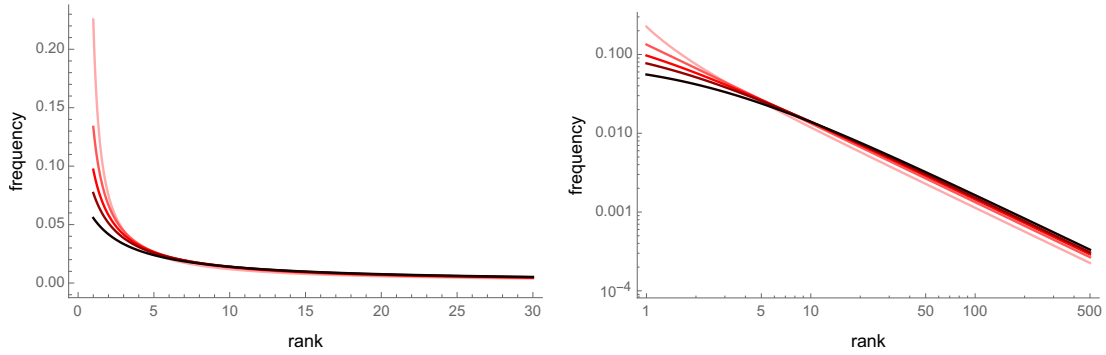
Figure 3.1: Word rank per word frequency graphs depicting the effect of α_1 on the steepness of the Zipf curve.

Zipf's law was further generalized by MANDELBROT [39], who noted that the highest ranked elements appeared less frequently than what was initially expected by the Zipf's law. To overcome this discrepancy of the distribution function in relation to the collected data, MANDELBROT [39] proposed the addition of an offset parameter α_2, which would penalize the highest ranked elements but not the remaining ones. This generalization is known as the Zipf-Mandelbrot law (ZM law) or the Zipf-Mandelbrot distribution (ZM distribution) and is characterized by the formula

$$f(r; \alpha_1, \alpha_2) \propto (r + \alpha_2)^{-\alpha_1}. \tag{3.1}$$

It can be noted that the penalization produced by the offset parameter only affects the highest ranked elements, because it would make the numerator of Equation 3.1 become much smaller when r is small enough. On the other hand, for large values of r the penalizing effect of α_2 would be negligible. The effect of different values of α_2 for the ZM distribution can be checked in Figure 3.2, where curves are plotted for a fixed $\alpha_1 = 1$ and five different values of α_2, -0.5, 0.0, 0.5, 1.0 and 2.0. Values of α_2 closer to -1 are represented by lighter colors.

Isolating languages use words with specific functions (e.g., copula and auxiliary verbs, pronouns and adpositions) far more often than other words. So, the steepness of their Zipf curves is very high and it is reflected on the distribution exponent α_1 tending to be greater than 1. In synthetic languages this special kind of words is rarely employed. Usually the use of such words would be substituted by an affix that would be appended to another one, creating a brand new word. In other words,



(a) Linear-linear plot (b) Log-log plot

Figure 3.2: Word rank per word frequency graphs depicting the effect of penalizing parameter α_2 on the ZM curve.

synthetic languages have larger vocabularies, but almost none of its words appear much more often than any other. Consequently, their Zipf curves are much less steep and have heavier tails. The exponent α_1 of those distributions tends to be somewhere between 0 and 1. The closer it is to 0 the more synthetic is a language.

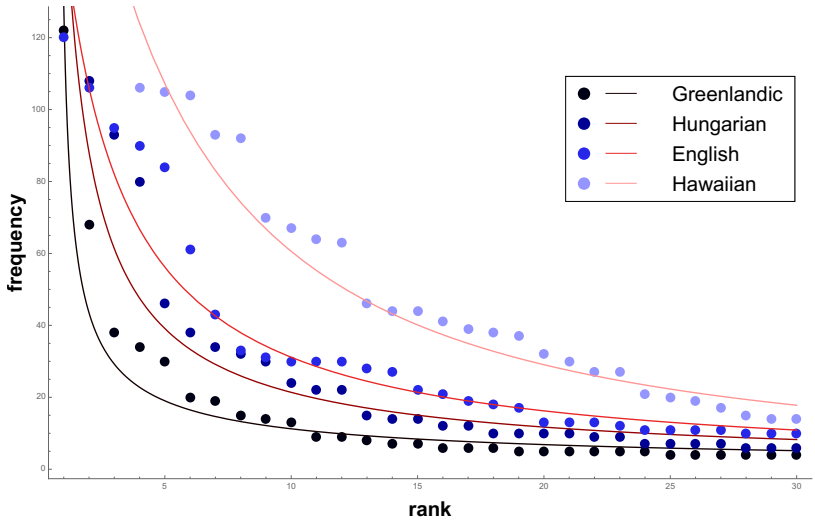


Figure 3.3: Zipf curves for different languages. The curves are fitted according to the dots, which express the observed data. The horizontal and vertical axes represent a same word rank and frequency, respectively.

The difference between the steepness of the Zipf curve of different languages is displayed in Figure 3.3. It depicts the rank-frequency relation between every word (represented by a dot) of the Universal Declaration of Human Rights in four very different languages, Hawaiian – the most isolating, English, Hungarian and Greenlandic – the most synthetic. The more synthetic a language the darker the color used in the graph of Figure 3.3. This way, the position of a language in the syntheticity spectrum can be quantified by knowing the steepness of this power law.

The relation between language syntheticity and lexical diversity [74–76] as well

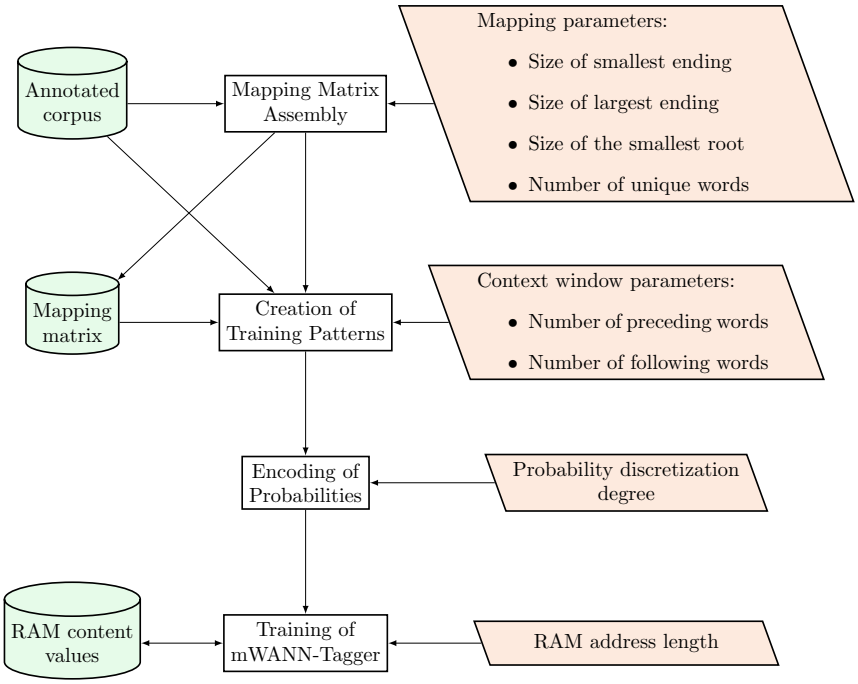
as the application of ZM parameters as quantitative measures for lexical diversity [77, 78] corroborate the plausibility of employing ZM parameters as syntheticity indicators.

ZM parameters α_1 and α_2 of a textual base can be estimated through a maximum likelihood (ML) estimation procedure [79]. Each type is ranked according to its frequency in that given text, assigning rank 1 for the most frequent type, rank 2 for the second most frequent one, and so on. This way, a textual base is converted to pairs (rank, frequency) and through them ML estimators $\hat{\alpha_1}$ and $\hat{\alpha_2}$ can be produced. These estimators should work as close estimators of the actual ZM parameters α_1 and α_2.

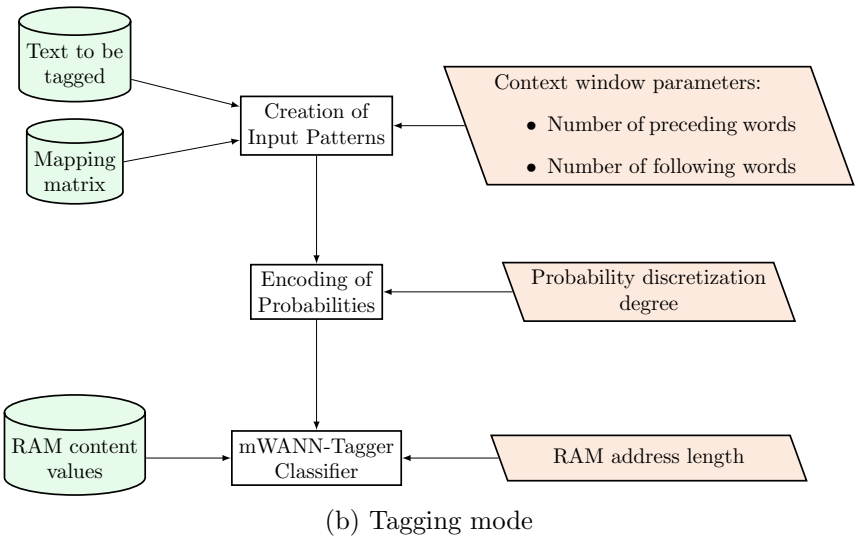
3.3 mWANN-Tagger Architecture

mWANN-Tagger is a feature-rich part-of-speech tagger and as such it has some design parameters that need to be tuned. The tagger was intended to be used for multilingual POS-tagging. Consequently, a fast-training learning model was employed as the core learning mechanism of mWANN-Tagger. It allowed for an exhaustive search for the best suited parameter configurations to be successfully done. mWANN-Tagger has two distinct modes, the training and the tagging mode. The tagger parameters can be checked in the schematic diagrams of the these modes, which are depicted in Figures 3.4a and 3.4b, respectively.

At the training mode (Figure 3.4a), mWANN-Tagger receives an annotated corpus as input and through the use of four of its design parameters it assembles a mapping matrix. The parameters used this way are called *mapping parameters*. They inform mWANN-Tagger which parts of the words in the corpus have relevant information capable of aiding mWANN-Tagger in accurately tagging OOV words. The mapping matrix works like a dictionary where each token extracted from the corpus is associated to a vector of relative frequencies, indicating how probable a token can be marked with a given tag. A context window is also used to prevent problems caused by homonymy. Two design parameters are used to produce this window, they are the number of words that precede the one being processed by mWANN-Tagger, and also the number of words that follow it. The tagging can then produce input patterns for training, by reading the words with their contexts in the annotated corpus and by mapping them to their corresponding vectors of relative frequencies. The WiSARD network only works with binary input data. To solve this, each relative frequency of the input pattern is encoded into an array of bits. The size of this array is defined by another mWANN-Tagger design parameter. Given a binary input, the WiSARD network can finally train that input pattern. The architecture of the employed network depends on a last design parameter, the



(a) Training mode



(b) Tagging mode

Figure 3.4: mWANN-Tagger modes

RAM address length. At this step, the RAM content values are read and updated for each new input pattern.

At the tagging mode (Figure 3.4b), mWANN-Tagger follows similar steps. It creates input patterns given a text to be tagged, the mapping matrix assembled at the training mode, and the design parameters that define a context window. Each input pattern produced this way is then encoded into an array of bits, which is sent to the WiSARD network to be classified in one of a group of tags. This classification is only possible due to the bleaching technique, since the RAM content values read in this step can be notably high for the network exposure to a large load of data.

3.3.1 Tagset

A universal tagset was proved necessary to create a multilingual POS-tagger. A correlation among the grammatical classes of every language and a normalization of the tagsets were intended, so that mWANN-Tagger could be of help for other multilingual NLP applications. Another benefit of the universal tagset was the adoption of a fixed amount of tags, reducing the differences among the datasets, so that only the parameters of the tagger need to be tuned. This was important because one of the goals of this work was to investigate a correlation between the ZM parameters, α_1 and α_2, and mWANN-Tagger design parameters. Furthermore, PETROV *et al.* [20] results corroborate the use of a universal tagset instead of specific ones for each language. The tagset employed by mWANN-Tagger has $T = 14$ tags (Table 3.2) and is similar to ones adopted in other works [18, 20].

Table 3.2: Tagset of mWANN-Tagger

N	Noun	ADJ	Adjective
ADV	Adverb	V	Verb
PRON	Pronoun	DET	Determiner
ADP	Adposition	NUM	Cardinal Number
CJ	Conjunction	MW	Measure Word
PART	Particle	INTJ	Interjection
PUNC	Punctuation	MISC	Miscellaneous

3.3.2 mWANN-Tagger Design Parameters

The WiSARD-based tagger mWANN-Tagger employs eight adjustable design parameters during its training and tagging modes, as depicted in Figures 3.4a and 3.4b. These design parameters are from now on referred solely as parameters. Four of those are related to word structure and are used to mitigate the problem of OOV words. They are hereinafter referred to as *mapping parameters* for their use in the mapping matrix assembly. Also, mWANN-Tagger is a sliding window tagger, and so two other parameters are needed to determine how much adjacent context is relevant for the tagging process. The last two parameters are related to characteristics of the WiSARD classifier employed as the core of this tagger.

As it is graphically explained in Figure 3.4a, the annotated corpus is initially read and through this a mapping matrix is assembled. This matrix works like a dictionary, where every line has a word and a list of relative frequencies (real values between 0 and 1) of this word to each tag. The number of relative frequencies assigned to a given word is equal to the number of possible tags. The mapping matrix structure is shown in Figure 3.5a. The relative frequency f of tag t_j assigned

	ADJ	ADP	ADV	CJ	DET	INTJ	MISC	MW	N	NUM	PART	PRON	PUNC	V
we	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
saw	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.02	0.0	0.0	0.0	0.0	0.98
her	0.0	0.0	0.0	0.0	0.64	0.0	0.0	0.0	0.0	0.0	0.0	0.36	0.0	0.0
duck	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.29	0.0	0.0	0.0	0.0	0.71
in	0.0	0.98	0.02	0.0	0.0	0.0	2e-4	0.0	5e-5	0.0	0.0	0.0	0.0	0.0
the	0.0	0.0	0.0	0.0	1.0	0.0	5e-5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
⋮														
·	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
⋮														
-ell	0.01	0.0	0.39	0.0	6e-4	0.08	0.0	0.0	0.28	0.0	0.0	0.0	0.0	0.24
-ness	0.0	0.0	0.0	0.0	0.0	0.0	8e-4	0.0	0.99	0.0	0.0	0.0	0.0	6e-3
-ly	0.08	0.0	0.85	0.0	0.0	1e-4	7e-5	0.0	0.05	0.0	0.0	0.0	0.0	0.01
⋮														
-	0.07	0.11	0.06	0.05	0.12	5e-4	1e-3	0.0	0.23	0.01	0.01	0.05	0.13	0.16

(a) Mapping matrix example lines

We saw her duck in the dell.
[we|PRON] [saw|V] [her|PRON] [duck|V] [in|ADP] [the|DET] [dell|N] [.|PUNC]

(b) Sentence and its corresponding annotated version

T

	ADJ	ADP	ADV	CJ	DET	INTJ	MISC	MW	N	NUM	PART	PRON	PUNC	V
$b \rightarrow$ -	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
we	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
$a \rightarrow$ saw	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.02	0.0	0.0	0.0	0.0	0.98

(c) Training pattern example with $b = 1$, $a = 1$ and $T = 14$ tags

	ADJ	ADP	ADV	CJ	DET	INTJ	MISC	MW	N	NUM	PART	PRON	PUNC	V
the	0.0	0.0	0.0	0.0	1.0	0.0	5e-5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
-ell	0.01	0.0	0.39	0.0	6e-4	0.08	0.0	0.0	0.28	0.0	0.0	0.0	0.0	0.24
·	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0

(d) Training pattern example with ending *-ell*

Figure 3.5: mWANN-Tagger procedure of encoding sentences into arrays of probabilities

to word w_i is

$$f(w_i, t_j) = \frac{Q(w_i, t_j)}{\sum_j Q(w_i, t_j)}, \tag{3.2}$$

where $Q(w_i, t_j)$ is a counting function that returns how many times w_i appears tagged as t_j in the annotated corpus.

The mapping matrix also has lines for every ending that counts as relevant according to the mapping parameters. A sequence of characters is considered a valid key to the mapping matrix if it is L characters long, such that $L_{\min} \leq L \leq L_{\max}$, where $L_{\min}$ and $L_{\max}$ are two mapping parameters, the **'size of the smallest ending'** and the **'size of the largest ending'**, respectively. Very small endings must be discarded, because they are not very informative. Usually such endings are

common to many words of pretty distinct parts of speech, and so employing them would eventually bias the tagger towards an undesired tag, harming its accuracy. Too large endings must also be set aside because they would make the mapping matrix unnecessarily big, decreasing the tagger performance.

Even if a sequence fulfills the above constraint, it can only be added as an entry of the mapping matrix if it appears as the ending of at least U unique words that have $L_{\text{root}} + L$ or more characters, where U and L_{root} represent parameters **'amount of unique words'** and **'size of the smallest root'**, respectively. This further constraint is proposed for two reasons. Firstly, an ending should be common to a certain amount of distinct words, for they are more probable of being present in an OOV word and also because this restriction avoids the mapping matrix getting filled with endings that happen only once. The second reason lies in the fact that some endings should not be treated as suffixes, but as parts of the root instead. For instance, the ending *-ly* is often associated with adverbs, like in *fairly*. However, neither *ugly* nor *fly* are adverbs. In the former the actual suffix is only *-y*, which stands for adjectives, while in the latter both characters belong to the root.

The training patterns are then created, given the mapping matrix, the annotated corpus and the desired amount of adjacent words in the sliding window, which is provided by two more parameters, the **'number of preceding words'** b and the **'number of following words'** a. They consist of arrays of relative frequencies, which are assigned to a particular tag. They are formed by concatenating smaller arrays obtained from the mapping matrix lines whose keys are the words of the context window, or their corresponding endings. Figure 3.5c shows an example where the context window [–, we, saw] from the sentence "We saw her duck in the dell." is encoded into an array of relative frequencies. It is worth noting that if the context window exceeds the limits of a sentence, empty strings are used to ensure that the tagger input size is kept. This example will be revisited in Section 3.3.3, where the tagger input construction will be explained in greater detail.

The remaining parameters of mWANN-Tagger are the **'probability discretization degree'** δ and the **'RAM address length'** n. The former is responsible for encoding the training pattern arrays of relative frequencies into bit strings, for the WiSARD input is composed only by bits. In this encoding procedure every relative frequency is transformed into a bit array through the thermometer code. That is, given a discretization degree δ, one must pick δ (usually equally-spaced) values from the interval $[0, 1]$, p_1, p_2, ..., p_δ. A relative frequency f encodes into a δ-bit string, such that each i-th bit is 1 if f is greater than p_i and 0 otherwise. For example, if one wants to discretize the value $f = 0.7$ into $\delta = 4$ bits by using the values 0.0, 0.25, 0.5 and 0.75 for p_1, p_2, p_3 and p_4, respectively, then its resulting bit string would be 1110, since $f = 0.7$ is lesser than $p_4 = 0.75$, but greater than the remaining p_is (see

Figure 3.6 for a graphical representation).

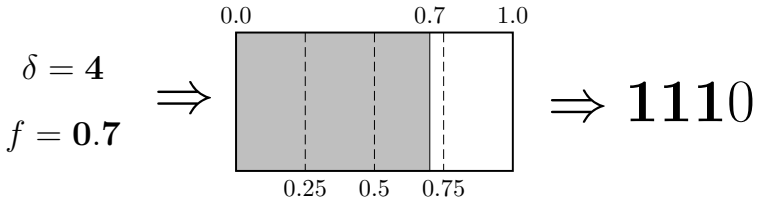


Figure 3.6: Discretization of a relative frequency into a string of bits

3.3.3 Input Construction

Each term in a sentence becomes a different input for mWANN-Tagger. For example, in the sentence "We saw her duck in the dell.", there should be eight different inputs, one for each term, *we*, *saw*, *her*, *duck*, *in*, *the*, *dell* and the period (.).

For each one of those terms, the tagger takes into account its context window, i.e. its adjacent words, to disambiguate any potential homonymy. The size of this window varies according to the syntheticity of the employed corpus. For instance, in Figure 3.5c the depicted context window consists of an empty string and words *we* and *saw*. The example shows a context of one word before (represented by $b = 1$) and one after ($a = 1$). The word being tagged is *we* and the surrounding others are *saw*, which lies right after the main word, and the empty string, that precedes it.

Every word of the sliding window is encoded into an array of empirical probabilities, which are actually its relative frequencies to each part of speech of the tagset (see Equation 3.2 and Figure 3.5c). This array represents the *a priori* word class empirical distribution of the encoded term. It is employed to aid the tagger in understanding what is intended to be tagged. Those empirical distributions are indirectly obtained from the corpus through the mapping matrix, which was described in Section 3.3.2.

If the term being encoded appears as the key of one of the mapping matrix lines, then the array of empirical probabilities is straightforwardly obtained from this line. If it does not show at the mapping matrix (in Figure 3.5d this happens to the word *dell*), then the chosen mapping matrix line is the one whose key is the largest existing ending of that term. For example, in Figure 3.5d the largest existing ending of *dell* is *-ell*, so its probability array should be used instead.

Note that the ending chosen this way must not conflict with the constraint imposed by the size of the smallest root. For example, ending *-ell* may only be employed if the size of the smallest root is 1 or smaller, since *d-* contains one character. If the size of the smallest root were 2, then the employed ending should be *-ll* instead.

Seldom if ever a term is not represented by any line of the mapping matrix, it is then encoded according to a default case, which is an array with the proportions

of each part of speech in the whole corpus. The default case and its corresponding array of probabilities is depicted in the last mapping matrix line of Figure 3.5a.

The concatenation of those arrays is sent to mWANN-Tagger as its input. At the training mode, the tagger receives this input and its expected word class. Only the nodes of the corresponding discriminator are accessed, having their stored values incremented accordingly. In the tagging mode, the tagger sends the input to all its discriminators. The output of mWANN-Tagger is then the word class associated to the discriminator chosen by its WiSARD network.

It is important to remind that the WiSARD network only receives Boolean values. So, mWANN-Tagger probabilities are sent to its core classifier as several binary inputs, according to its discretization degree δ, as presented in Section 3.3.2.

3.4 Experiments

Next, one intended to discover potential benefits that previous knowledge on the word distribution of a corpus has in part-of-speech tagging. In order to do so, a set of corpora was collected, with distinct ways of encoding information. In other words, a set of annotated corpora in languages with contrasting degrees of synthesis and lexical diversities was used.

At first, mWANN-Tagger parameters were finely tuned in search for the most suitable parameter configurations for eight corpora in notably different languages. These annotated corpora were employed as a means to have a good representation of a large slice of the syntheticity spectrum. Next, ML estimators $\hat{\alpha_1}$ and $\hat{\alpha_2}$ were obtained from these corpora. Pearson coefficients were calculated to verify if there is some correlation between mWANN-Tagger finely tuned design parameters and the ZM parameter estimators $\hat{\alpha_1}$ and $\hat{\alpha_2}$.

Preliminary analyses suggested that a single parameter configuration may be applied to the eight aforementioned languages and that the mWANN-Tagger instance produced by this configuration was as accurate as the language-dependent ones obtained through tuning. That single parameter configuration was subjected to six additional corpora representing different parts of the syntheticity spectrum, including both extrema. Finally, the performance of this universal instance of mWANN-Tagger was compared to those of state-of-the-art multilingual part-of-speech taggers through experiments on Universal Dependencies treebanks [22].

3.4.1 The Zipfian Nature of Languages

Eight corpora were used to verify a potential relation between the lexical diversity (and syntheticity) of a corpus and the parameter configuration that produces the

best performing mWANN-Tagger instance for that corpus.

This choice of datasets came up from the fact that this set of corpora is satisfactorily heterogeneous, i.e., their languages use very different (and sometimes even contrasting) forms of encoding information into text. The main goal of this work was that one could gauge the optimal parameter configuration for any other corpus given its ZM parameters. So, the heterogeneity is crucial for the assessment of the aforementioned relation between lexical diversity and the mWANN-Tagger parameter configuration. The employed corpora are presented in Table 3.3.

Table 3.3: Original corpora

Short name	**Corpus**	**Language**	**# types**	**# tokens**
BNC	Brown National Corpus [80]	English	55819	1166139
NEGRA	NEGRA Corpus [81]	German	47921	337881
TUT	Turin University Treebank [82]	Italian	10828	76927
TüBa-J/S	Tübinger Baumbank des Japanischen Spontansprache [83]	Japanese	3249	156543
PennChinese	Penn Chinese Treebank 6.0 [84]	Mandarin Chinese	52120	1098801
Bosque	Bosque (Floresta Sintá(c)tica) [85]	Portuguese	27686	214003
RNC	Russian National Corpus [86]	Russian	122597	1288969
METU	METU-Sabancı Turkish Treebank [87]	Turkish	19381	53869

The corpora appear in eight different languages. They are English, German, Italian, Japanese, Mandarin Chinese, Portuguese, Russian and Turkish. They range from analytic and rather isolating languages, like Mandarin Chinese and English, to synthetic ones, like Russian and Turkish. These languages show very distinct ways of organizing information into sentences. A brief explanation on their main characteristics, concerning syntheticity and some particular peculiarities, are presented below:

- English: Nouns are inflected by number, but there is no gender. Every word that modifies a noun is kept uninflected. Verbs inflect in tense and mood, but neither in person nor in number. Notable exceptions are the verb 'to be' and the present tense third person singular form of most verbs. It is very common to use periphrastic constructions, like auxiliary verbs or words (do, have, be, will, shall) together with non-finite verb forms [88–90];

- German: It retains several synthetic elements of its Indo-European ancestry, like compound words and intricate verb conjugation, case and plural systems. However, some simplifications also occurred through time, e.g., there is no gender distinction in plural words, the case system almost never applies to the nouns (but applies to adjectives, determiners and pronouns), and some word classes may also have a simplified declension, also known as weak declension. Besides, despite its synthetic trait, German has a not so free word order [91, 92];

- Italian: As a Romance language, its verb conjugation system is quite complex. There are three plural markers, they are *-i*, *-e* and *-a*. Italian has a very fusional nature. For instance, it employs many contractions and its plural markers are used by substituting the final vowel of a word by them and not only appending them to its end [93];

- Japanese: It is slightly more synthetic than English due to a more complex verb system and to compound words borrowed from Chinese languages. Japanese also uses a large inventory of particles, which make the language seem more synthetic when they are counted as clitics and not as words on their own [94];

- Mandarin Chinese: It is an analytic language, i.e. none of its words suffers any kind of inflection. There is no plural, and tense and mood are represented by additional words instead of suffixes, as in English. However, many words are composed by two or more roots [95, 96];

- Portuguese: It has a very complex verb system, like all Romance languages. However, its nouns and adjectives suffered a simplification process, in which their plural form are made by simply appending the plural marker *-s* at the end of the word (with a few exceptions) [97, 98];

- Russian: It is an Indo-European language. It suffered far less simplifications than its sibling languages, making it a very synthetic one. For instance, its case system contains 6 cases, compared to 4 of German, and they inflect nouns, and not only adjectives, determiners and pronouns. Its word order is also freer than that of any aforementioned language [99];

- Turkish: It is a very synthetic and agglutinative language. Many words can be coined by appending suffixes to simpler words. More than ten suffixes can be appended this way [100].

In Section 3.1, it was said that words (and other strings of symbols) of a text are distributed according to a Zipf-Mandelbrot distribution, which contains two

parameters, α_1 and α_2, and that they can be approximated by ML estimators $\hat{\alpha_1}$ and $\hat{\alpha_2}$. Such estimators were produced for every corpus presented on Table 3.3. Their values are displayed at Table 3.4.

Table 3.4: Zipf-Mandelbrot parameters of the original corpora

Corpus	**Language**	$\hat{\alpha_1}$	$\hat{\alpha_2}$
BNC	English	1.108	1.442
NEGRA	German	1.042	0.889
TUT	Italian	1.063	0.771
TüBA-J/S	Japanese	1.337	2.755
PennChinese	Mandarin Chinese	1.048	0.512
Bosque	Portuguese	1.071	0.348
RNC	Russian	1.036	0.309
METU	Turkish	0.815	-0.769

It can be perceived in Table 3.4 that the values for $\hat{\alpha_1}$ do not necessarily follow the syntheticity of languages. This occurs due to the very different nature of the texts of the corpora used. For instance, Japanese is more synthetic than English, but the steepness parameter $\hat{\alpha_1}$ of TüBA-J/S was farther from 0 than BNC's. TüBA-J/S is a corpus composed of sentences extracted from spontaneous speech, whereas BNC was produced with texts from a wide variety of genres, including spoken, fiction, magazines, newspapers, and academic sources.

Several distinct sets of mWANN-Tagger parameters were used to find which parameter configuration would produce the best performing tagger for each corpus of Table 3.3. Configurations found this way and also their corresponding accuracies and sample deviations are described in Table 3.5.

Two parameters were not included in Table 3.5, the size of the largest ending and the discretization degree. The values of those parameters are not provided because they work in a quite particular way. Searches for optimal values for those parameters showed that the higher their values the higher is the tagger accuracy. This improvement, however, is negligible when their values lie above given thresholds (approximately 7 for the size of the largest ending and close to 160 for the discretization degree). This way, the use of any value above those thresholds would produce a quite accurate tagger. The values 10 and 200 were respectively employed for the size of the largest ending and the discretization degree in the configurations presented in Table 3.5.

As a first step to show a potential relation between ZM and mWANN-Tagger parameters, Pearson correlation coefficients between them were extracted. A coefficient with an absolute value close to 1 would indicate that there is an almost clear relation between an mWANN-Tagger and a ZM parameter. The results of this task are depicted in Figure 3.7. Only the context window parameters show a no-

Table 3.5: Empirically-obtained mWANN-Tagger parameter configurations for the corpora of Table 3.3. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure.

Language	$\hat{\alpha_1}$	$\hat{\alpha_2}$	**Smallest ending**	**Smallest root**	**# unique words**	**Context window**	**Address length** (n)	**Acc. (%)**
English	1.108	1.442	2	1	1	[1,1]	122	97.76 (0.05)
German	1.042	0.889	1	2	1	[1,1]	108	97.23 (0.15)
Italian	1.063	0.771	1	1	2	[1,1]	85	96.32 (0.26)
Japanese	1.337	2.755	1	0	1	[1,1]	80	98.88 (0.09)
Mandarin	1.048	0.512	1	2	1	[1,1]	118	94.17 (0.08)
Portuguese	1.071	0.348	1	2	2	[1,1]	86	97.01 (0.12)
Russian	1.036	0.309	1	1	1	[1,1]	60	97.53 (0.15)
Turkish	0.815	-0.769	3	1	1	[2,3]	50	92.34 (0.40)

table correlation coefficient absolute value. Every other parameter has a correlation coefficient whose absolute value is lower than 0.5.

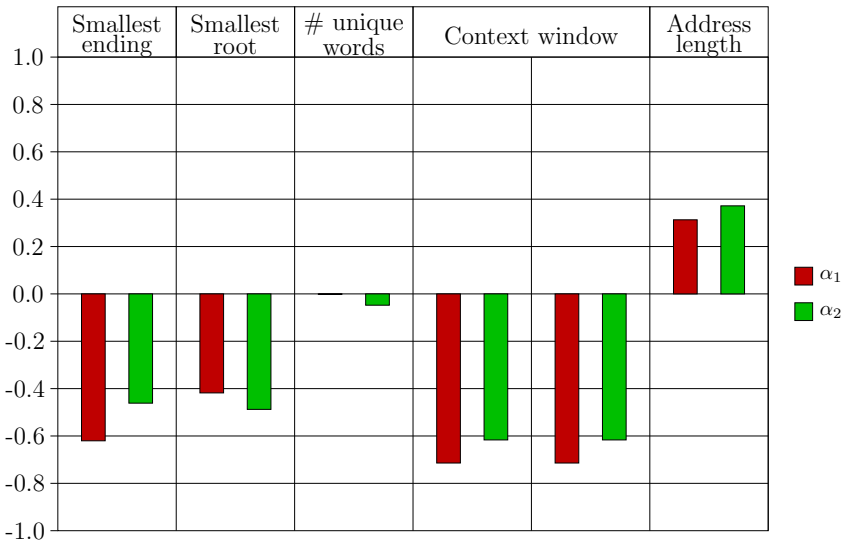


Figure 3.7: Correlation between mWANN-Tagger and Zipf-Mandelbrot parameters of corpora of Table 3.3.

Because the correlation coefficient were low, a further experiment was required. A cross-linguistic analysis was made to determine if the configurations presented in Table 3.5 were truly best suited for their corresponding corpora. This analysis consisted in subjecting each corpus of Table 3.3 to all tagger parameter configurations of Table 3.5. It was intended to verify how better a tagger can be compared to others, when it was employing the parameter configuration best suited for a given

corpus. The experiment, however, showed that there were little to no variation in the tagger accuracy if it employed a particular parameter configuration. Table 3.6 shows that there were even some cases where a corpus was most accurately tagged with configurations best suited for other corpora.

Table 3.6: Cross-linguistic analysis. Rows represent the corpora being tested and columns the corresponding tagger configurations. Table cells show the accuracy (%) of a tagger in this particular scenario. Bolded cells represent the greatest accuracy achieved for a given corpus.

	English	German	Italian	Japanese	Mandarin	Portuguese	Russian	Turkish
English	**97.76**	97.75	97.69	97.74	97.75	97.74	97.75	97.50
German	**97.26**	97.19	96.97	97.19	97.23	97.22	97.14	96.09
Italian	96.10	96.22	**96.37**	96.08	96.15	96.35	96.11	95.32
Japanese	98.86	98.87	98.82	98.87	98.84	98.84	**98.89**	98.37
Mandarin	94.12	94.15	94.02	94.09	**94.16**	94.10	94.01	93.73
Portuguese	96.90	96.97	96.93	96.93	**97.00**	**97.00**	96.90	96.56
Russian	97.03	97.02	96.87	97.02	97.07	96.99	**97.09**	96.80
Turkish	91.90	91.87	91.07	91.92	91.87	91.48	92.04	**92.38**

3.4.2 The Optimal Set of Parameters

The correlation test performed on Section 3.4.1 indicated that only the context window has correlation greater than 0.5 for both ZM parameters. The size of the smallest ending is also another parameter that has correlation greater than 0.5 with ZM-distribution steepness parameter α_1.

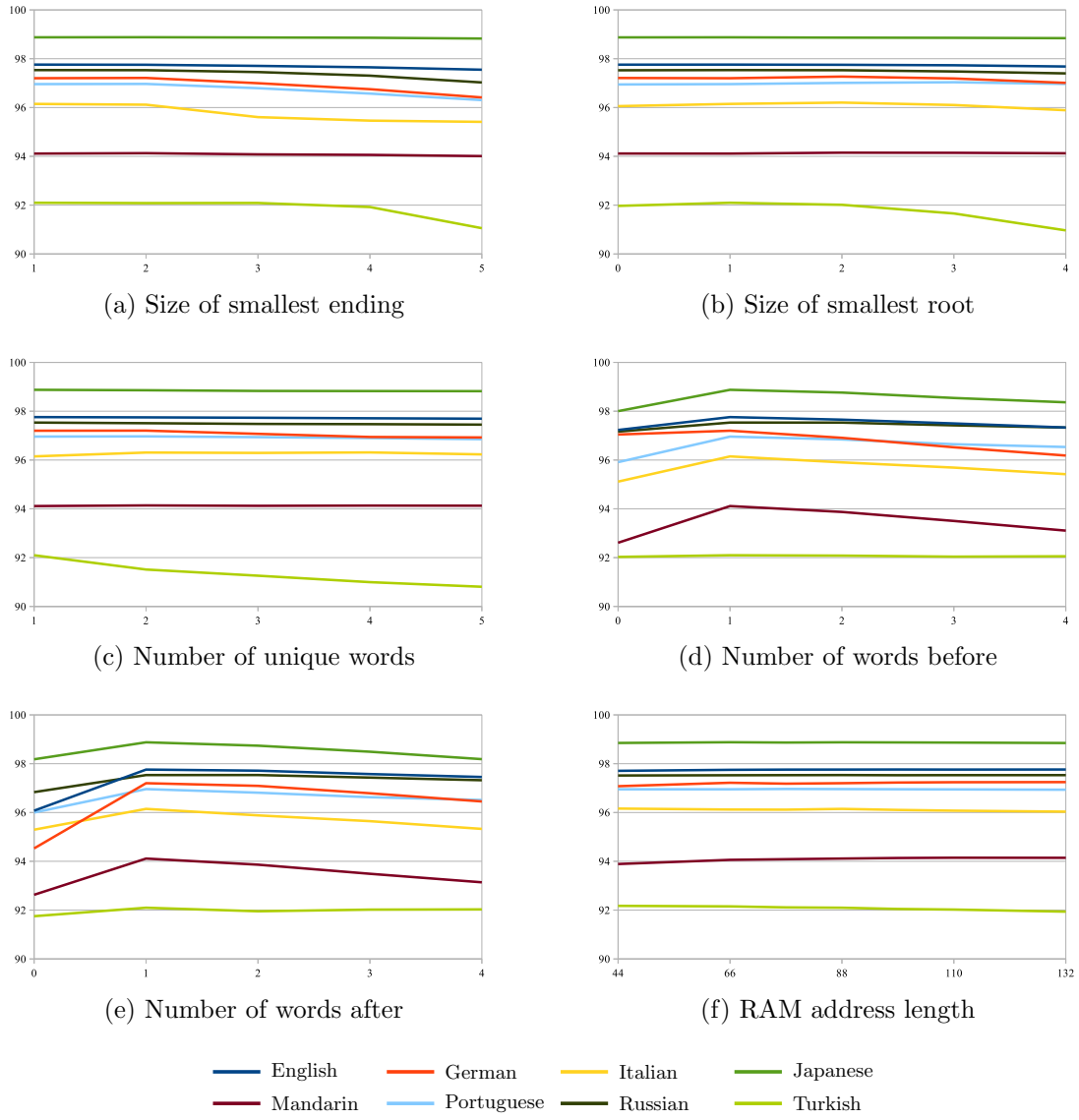
During the search for the optimal set of parameters described in Section 3.4.1, two of them showed to be indifferent to language syntheticity. Most of the remaining tagger parameters also had an absolute correlation coefficient less than 0.5 to the ZM parameters – some of them pretty close to 0, like the *number of unique words*.

The cross-linguistic analysis also indicated no noteworthy variation for the tagger accuracy when using a configuration distinct from the one presented in Table 3.5. A single tagger parameter configuration was then proposed. It was such that the tagger assembled from that configuration should achieve accuracies for Table 3.3 corpora that were close to the language-specific ones of Table 3.5. To test its robustness and reliability, a sensitivity analysis was done, by testing some variations in the parameter configuration. The same sensitivity analysis procedure was done to six new corpora to further test the robustness and reliability of the tagger configuration.

mWANN-Tagger Universal Parameter Configuration

Several parameters of Table 3.5 showed little variation among the different languages. This can be verified by how close are the accuracies achieved by different

instances of mWANN-Tagger when subjected to a same corpus. Due to that little variation, a universal tagger parameter configuration was assembled. The values of its components were derived from the mode of the values of their corresponding parameters. The only exception being the RAM address length, for which the average was used instead.



(a) Size of smallest ending

(b) Size of smallest root

(c) Number of unique words

(d) Number of words before

(e) Number of words after

(f) RAM address length

Figure 3.8: Sensitivity analysis for the tagger parameters. Graphs depict the accuracy variation caused by fluctuations on parameter values.

Table 3.7 revisits the parameter configurations presented in Table 3.5 and introduces the universal configuration produced by the aforementioned procedure. It was tested on every corpus of Table 3.3 and mWANN-Tagger achieved accuracies close to the ones displayed in Table 3.5. Table 3.8 shows that mWANN-Tagger is as quite as accurate with the universal configuration as it is using the empirically obtained language-specific ones of Table 3.5.

Table 3.7: mWANN-Tagger parameter configurations of Table 3.5 and the derived universal configuration

Language	$\hat{\alpha_1}$	$\hat{\alpha_2}$	**Smallest ending**	**Smallest root**	**# unique words**	**Context window**	**Address length** (n)
English	1.108	1.442	2	1	1	[1,1]	122
German	1.042	0.889	1	2	1	[1,1]	108
Italian	1.063	0.771	1	1	2	[1,1]	85
Japanese	1.337	2.755	1	0	1	[1,1]	80
Mandarin	1.048	0.512	1	2	1	[1,1]	118
Portuguese	1.071	0.348	1	2	2	[1,1]	86
Russian	1.036	0.309	1	1	1	[1,1]	60
Turkish	0.815	-0.769	3	1	1	[2,3]	50
Universal config.	—	—	**1**	**1**	**1**	**[1,1]**	**88**

Table 3.8: Accuracies for the empirical and universal configurations. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure.

Language	**Empirical config. (%)**	**Universal config. (%)**
English	97.76 (0.05)	97.76 (0.04)
German	97.23 (0.15)	97.20 (0.14)
Italian	96.32 (0.26)	96.15 (0.22)
Japanese	98.88 (0.09)	98.88 (0.10)
Mandarin	94.17 (0.08)	94.11 (0.07)
Portuguese	97.01 (0.12)	96.96 (0.13)
Russian	97.53 (0.15)	97.53 (0.04)
Turkish	92.34 (0.40)	92.10 (0.41)

To evaluate the robustness and reliability of the configuration, a sensitivity analysis was performed. It constituted of testing mWANN-Tagger for similar configurations close to the universal one. Those similar configurations were produced by fixing every parameter but one, which could assume values close to the corresponding parameter of the universal configuration. Figure 3.8 shows that the highest accuracies are achieved when the parameters assume the values they already have at the desired configuration, indicating the robustness of mWANN-Tagger with this language-independent configuration.

Summary of Results: In this section, a universal parameter configuration for mWANN-Tagger was introduced and its accuracy was compared to the accuracies for language-specific configurations. It was then subjected to a sensitivity analysis, where mWANN-tagger was tested with similar configurations. The universal configuration showed to be robust, for the highest accuracy achieved on those experiments

Table 3.9: New corpora

Short name	Corpus	Language	# types	# tokens
BTB	BulTreeBank [101]	Bulgarian	34365	253124
TDT	Turku Dependency Treebank [102]	Finnish	49935	181022
Paris7	French Treebank – Paris 7 [103]	French	38520	587149
LDT	Latin Dependency Treebank [104]	Classical Latin	23474	165551
Pirahã	MIT Pirahã Corpus [105]	Pirahã	1253	3720
VLSP	VLSP Vietnamese Treebank [106]	Vietnamese	13666	223303

were produced by mWANN-Tagger with that same configuration. Even when some of its parameters changed by only a small amount, the tagger accuracy lowered by solely a tiny fraction.

New Corpora

To further ensure the applicability and robustness of the universal configuration, the aforementioned procedures should be performed to new corpora, especially for languages that are at least somewhat different from the ones of Table 3.3. Six new corpora were used to subject this configuration to further tests. Table 3.9 presents the chosen corpora. They are documents composed in very distinct kinds of language, ranging from the very isolating Vietnamese (VLSP Vietnamese treebank [106]) to a polysynthetic Amazonian language, Pirahã (MIT Pirahã Corpus [105]). The corpora of Table 3.9 were chosen due to their languages, which have some peculiarities that made them good test cases for mWANN-Tagger universal configuration. These languages and their main peculiarities are:

- Bulgarian: It is one of the least synthetic Slavic language together with Macedonian, especially concerning their noun systems. It suffered a simplification process common to all the Balkans linguistic area. The Slavic background, responsible for a historically complex grammar and a more recent simplification on its grammatical structure makes Bulgarian a good candidate to test mWANN-Tagger universal configuration. Even more because the configuration already applies to another Slavic language (Russian) and to languages far less synthetic than the ones of Slavic family [107];

- Finnish: It is a member of the Uralic family, known for having a very complex grammar, with several grammatical cases and conjugations, but no gender.

Finnish is a very agglutinative language and possesses a consonant gradation system, which makes some roots look notably different from their base forms. Its highly agglutinative nature and the use of consonantal gradation make Finnish a good candidate. The fact that the universal configuration applies to Turkish, an agglutinative language with no consonantal gradation, corroborates the choice for Finnish [108];

- French: It is a Romance language, like Italian and Portuguese, which were in Table 3.3. But differently from the two mentioned languages, French passed through enormous phonetic changes, producing several homophones. However, its spelling barely changed since Old French. The effects of this process can be perceived in words like *chantez*, *chanter* and *chanté*, which are all pronounced the same but are spelled quite differently. Hence, French has a hybrid nature. It remains with a high lexical diversity due to the variety of inflections (plural markers, conjugation and so on), but it also shows characteristics of analytic languages, like the extensive use of pronouns and a quite regular word order. The hybrid nature of French is the main reason it was chosen as a test case [109];

- Classical Latin: It was chosen due to its relation to the Romance languages already presented, but also because it is far more synthetic than all its daughter languages. For example, no Romance language retains Classical Latin six grammatical cases or its passive voice conjugation system. The contrast between the similarities it has with its daughter languages and the syntheticity it had and its daughter languages no longer do is the main reason for the use of Classical Latin [110];

- Pirahã: It is a polysynthetic Amazonian language. It represents an extremum in the syntheticity spectrum that mWANN-Tagger was not yet subjected to. Pirahã is also quite known for neither having a counting system nor recursive sentence-embedding structures [105, 111];

- Vietnamese: It is a remarkable example of a language of the Mainland Southeast Asia (MSEA) linguistic area, which is mainly composed by analytical and isolating languages. Vietnamese is an isolating language, whose words are mostly monosyllabic and some others are disyllabic. Its word order is very fixed and almost no composition or derivation occurs. For belonging to a region of the syntheticity spectrum that mWANN-Tagger was not yet subjected to, Vietnamese was chosen as a candidate to test the applicability and robustness of the universal parameter configuration [112].

ML estimators were produced for the ZM parameters of the corpora of Table 3.9, as explained in Section 3.1. Table 3.10 introduces the estimators $\hat{\alpha_1}$ and $\hat{\alpha_2}$ for the corpora of Table 3.9. It also informs the accuracy of mWANN-Tagger with the universal parameter configuration for the same corpora.

Table 3.10: Zipf-Mandelbrot parameters and accuracy for the new corpora. The numbers in parentheses in column "Acc." represent its sample deviation (%) in a 10-fold-cross-validation procedure.

Corpus	**Language**	$\hat{\alpha_1}$	$\hat{\alpha_2}$	**Accuracy for the universal config. (%)**
BTB	Bulgarian	1.019	0.415	98.03 (0.10)
TDT	Finnish	0.870	-0.518	95.35 (0.17)
Paris7	French	1.108	1.393	97.81 (0.08)
LDT	Classical Latin	0.982	1.284	96.17 (0.15)
Pirahã	Pirahã	0.832	-0.558	87.84 (1.50)
VLSP	Vietnamese	1.153	6.246	92.43 (0.14)

Sensitivity analysis on the new corpora

The same sensitivity analysis procedure of Section 3.4.2 was done for the corpora of Table 3.9. It was intended to verify if the accuracies in Table 3.10 were actually the best for those corpora. This experiment was important because the tagger was subjected to more extreme conditions, like very isolating and polysynthetic languages.

The result of the sensitivity analysis is depicted in Figure 3.9, where it can be seen that either the highest accuracies were achieved by the universal configuration or the sensitivity analysis produced little to no accuracy variation. The same result appeared in the sensitivity analysis of Section 3.4.2 (cf. Figure 3.8). The only exception lies with the Pirahã corpus, as there are configurations that perform better than the universal one by up to 1% (cf. Figure 3.9c).

However, the Pirahã corpus is very small when compared to other bases, as can be perceived in Tables 3.3 and 3.9. The effects of its tiny size can be noted in the relatively high sample deviation of its accuracy, in Table 3.10. The 1.5% sample deviation of the tagger accuracy for the Pirahã corpus with the universal configuration is even higher than the aforementioned 1% accuracy difference in the sensitivity analysis. So, although there are configurations that seem to perform a little bit better than the universal one, the sensitivity analysis for the Pirahã corpus is yet inconclusive and the universal parameter configuration can still be considered a good candidate for the best mWANN-Tagger parameter configuration for any language.

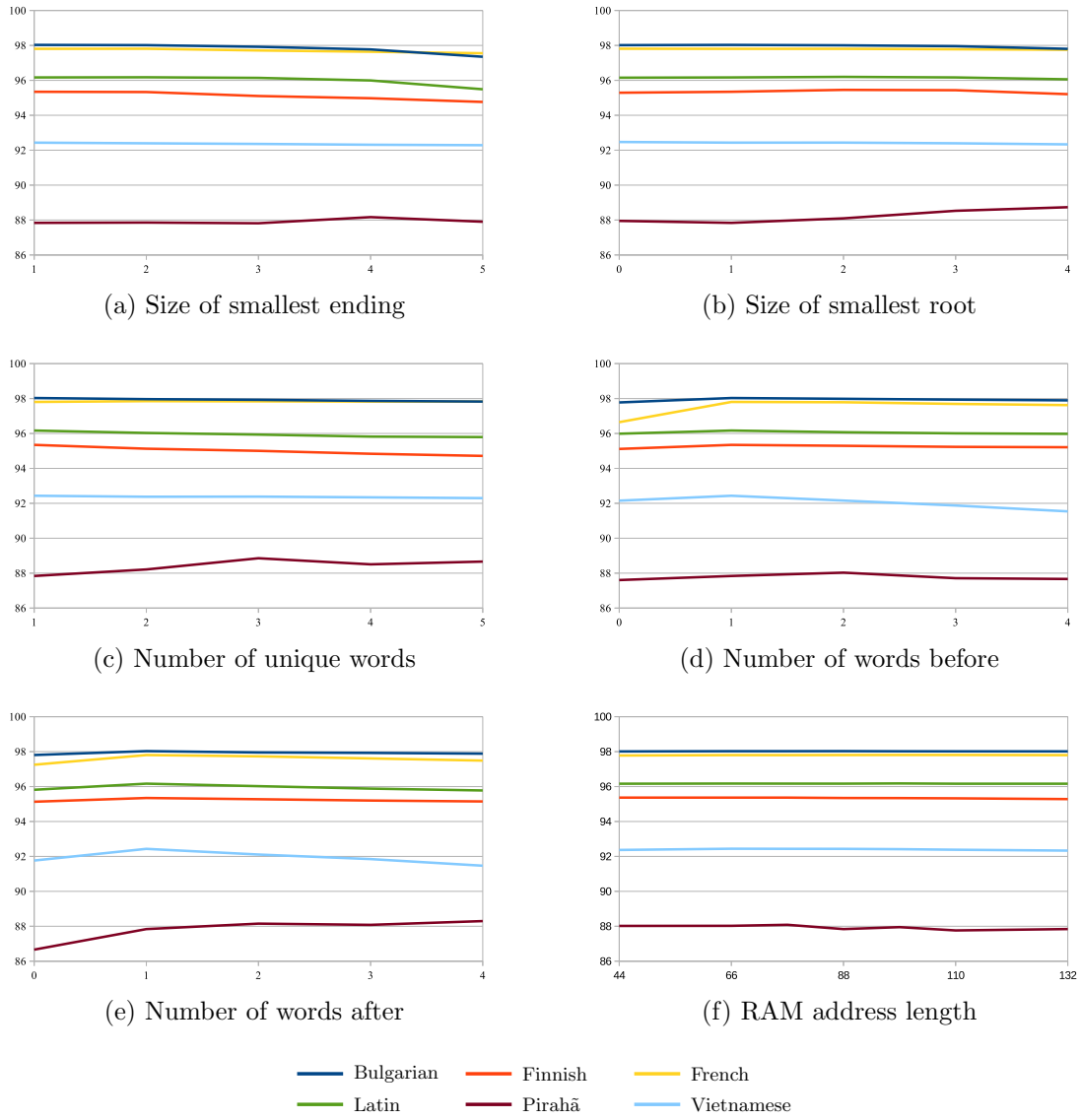


Figure 3.9: Sensitivity analysis for the new corpora. Graphs depict the accuracy variation caused by fluctuations on parameter values.

3.4.3 Comparison with the State of the Art

To ensure the multilingual part-of-speech tagging capability of the universal instance of mWANN-Tagger, it was subjected to every treebank of Universal Dependencies (UD) project [22] (version 1.4). Each treebank of UD has a training, a development and a test dataset. For each treebank, mWANN-Tagger was trained on its training dataset and tested on the respective test dataset. The accuracies achieved by mWANN-Tagger were compared to those of state-of-the-art part-of-speech taggers. The ones chosen for this experiment were the Ripple Down Rules Part-of-Speech-Tagger (RDRPOSTagger) of NGUYEN *et al.* [2], Bidirectional Long Short-Term Memory Tagger (bi-LSTM) of PLANK *et al.* [4] and Yet Another Parser (YAP) of MORE and TSARFATY [3], for they had a sufficient amount of performance

results on UD treebanks. Comparisons were also made with MarMoT, the higher order conditional random field (CRF) morphological tagger of MÜLLER *et al.* [1], because it was the most accurate part-of-speech tagger found in literature.

Table 3.11: Comparison between part-of-speech accuracies (%) on UD treebanks. Columns mWANN – Suffix and mWANN – Ext. offer the accuracies of (suffix-based) mWANN-Tagger universal configuration and its extension, respectively. The remaining ones, labeled MM, RDR, YAP, bi-LSTM and MM, hold the results of MarMoT [1], RDRPOSTagger [2], YAP [3] and bi-LSTM [4]. Highlighted cells represent cases where no statistically significant difference was attested between the performances of mWANN-Tagger extension and MarMoT.

Treebank	**mWANN**		**MM**	**RDR**	**YAP**	**bi-LSTM**	
	Suffix	Ext.				$\vec{w}$	$\vec{w} + \vec{c}$
Ancient Greek	92.14	93.36	**93.98**	91.57	92.0	–	–
Ancient Greek – PROIEL	95.87	96.76	97.07	95.72	**97.3**	–	–
Arabic	93.66	94.76	95.82	94.41	95.9	95.48	**98.89**
Basque	91.77	94.12	94.67	92.43	94.8	88.00	**94.91**
Bulgarian	96.15	96.98	97.64	96.13	97.9	95.12	**98.25**
Catalan	95.84	97.11	**97.84**	96.52	97.7	–	–
Chinese	90.52	91.28	**92.12**	89.45	91.4	–	–
Coptic	89.23	91.57	**93.21**	–	–	–	–
Croatian	94.47	95.39	**95.83**	93.87	95.1	89.24	95.59
Czech	97.46	97.89	**98.32**	97.68	–	93.77	97.93
Czech – CAC	97.44	97.75	**98.28**	97.83	**98.3**	–	–
Czech – CLTT	97.69	97.37	**97.73**	97.01	95.8	–	–
Danish	92.42	93.85	95.56	93.47	95.4	91.96	**95.94**
Dutch	87.80	88.43	89.34	88.76	90.2	84.96	**92.07**
Dutch – LassySmall	94.28	95.35	**96.36**	94.36	95.6	–	–
English	92.11	93.03	94.43	92.70	94.2	92.10	**94.61**
English – LinES	94.35	94.69	**95.84**	94.39	95.0	–	–
Estonian	93.69	94.86	**95.26**	93.84	95.1	–	–
Finnish	92.37	94.65	**95.34**	92.24	95.0	87.95	95.18
Finnish – FTB	91.40	93.51	**94.15**	90.96	88.3	–	–
French	94.99	95.75	**96.44**	95.23	95.9	94.44	96.04
Galician	96.51	96.63	**97.16**	96.31	97.1	–	–
Galician – TreeGal	92.58	95.14	**95.96**	–	–	–	–
German	90.93	92.24	92.67	90.40	92.9	90.33	**93.11**
Gothic	94.47	**95.83**	95.70	93.85	95.3	–	–
Greek	97.00	97.72	97.65	96.86	**97.8**	–	–
Hebrew	93.07	94.08	94.95	93.52	**96.7**	93.97	95.92
Hindi	94.72	95.12	96.32	95.02	95.3	95.99	**96.64**
Hungarian	89.11	92.96	**94.57**	88.69	91.5	–	–
Indonesian	90.30	93.22	**93.63**	90.75	93.2	90.48	92.79
Irish	90.26	**91.42**	91.39	90.61	90.0	–	–
Italian	95.95	96.69	**97.69**	96.48	97.0	96.57	97.64
Japanese	89.53	**93.77**	93.08	–	–	–	–

Table 3.12: Comparison between part-of-speech accuracies (%) on UD treebanks. Columns mWANN – Suffix and mWANN – Ext. offer the accuracies of (suffix-based) mWANN-Tagger universal configuration and its extension, respectively. The remaining ones, labeled MM, RDR, YAP, bi-LSTM and MM, hold the results of MarMoT [1], RDRPOSTagger [2], YAP [3] and bi-LSTM [4]. Highlighted cells represent cases where no statistically significant difference was attested between the performances of mWANN-Tagger extension and MarMoT. (cont.)

Treebank	**mWANN**		**MM**	**RDR**	**YAP**	**bi-LSTM**	
	Suffix	Ext.				$\vec{w}$	$\vec{w}+\vec{c}$
Kazakh	82.25	**84.88**	84.72	79.22	–	–	–
Latin	90.42	92.76	**92.98**	90.40	89.7	–	–
Latin – ITTB	97.82	97.88	**98.60**	98.24	**98.6**	–	–
Latin – PROIEL	95.49	96.40	96.75	95.79	**96.9**	–	–
Latvian	87.04	89.37	**90.70**	86.35	88.0	–	–
Norwegian Bokmål	94.45	95.68	97.28	94.60	97.1	94.39	**97.77**
Old Church Slavonic	95.06	96.10	96.26	94.63	**96.5**	–	–
Persian	96.12	96.34	96.57	96.00	96.0	95.31	**96.89**
Polish	94.06	95.09	96.26	94.08	96.1	89.73	**96.62**
Portuguese	94.55	95.63	96.89	95.08	96.7	94.24	**97.48**
Portuguese – BR	94.45	95.94	**97.25**	95.09	–	–	–
Portuguese – Bosque	89.23	92.20	**93.80**	–	–	–	–
Romanian	95.37	96.13	**96.90**	94.52	96.1	–	–
Russian	92.11	94.02	**95.57**	–	**95.6**	–	–
Russian – SynTagRus	96.77	97.65	**98.32**	97.65	–	–	–
Sanskrit	62.29	**69.92**	65.68	–	–	–	–
Slovak	89.57	91.32	**93.85**	–	–	–	–
Slovenian	93.74	94.75	96.22	94.03	95.8	91.09	**97.78**
Slovenian – SST	90.51	91.26	**91.66**	91.16	89.5	–	–
Spanish	94.08	94.91	95.08	95.13	**95.5**	93.60	95.34
Spanish – AnCora	96.34	97.32	**98.17**	96.79	–	–	–
Swedish	94.55	95.05	96.25	94.40	95.4	93.32	**96.30**
Swedish – LinES	94.36	95.11	**95.83**	94.47	95.5	–	–
Tamil	85.02	86.38	**87.53**	82.09	85.6	–	–
Turkish	92.54	**94.20**	94.16	91.93	89.3	–	–
Ukrainian	80.52	**85.06**	80.52	–	–	–	–
Uyghur	81.09	82.43	**83.38**	–	–	–	–
Vietnamese	88.50	89.97	**90.09**	–	–	–	–

Tables 3.11 and 3.12 present the accuracies of the universal configuration of mWANN-Tagger at column mWANN – Suffix and of other part-of-speech taggers on UD treebanks at the remaining ones. The data for RDRPOSTagger do not appear in [2], but can be found at `http://github.com/datquocnguyen/RDRPOSTagger/blob/master/Models/UniPOS/Readme.md`. RDRPOSTagger and YAP [3] accuracies were obtained through experiments on release 1.3 of Universal Dependencies. The ones of bi-LSTM were achieved by tests on release 1.2 [4]. Several instances of bi-LSTM are introduced in [4]. Here bi-LSTM$_{\vec{w}}$ and bi-LSTM$_{\vec{w}+\vec{c}}$ are used. The

former was chosen for being the one that is most equivalent to mWANN-Tagger, as it uses only word knowledge. The latter is also employed in this comparison because it had the highest accuracy among all instances. PLANK *et al.* [4] also offer bi-LSTM instances that apply off-the-shelf Polyglot word embeddings [113] for accuracy improvement, but they are not used for comparison because none of the remaining taggers make use of external word vector representations. The accuracies for MarMoT were produced by training it with its default parameters on the UD train files and then using it on the corresponding test files.

Tables 3.11 and 3.12 show that the universal configuration of mWANN-Tagger performs worse than MarMot and bi-LSTM$_{\vec{w}+\vec{c}}$, although it is competitive to RDRPOSTagger and bi-LSTM$_{\vec{w}}$, and performs better than YAP in 8 treebanks. A possible explanation of this may lie on the fact that RDRPOSTagger and bi-LSTM$_{\vec{w}}$ rely solely on words, whereas the others employ additional information. Furthermore, MarMoT makes use of not only suffixes like mWANN-Tagger, but also prefixes.

An extension to mWANN-Tagger was then proposed, by allowing it to employ much more information than just the suffixes of words within a context window. Initially, given a sentence $\mathbf{w} = [w_1, w_2, \ldots, w_n]$, if one wanted to tag its i-th word, w_i, mWANN-Tagger used the empirical conditional probabilities of w_i being tagged t_i given σ_{i-1}, σ_i, σ_{i+1}, i.e., its suffix and the ones of its adjacent words. Its extension differed from mWANN-Tagger by using not only the empirical conditional probabilities concerning σ_{i-1}, σ_i and σ_{i+1}, but also π_{i-1}, π_i and π_{i+1}, the prefixes of those same words. It also applied the empirical conditional probabilities of tagging w_i as t_i given (t_{i-1}, π_i) and (t_{i-1}, σ_i), that is, w_i having σ_i as its suffix (or π_i as its prefix) and w_{i-1} being tagged t_{i-1}.

Column mWANN – Ext. of Tables 3.11 and 3.12 display how the addition of prefix information as well as the tag of the previous word enhanced mWANN-Tagger performance and how it came closer to that of MarMoT. A McNeman test was done to assess the statistical significance of its results when compared to those of MarMoT. The extension of mWANN-Tagger outperforms the state of the art in Sanskrit, Ukrainian and Japanese treebanks with statistical significance ($p < 0.05$ for Sanskrit and Ukrainian, and $p < 0.001$ for Japanese). In eleven other textual bases, there is no statistically significant difference between the performance of both models ($p > 0.05$), as it is displayed in Tables 3.11 and 3.12. It is interesting to note that the accuracy difference in Coptic is not statistically significant, despite it being greater than 1.5% and existing lower accuracy differences with statistical significance, e.g. Persian. This happens potentially due to the size of Coptic treebank, whose test dataset has only 427 tokens to be tagged, whereas the one of Persian has 16024 tokens. At last, this mWANN-Tagger extension does not employ character-level representation, such as bi-LSTM$_{\vec{w}+\vec{c}}$ and to a lesser extent MarMoT.

Such feature would eventually bring mWANN-Tagger even higher and thus it could get competitive with MarMoT even in the larger treebanks.

3.5 Discussion

This chapter presented a single universal parameter configuration for mWANN-Tagger, so that it could be applied to any corpora and achieve accuracies extremely close to the highest possible ones. It was subjected to several corpora of very different languages, from isolating to polysynthetic ones. Sensitivity analyses indicated that mWANN-Tagger obtains very satisfactory accuracies when using that single parameter configuration for any corpora, especially those in predominantly suffixing languages. Experiments with state-of-the-art part-of-speech taggers also showed that mWANN-Tagger has some room for improvement, as including additional features to the tagger input raised its accuracy on every UD treebank. In particular, mWANN-Tagger might also benefit from better word representations, as a means to aid it in disambiguating tags even further and improving its accuracy.

Two additional points could be considered to further advance the research with mWANN-Tagger. They are:

1. A most widely accepted universal tagset could be used instead of the one of Table 3.2, e.g., Google Universal Tagset [20]. However, by the time mWANN-Tagger was conceived, no widely accepted universal tagset existed. Also, changing mWANN-Tagger tagset would imply in redoing some key points of the research, making it take longer. The Google Universal Tagset could be a good candidate for an alternative tagset for mWANN-Tagger as it is already used in the UD project [22, 23], even though there were some changes in the tagset from its conception up to now. By the time this thesis was written (June 2017), Google Universal Tagset was lastly changed for the Universal Dependencies v2 (March 2017).

2. Another word representation could be employed for mWANN-Tagger. Character-level representations that use different parts of the word being tagged prove to produce pretty accurate taggers (see Section 3.4.3). mWANN-Tagger could also benefit from word representations that do not depend on the size of particular parts of the word, because languages use diverse orthographical rules and writing systems that work in varied ways, e.g., alphabets, abjads, abugidas, syllabaries, logographic and hybrid scripts. A more compact, but yet very informative, word representation is also pertinent for mWANN-Tagger. The tagger architecture can become very big by employing word representations that produce long chains of bits. As a consequence of such a

big system architecture, the training of the universal mWANN-Tagger instance was slightly slower than MarMoT's. Unfortunately, no further study was conducted on word representations. The comparisons made in Section 3.4.3 were performed after the universal mWANN-Tagger parameter configuration was assembled. Furthermore, much of the issues raised by the comparison displayed in Tables 3.11 and 3.12 could only be verified recently (close to June 2017), since most POS-taggers used there were published in 2016 and also because Universal Dependencies notably grew just close to 2016.

Chapter 4

VC Dimension of n-tuple Learning Models

Saturation was a common problem of WiSARD n-tuple classifier. As the recognition machine was fed with (especially if noisy) data, it started filling up every position of its memory nodes, deteriorating its capability of discriminating patterns. Among the studies intended to lessen the effects of saturation, it is worth mentioning the work of BRADSHAW [28], where lower and upper bounds for the VC dimension of WiSARD n-tuple classifier were calculated. Despite not finding an exact value for that measure, the research made by BRADSHAW [28] provided a fertile ground for further analyses on this field. Unfortunately, the solution to the saturation problem there proposed was not that successful, as it relied on convergence and had no guarantee that it would actually occur whatsoever.

In 2010 [9] an actual way of mitigating saturation was devised, called the bleaching technique. It allowed the network to be exposed to loads of data (noisy and unnoisy) and yet keep the fidedignity of its pattern discrimination capability. This improvement considerably enhanced both accuracy and precision of traditional WiSARD n-tuple classifier applications with no performance harm on their training procedures and just a small bit on the classification step [11, 13–16]. That technique granted WiSARD competitiveness with trending learning systems, by achieving high accuracies with low variance in experimental works. However, there was no theoretical background for bleaching up to this point.

4.1 Basic Concepts: Introduction to Learning Theory

Given a limited number of examples, learning algorithms look for a desired dependence that best explains them. VAPNIK [27] described the learning process

through three components, a generator $\mathcal{G}$, a supervisor $\mathcal{S}$ and a learning machine $\mathcal{L}$. $\mathcal{G}$ randomly generates input vectors $\mathbf{x} \in \mathcal{X}$, where $\mathcal{X}$ is the input space. These vectors are drawn by an unknown probability distribution $P(\mathbf{x})$. They are presented to supervisor $\mathcal{S}$, which returns a label $y \in \mathcal{Y}$, where $\mathcal{Y}$ is the system output space. The label production of $\mathcal{S}$ is defined by the conditional probability distribution $P(y|\mathbf{x})$, also unknown. The learning machine $\mathcal{L}$ implements a set of functions $\mathcal{H} = \{h(\mathbf{x};\theta) : \theta \in \Theta\}$, where Θ is the set of parameters of the learning machine and h is a function $h : \mathcal{X} \times \Theta \to \mathcal{Y}$. Given a training dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\}$ of length ℓ drawn according to $P(\mathbf{x}, y) = P(\mathbf{x})\,P(y|\mathbf{x})$, the learning machine then must choose from the functions in $\mathcal{H}$ the one that best approximates the supervisor's response.

4.1.1 Risk Minimization

For that choice to produce the function that works as the best approximation to the supervisor's response, one should use of a **loss function** $V(y, h(\mathbf{x}, \theta))$ to represent the discrepancy between the supervisor's response y to input $\mathbf{x}$ and the output $h(\mathbf{x}, \theta)$ yielded by the learning machine. The objective of the learning machine is to find the $\theta^* \in \Theta$ that minimizes the expected value of this loss, given by the **risk functional**

$$R(\theta) = \mathbb{E}[V(y, h(\mathbf{x}, \theta))] = \int V(y, h(\mathbf{x}, \theta))\; dP(\mathbf{x}, y)\,. \tag{4.1}$$

Since $P(\mathbf{x}, y)$ is unknown, there is no straightforward way to minimize Equation 4.1. However, this procedure can be done by using training dataset $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\}$ instead and substituting $R(\theta)$ by the **empirical risk functional** $R_{emp}(\theta)$Empirical risk functional

$$R_{emp}(\theta) = \frac{1}{\ell} \sum_{i=1}^{\ell} V(y_i, h(\mathbf{x}_i, \theta))\,. \tag{4.2}$$

This way, one may find the $\theta^*_\ell \in \Theta$ that minimizes $R_{emp}(\theta)$, which should work as an approximation to the $\theta^* \in \Theta$ that minimizes $R(\theta)$. This principle is called **empirical risk minimization (ERM principle)**.

4.1.2 Bounding the Risk

VAPNIK and CHERVONENKIS [26] showed that the risk functional is bounded by its empirical form plus a quantity derived from the learning machine itself. This quantity depends on some keys concepts that need to be defined, e.g., dichotomies.

Let X be a set of data points to be classified and Θ a set of parameterizations of a learning machine $\mathcal{L}$. $\mathcal{L}$ classifies data points $\mathbf{x} \in X$ with labels $y \in \{-1, 1\}$ according to a parameter vector $\theta \in \Theta$. In other words, a learning system $\mathcal{L}$ can be seen as a function $g : X \times \Theta \to \{-1, 1\}$.

Definiton 1. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\ell \in X$. The set of **dichotomies** $\mathcal{L}$ can realize on X is defined as

$$\Delta_\Theta(X) = \{(g(\mathbf{x}_1; \theta), g(\mathbf{x}_2; \theta), \ldots, g(\mathbf{x}_\ell; \theta)) : \theta \in \Theta\}. \tag{4.3}$$

That is, the dichotomies are the distinct forms a learning machine can split a set of points into two, each one assigned to a different class. For instance, Figure 4.1 presents all eight dichotomies a linear separator can make on a set of three points.

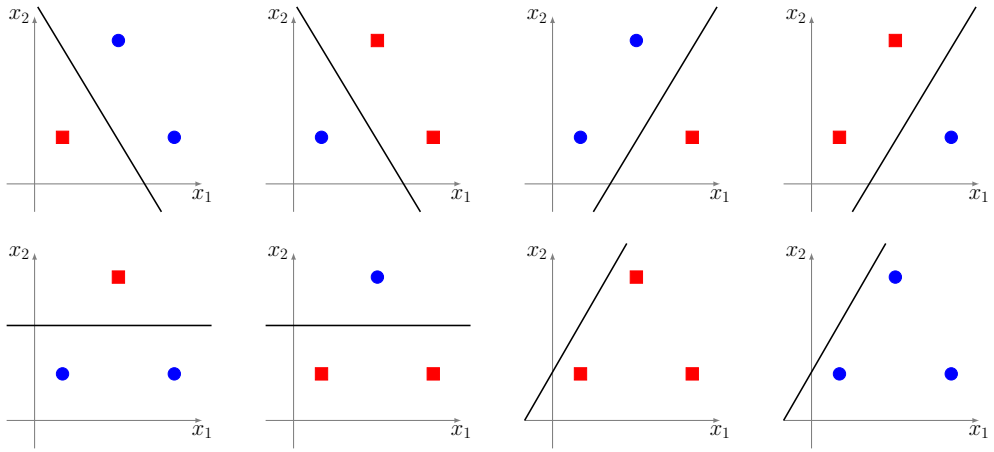


Figure 4.1: A linear separator shatters a set with three points

The amount of dichotomies a learning system $\mathcal{L}$ realizes on a set X plays an important role in VC dimension study. The **growth function** and the concept of **shattering** derive from that quantity.

Definiton 2. A set X is said to be **shattered** by a learning system $\mathcal{L}$ if $\text{card}(\Delta_\Theta(X)) = 2^{\text{card}(X)}$.

In other words, shattering means every subset of X can be separated from the rest. X is shattered by $\mathcal{L}$ if $\mathcal{L}$ can implement all possible dichotomies of X. Figures 4.1 and 4.2 shows that the linear separator can shatter a set with three points, but not one with four points. There is no single line one can draw that splits the set depicted in Figure 4.2 according to the colors there displayed.

Definiton 3. The **growth function** $\Pi_\Theta(\ell) : \mathbb{N} \to \mathbb{N}$ is defined as the maximum number of dichotomies $\mathcal{L}$ can implement on samples of size ℓ, i.e., the growth function is defined as

$$\Pi_\Theta(\ell) = \max_{X:\text{card}(X)=\ell} \text{card}(\Delta_\Theta(X)). \tag{4.4}$$

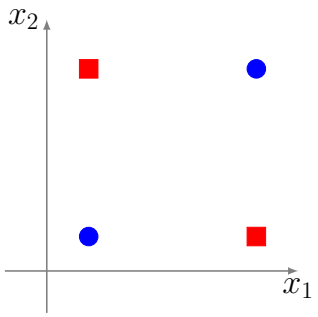


Figure 4.2: A linear separator does not shatter a set with four points

Definitions 2 and 3 imply that the growth function $\Pi_\Theta(\ell)$ is 2^ℓ if there is at least one set $X, \text{card}(X) = \ell$, that is shattered by $\mathcal{L}$. Note that if $\mathcal{L}$ shatters a set X, it also does every subset $X' \subset X$. So, if $\Pi_\Theta(\ell) = 2^\ell$, then $\Pi_\Theta(k) = 2^k$ for every $k \leq \ell$. However, the growth function may have a non-exponential nature if ℓ is large enough, so that there is no set $\mathcal{L}$ can shatter. The largest value of ℓ for which the growth function $\Pi_\Theta(\ell)$ is 2^ℓ is called the **VC dimension** of learning machine $\mathcal{L}$ [26, 27, 114].

Definiton 4. The **VC dimension** d_{VC} of learning machine $\mathcal{L}$ is defined as the cardinality of the largest set X that is shattered by $\mathcal{L}$. If for every ℓ, $\mathcal{L}$ shatters a set of such cardinality, then $d_{VC}(\mathcal{L})$ is infinite.

So, the linear separator used as a learning machine example has a VC dimension $d_{VC} = 3$, because it shatters a set with three points, but does not a set with four points. In the next subsections this same capacity measure is determined for the traditional WiSARD n-tuple classifier and for the min-max bleaching n-tuple classifier.

The growth function is a key part of the upper bound given by VAPNIK and CHERVONENKIS [26]. For a given error ε, inequality

$$P\left(\sup_{\theta\in\Theta} |R(\theta) - R_{emp}(\theta)| > \varepsilon\right) \leq 4\Pi_\Theta(2\ell)\, e^{-\frac{\ell\varepsilon^2}{8}} \tag{4.5}$$

holds true. In other words, given a tolerance level

$$\eta = 4\Pi_\Theta(2\ell)\, e^{-\frac{\ell\varepsilon^2}{8}}, \tag{4.6}$$

$0 < \eta < 1$, the risk functional is bounded above by

$$R(\theta) \leq R_{emp} + \varepsilon = R_{emp}(\theta) + \sqrt{\frac{8}{\ell} \ln \frac{4\Pi_\Theta(2\ell)}{\eta}} \tag{4.7}$$

with probability $1 - \eta$.

Except for a few trivial learning machines, the growth function is not easily calculated. However, it can be represented by an upper bound that shows that it is strongly constrained by the VC dimension of the learning machine [115, 116]. For a given learning machine $\mathcal{L}$ with VC dimension d_{VC}, its growth function is bounded above by

$$\Pi_\Theta(\ell) \begin{cases} = 2^\ell & , \text{if } \ell \leq d_{VC} \\ \leq \sum_{i=0}^{d_{VC}} \binom{\ell}{i} & , \text{if } \ell > d_{VC} \end{cases} . \tag{4.8}$$

And because

$$\sum_{i=0}^{d_{VC}} \binom{\ell}{i} \leq \left(\frac{\ell}{d_{VC}}\right)^{d_{VC}} \sum_{i=0}^{d_{VC}} \binom{\ell}{i} \left(\frac{d_{VC}}{\ell}\right)^i \leq \left(\frac{\ell}{d_{VC}}\right)^{d_{VC}} \left(1 + \frac{d_{VC}}{\ell}\right)^\ell < \left(\frac{e\ell}{d_{VC}}\right)^{d_{VC}}, \tag{4.9}$$

then Equation 4.8 can be rewritten as

$$\Pi_\Theta(\ell) \begin{cases} = 2^\ell & , \text{if } \ell \leq d_{VC} \\ < \left(\frac{e\ell}{d_{VC}}\right)^{d_{VC}} & , \text{if } \ell > d_{VC} \end{cases} . \tag{4.10}$$

Equation 4.5 indicates that $R_{emp}(\theta)$ converges to $R(\theta)$ when $\ell \to \infty$ when $\Pi_\Theta(2k)$ is polynomial. This way, $e^{-\frac{\ell\varepsilon^2}{8}}$ tends to 0 faster than the growth function reaches infinity. VAPNIK [27] called it that the ERM principle is **consistent**. Through Equation 4.10, it can be deduced that this consistency is achieved if the VC dimension is finite. BRADSHAW [28] argued that the ERM method is always consistent for finite sets of functions, such as those produced by the traditional WiSARD n-tuple classifier. However, the same cannot be affirmed for the bleaching variant, for its memory positions can store any integer value, and so, the model can produce an infinite set of functions. In light of this, it is important to discover if the VC dimension of bleaching n-tuple classifier is finite.

4.2 Determination of the VC Dimension of WiSARD n-tuple Classifier

Studies on the VC dimension of the traditional WiSARD n-tuple classifier were first conducted in [28]. The work intended to find the generalization bounds on this learning system, as a means to allow potential comparisons with other machine learning models, as well as to find ways to improve its generalization and mitigate the saturation problem.

BRADSHAW [28] achieved an exact value for the VC dimension of the maximal-discriminator n-tuple classifier, that is a network with a single discriminator, which accepts an input if its score is maximum, and rejects otherwise. BRADSHAW [28]

found $d_{VC} = N\,(2^n - 1)$ for this model, where d_{VC} is the VC dimension, and N and n are the amount of nodes and the addressing tuple length, respectively.

Nevertheless, no exact value was found for the VC dimension of the two-discriminator n-tuple classifier. The studies of BRADSHAW [28] fixed lower and upper bounds for this dimension, asserting that it should be $N\,(2^n - 1) \leq d_{VC} \leq \log_2 3 \cdot N2^n$. Lastly, BRADSHAW [28] suggested as a conjecture that an exact value for the VC dimension of the two-discriminator architecture is attainable, and that it is $d_{VC} = N\,(2^n - 1)$.

4.2.1 Preliminaries

It is important to raise some points before entering the proof itself. Firstly, only n-tuple classifiers with two discriminators are considered, despite the multiclass potential of the model. This is done because the VC dimension is defined upon the idea of dichotomies (see Section 4.1.2). Secondly, the addressing function is pseudorandomly generated at the network setup and it is not changed afterwards. Inserting new knowledge to the network does not affect its pseudorandom mapping. So, that mapping function does not count as an effective parameter of the n-tuple classifying learning model, not playing any role on its learning capacity. This way, every time an input observation should be considered, its equivalent feature vector is used instead. It brings the benefit of easing the comprehension of the proof without jeopardizing it. Thirdly, an arbitrary class must be chosen in the case of a draw. This work uses 1 as the opted class. Lastly, a sketch of the proof and a compilation of the needed notation is presented in this section on an attempt to aid the reader.

Sketch of Proof

The proof of the VC dimension is divided in two parts, the proof of its lower bound and the one of its upper bound. A flowchart is provided in Figure 4.3 to aid the reader in following the idea of the proof. To prove the first part, it is defined a set of points whose cardinality is the intended lower bound. Then, it is shown that for each possible class attribution there is one n-tuple classifier capable of classifying that set of points accordingly. This proves that the n-tuple classifying model can shatter that set of points, concluding this part of the proof. To prove the upper bound, a system of linear inequalities is produced. This system is based on the score measure of Equation 2.4. It should represent the classification of a generic set of points, whose cardinality is higher than the intended upper bound, into an also generic set of classes. Next, it is proved that there is no solution to such system of inequalities. So, there is at least one set of classes in which there is no way to classify a set of points that large, and thus this set is not shattered. This concludes

the proof of the upper bound. Given that the inner and upper bounds are identical, the VC dimension has an exact value.

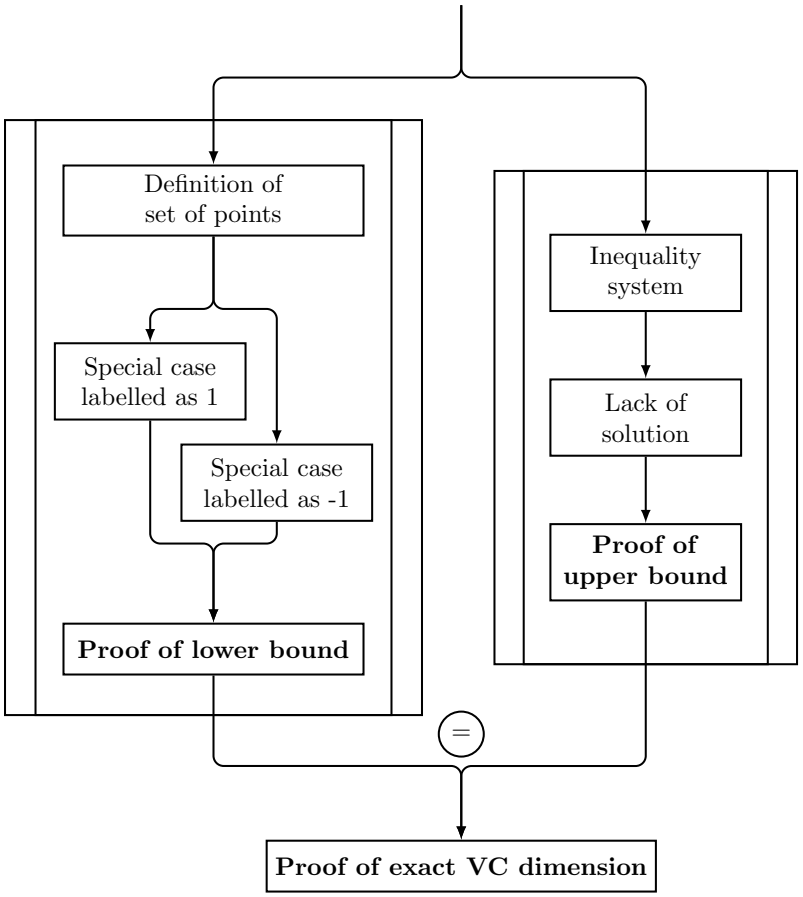


Figure 4.3: Proof flowchart for determination of the VC dimension of traditional WiSARD n-tuple classifier.

Notation

The VC dimension is a measure used for systems capable of disambiguating between two classes, i.e. functions that partition a set of points into two subsets, each one assigned to a particular class. Let $C = \{-1, 1\}$ be the class set employed for the proof. A generic class of this set is herein denoted by c, $c \in \{-1, 1\}$. Let N and $n \in \mathbb{N}$ stand for the number of nodes of an n-tuple classifier and the length of their addressing tuples. The family of functions that represent the N-node n-tuple classifying models is denoted by $\mathcal{W}_{N,n}$. It supersedes the notation of $\mathcal{W}_{N,n,C}$ introduced in Section 2.1.4 for set $C = \{-1, 1\}$.

Every n-tuple classifier of $\mathcal{W}_{N,n}$ has two discriminators, $\mathcal{D}_{-1}$ and $\mathcal{D}_1$. Their learned knowledge is represented by memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1 \in \{0, 1\}^{N \times 2^n}$. Henceforth $\mathcal{D}_c$ and $\mathbf{M}_c$ are used to represent discriminators and memory matrices of class c. Each discriminator $\mathcal{D}_c$ is composed of N nodes $d_{c,i}$, $i \in \{1, \ldots, N\}$. Each node $d_{c,i}$ has 2^n addressable positions and can store either 0 or 1. The content of

the j-th position of $d_{c,i}$ is represented by $m_{c,i,j}$, the element of i-th row and j-th column of $\mathbf{M}_c$ ($m_{-1,i,j}$ for $\mathbf{M}_{-1}$ and $m_{1,i,j}$ for $\mathbf{M}_1$).

Let X be an ordered set of input data points. Its k-th element is denoted $\mathbf{x}_k \in \{0,1\}^{Nn}$. Each input data point $\mathbf{x}_k$ can also be characterized by an addressing nomenclature. For the sake of readability, $\mathbf{x}_k$ can be rewritten as $\mathbf{x}_{k,1} : \mathbf{x}_{k,2} : \ldots : \mathbf{x}_{k,N}$, where $\mathbf{x}_{k,i}$ represents the i-th n-tuple of $\mathbf{x}_k$, which addresses every memory node $d_{c,i}$. These n-tuples can be 0^n, an all-zero n-tuple, 1^n, an all-one one, or $\{0,1\}^n$, a generic binary one.

Let Z be an ordered set of the respective feature matrices produced by the addressing function $A : \{0,1\}^{Nn} \to \{0,1\}^{N\times 2^n}$ that describes the pseudorandom mapping of that WiSARD learning machine. The k-th element of Z is denoted $\mathbf{Z}_k \in \{0,1\}^{N\times 2^n}$. Its i-th row, $\mathbf{z}_{k,i}$, characterizes which position should be addressed in nodes $d_{-1,i}$ and $d_{1,i}$. $z_{k,i,j}$, its j-th element, is 1 if the j-th positions of those nodes are addressed, and 0 otherwise. Also, let $\mathbf{y}$ be a generic vector of classes. It should have as many elements as Z, so that $y_k \in \{-1,1\}$, k-th element of $\mathbf{y}$, be the expected class of $\mathbf{Z}_k$.

Let $W_{N,n}(\cdot;\mathbf{M}_{-1},\mathbf{M}_1) \in \mathcal{W}_{N,n}$ be the function that represents the N-node n-tuple classifier defined by memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1$. It is a function $W_{N,n} : \{0,1\}^{N\times 2^n} \to \{-1,1\}$, which given a feature matrix $\mathbf{Z}_k$ returns a class $c \in \{-1,1\}$. The return of $W_{N,n}(\mathbf{Z}_k;\mathbf{M}_{-1},\mathbf{M}_1)$ is obtained through scores $s_{-1}(\mathbf{Z}_k)$ and $s_1(\mathbf{Z}_k)$, which are defined according to Equation 2.4. These scores can be organized in a score vector $\mathbf{s}(\mathbf{Z}_k) = [s_{-1}(\mathbf{Z}_k), s_1(\mathbf{Z}_k)]^{\mathrm{T}}$. The class chosen by $W_{N,n}(\mathbf{Z}_k;\mathbf{M}_{-1},\mathbf{M}_1)$ is the one whose score is the greatest.

Part of the notation presented in this subsection was introduced in Section 2.1.4 and was herein revisited. The notation is condensed in Table 4.1 for future reference.

4.2.2 The Lower Bound

As explained in Section 4.2.1, the proof of the lower bound of the VC dimension is attained by construction, where a set of points X is presented and there are n-tuple classifiers that can categorize it in every $2^{\mathrm{card}(X)}$ possible ways. X must be such that its cardinality is equal to the intended lower bound for the VC dimension.

Lemma 1. *The VC dimension, d_{VC}, of a model $\mathcal{W}_{N,n}$, as defined in Section 4.2.1, is bounded below by $N(2^n - 1) + 1$.*

Table 4.1: Notation for traditional n-tuple classifier.

N	Number of nodes
n	Addressing tuple length
$\mathcal{W}_{N,n}$	N-node n-tuple classifying model
c	Generic class (-1 or 1)
$\mathbf{M}_{-1}$	Memory matrix of class -1
$\mathbf{M}_1$	Memory matrix of class 1
$\mathbf{M}_c$	Memory matrix of class c
$m_{c,i,j}$	Element at i-th row and j-th column of $\mathbf{M}_c$
$W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$	N-node n-tuple classifier with memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1$
$\mathcal{D}_{-1}$	Discriminator for class -1
$\mathcal{D}_1$	Discriminator for class 1
$\mathcal{D}_c$	Discriminator for class c
$d_{c,i}$	i-th memory node of $\mathcal{D}_c$
X	Ordered set of input data points
$\mathbf{x}_k$	k-th element of X
$\mathbf{x}_{k,1} : \ldots : \mathbf{x}_{k,N}$	Addressing representation of $\mathbf{x}_k$
$\mathbf{x}_{k,i}$	i-th n-tuple of $\mathbf{x}_k$, which addresses $d_{c,i}$
$\{0,1\}^n$	Generic binary n-tuple
0^n	All-zero n-tuple
1^n	All-one n-tuple
Z	Ordered set of feature matrices
$\mathbf{Z}_k$	k-th element of Z
$\mathbf{z}_{k,i}$	i-th row of $\mathbf{Z}_k$
$z_{k,i,j}$	j-th element of $\mathbf{z}_{k,i}$
$\mathbf{y}$	Vector of classes, one for each element of Z
y_k	k-th element of $\mathbf{y}$, expected class of $\mathbf{Z}_k$
$s_{-1}(\mathbf{Z}_k)$	Score of $\mathcal{D}_{-1}$ for $\mathbf{Z}_k$
$s_1(\mathbf{Z}_k)$	Score of $\mathcal{D}_1$ for $\mathbf{Z}_k$
$\mathbf{s}(\mathbf{Z}_k)$	Score vector $\mathbf{s}(\mathbf{Z}_k) = [s_{-1}(\mathbf{Z}_k), s_1(\mathbf{Z}_k)]^{\mathrm{T}}$

Proof. Let X be an ordered set of input data points defined as

$$\begin{aligned} X = & \overbrace{\{0,1\}^n : 0^n : \ldots : 0^n}^{N} \cup \\ & 0^n : \{0,1\}^n : \ldots : 0^n \cup \\ & \ldots \cup \\ & 0^n : 0^n : \ldots : \{0,1\}^n. \end{aligned} \tag{4.11}$$

To verify if $\mathcal{W}_{N,n}$ shatters X, two particular cases should be analyzed: *a*) when

$0^n : 0^n : \ldots : 0^n$ is classified as 1; and b) when it is classified as -1.

To ease the understanding of this proof, an example is provided for each particular case. Both examples describe dichotomies a $\mathcal{W}_{2,2}$ n-tuple classifying model (i.e. $N = 2$ and $n = 2$) can perform on a set of input points. By applying $N = 2$ and $n = 2$ to set X, displayed in Equation 4.11, one gets $X_{2,2} = \{00{:}00,\ 01{:}00,\ 10{:}00,\ 11{:}00,\ 00{:}01,\ 00{:}10,\ 00{:}11\}$. It is not possible to consider every 2^7 dichotomies $\mathcal{W}_{2,2}$ can perform on $X_{2,2}$, but two of them will be described, one for each particular case. Also, a way to generalize them to every other possible dichotomy, and for all values of N and n, is provided as well. In both dichotomies, $10:00$, $00:10$ and $00:11$ are to be classified as 1, and $01:00$, $11:00$ and $00:01$ as -1. Input point $00:00$ is going to be classified according to the description of each particular case. For a quick reference, these dichotomies are disposed in Table 4.2.

Table 4.2: Dichotomies of $X_{2,2}$

Classified as		**Particular case**
1	-1	
10:00	01:00	00:00
00:10	11:00	
00:11	00:01	

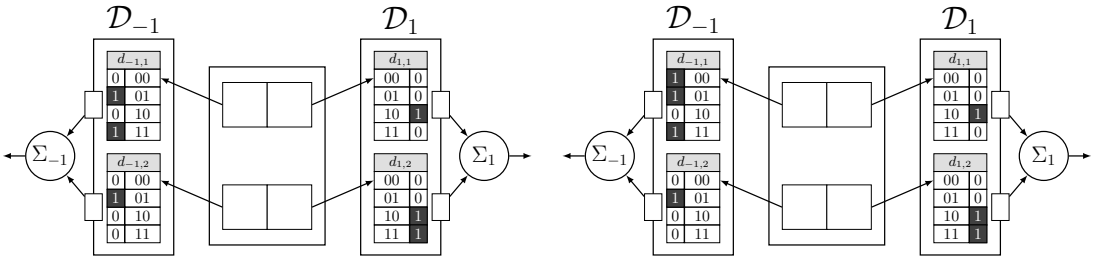
It is worth noting that X is the set with every input point with at most one non-null addressing n-tuple. Its first line refers to all input points whose first addressing n-tuple can be any binary sequence, but the remaining ones must be all-zero n-tuples. Its second line refers the input points whose second n-tuple can be any sequence of binary values and the remaining ones must be composed only by 0s. The same pattern applies to every other line of X. This is reflected on $X_{2,2}$, which has elements $00:00$, $01:00$, $10:00$ and $11:00$ corresponding to the first line of X, and $00:00$, $00:01$, $00:10$ and $00:11$ for another line. Note that $00:00$ repeats in both groups of input points, but it is only shown once in $X_{2,2}$, for it is a set.

The main intuition behind the choice for X lies in that fact that it is the largest set of input points that does not allow an undesirable property. It does not have a subset of four points, where each of them differs from two of the other points by a single n-tuple, e.g., the set composed by $00:00$, $01:00$, $01:01$ and $00:01$. Such set of points, or any other that contains it, cannot be shattered. There is no WiSARD n-tuple classifier that can classify $00:00$ and $01:01$ as 1, and $01:00$ and $00:01$ as -1. This scenario is analogous to the one depicted in Figure 4.2, where there were no line that could be drawn in that plane, splitting it into two regions, such that all red squares would lie in one region and all blue circles in the other.

$0^n : 0^n : \ldots : 0^n$ **classified as 1:** Given a class vector $\mathbf{y} = [y_1, y_2, \ldots]^{\mathrm{T}}$, let $\mathbf{M}_{-1}$ and $\mathbf{M}_1$ be memory matrices, whose elements $m_{c,i,j}$ are given by

$$m_{c,i,j} = \begin{cases} 1 & \text{, if } \mathbf{x}_{k,i} \neq 0^n \text{ addresses } j\text{-th position of } d_{c,i} \text{ and } y_k = c \\ 0 & \text{, otherwise} \end{cases} .$$

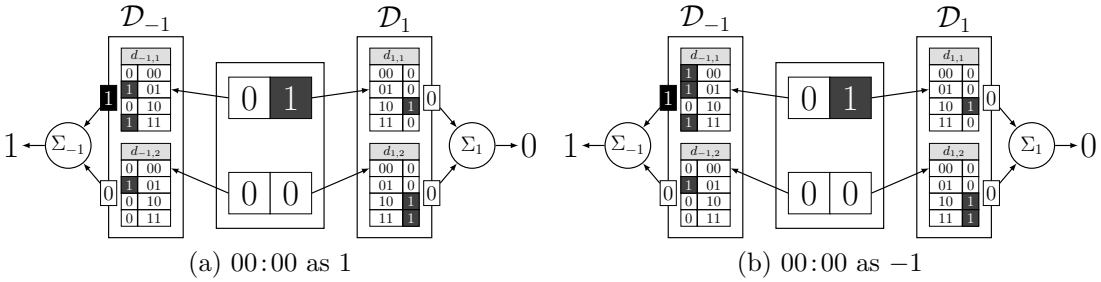
This filling of the memory positions of the WiSARD n-tuple classifier can be noted in Figure 4.4a, where position 10 of $d_{1,1}$, and positions 10 and 11 of $d_{1,2}$ store the value 1. These positions are respectively the ones addressed by the non-null n-tuples of $10:00$, $00:10$ and $00:11$, the input points to be classified as 1. Analogously, positions 01 and 11 of $d_{-1,1}$ and position 01 of $d_{-1,2}$ also store 1 because they are addressed by the non-null n-tuples of the input points to be classified as -1.



(a) 00:00 as 1 (b) 00:00 as -1

Figure 4.4: WiSARD memory position fillings

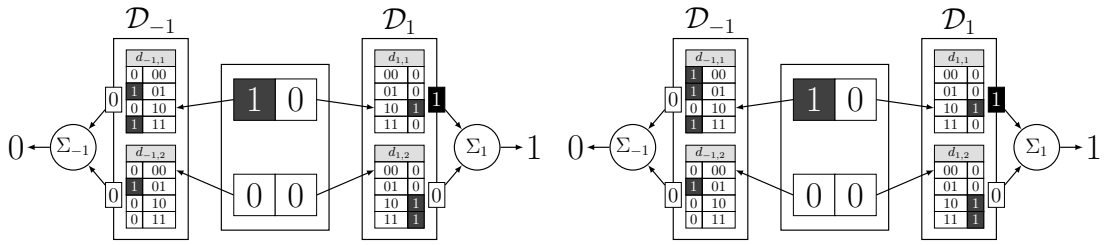
Let there be an n-tuple classifier $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1) \in \mathcal{W}_{N,n}$. If an entry $\mathbf{x}_k \in X \setminus \{0^n : 0^n : \ldots : 0^n\}$ is presented to the classifier, it yields a score vector $\mathbf{s} = [1, 0]^{\mathrm{T}}$ if $y_k = -1$, or $\mathbf{s} = [0, 1]^{\mathrm{T}}$ if $y_k = 1$. This happens because only one non-null position is addressed. This can be confirmed by checking Figures 4.5a, 4.6a, 4.7a and 4.8a. It is easy to verify that the same result occurs for every other input point of Table 4.2 that is not $00:00$. Also, this result can be reproduced even if given higher values of N or n, or any other dichotomy that is considered by the particular case.



(a) 00:00 as 1 (b) 00:00 as -1

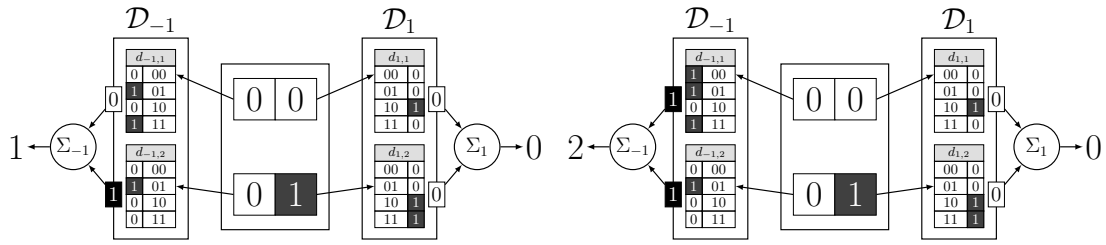
Figure 4.5: Classification of $01:00$

If $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ receives $\mathbf{x}_k = \{0^n : 0^n : \ldots : 0^n\}$, its score vector is $\mathbf{s} = [0, 0]^{\mathrm{T}}$, and thus the network opts for class 1, as expected, for it is a draw. Figure 4.9a



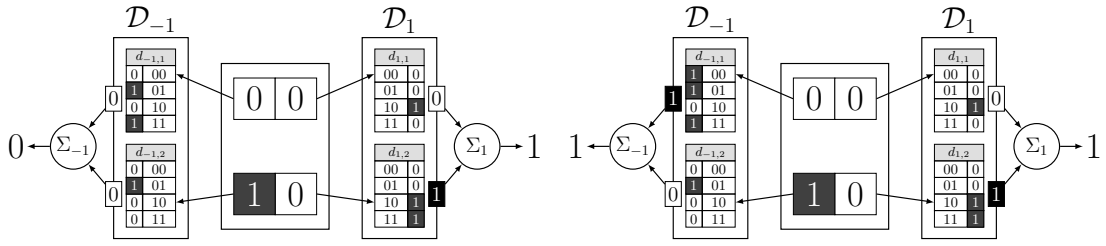
(a) 00:00 as 1 (b) 00:00 as −1

Figure 4.6: Classification of 10:00



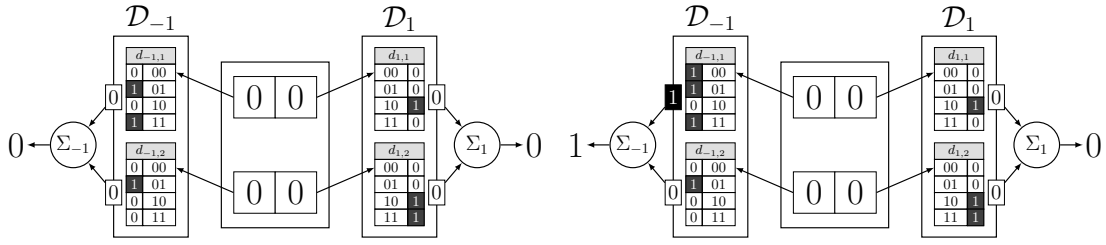
(a) 00:00 as 1 (b) 00:00 as −1

Figure 4.7: Classification of 00:01



(a) 00:00 as 1 (b) 00:00 as −1

Figure 4.8: Classification of 00:10



(a) 00:00 as 1 (b) 00:00 as −1

Figure 4.9: Classification of 00:00

shows how this classification proceeds in the example with $N = 2$ and $n = 2$ through the obtention of a tie between the network discriminators. One can visualize that a similar procedure happens for higher values of N and n.

In short, $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ accurately classifies any $\mathbf{x}_k \in X \setminus \{0^n : 0^n : \ldots : 0^n\}$ and also $0^n : 0^n : \ldots : 0^n$ if it is to be classified as 1.

$0^n : 0^n : \ldots : 0^n$ **classified as** -1**:** Given a class vector $\mathbf{y} = [y_1, y_2, \ldots]^{\mathrm{T}}$, let $\mathbf{M}_{-1}$ and $\mathbf{M}_1$ be memory matrices, whose elements $m_{c,i,j}$ assume the same values of the previous case. The only single exception being $m_{-1,1,1}$, which should always be equal to 1. This memory position filling is depicted in Figure 4.4b. The content of the network memory nodes are identical to those of Figure 4.4a, except for position 00 of $d_{-1,1}$, which stores 1 in Figure 4.4b, whereas it stores 0 in Figure 4.4a.

Let there be an n-tuple classifier $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1) \in \mathcal{W}_{N,n}$. If an entry $\mathbf{x}_k \in \{\{0,1\}^n : 0^n : \ldots : 0^n\}$ is presented to the classifier, it yields a score vector $\mathbf{s} = [1,0]^{\mathrm{T}}$ if $y_k = -1$, or $\mathbf{s} = [0,1]^{\mathrm{T}}$ if $y_k = 1$. A particular case being $\mathbf{x}_k = 0^n : 0^n : \ldots : 0^n$, where $\mathbf{s} = [1,0]^{\mathrm{T}}$. This happens because only one non-null position is addressed. Figures 4.5b and 4.6b show how their instance of the n-tuple classifying model produce the same scores, and consequently the same correct classification, of Figures 4.5a and 4.6a. This happens because the first addressing n-tuple is not 00, so position 00 of $d_{-1,1}$ is not addressed. Since the only difference between these n-tuple classifiers does not play any role in the classification of those input points, this classification procedure can be generalized, likewise that of Figures 4.5a and 4.6a. The particular case of $00:00$ is presented in Figure 4.9b. Unlike what happens in Figure 4.9a, this classification produces a different score for each discriminator. An identical score would favor class 1, but the scenario of Figure 4.9b shows a clear winning of class -1, as intended. Any higher value of N or n would keep the score of $\mathcal{D}_{-1}$ greater than that of $\mathcal{D}_1$. So, any point $0^n : 0^n : \ldots : 0^n$ would be correctly classified as -1 for every N and n.

If $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ receives $\mathbf{x}_k \in X \backslash \{\{0,1\}^n : 0^n : \ldots : 0^n\}$, two possible scenarios may occur: (*i*) if $y_k = -1$, then the network produces a score vector $\mathbf{s} = [2,0]^{\mathrm{T}}$, due to its non-all-zero addressing n-tuple and to $m_{-1,1,1}$; and (*ii*) if $y_k = 1$, the attained score is $\mathbf{s} = [1,1]^{\mathrm{T}}$, for the same reason. The first scenario is represented in Figure 4.7b, where the first n-tuple addresses position 00 of $d_{-1,1}$, and the other n-tuple addresses a position of $d_{-1,2}$ which stores 1. If n were higher a similar result would be produced. If a higher N were given, every other addressing n-tuple would be 00 and would address positions storing 0 in both discriminators. This way, the score vector $\mathbf{s} = [2,0]^{\mathrm{T}}$ would be obtained in this scenario for all possible values of N and n. If a different dichotomy were used, a different memory node filling would be generated, but it would lead to this same result. The second scenario is depicted in Figure 4.8b, where only position 00 of $d_{-1,1}$ contributes positively to the score of $\mathcal{D}_{-1}$. Likewise, the only memory node that produces a positive output for $\mathcal{D}_1$ is the one that also does it in Figure 4.6a. This scenario can be generalized for higher values of N and n, and also for other dichotomies, for the same reasons that apply to the previous one.

In short, $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ classifies any entry $\mathbf{x}_k \in \{\{0,1\}^n : 0^n : \ldots : 0^n\}$

accurately. Its score vector shows that the network also correctly does it for $\mathbf{x}_k \in X \setminus \{\{0,1\}^n : 0^n : \ldots : 0^n\}$ if $y_k = -1$. For the scenario where $y_k = 1$, its score vector points to a draw, which properly leads the classifier to the expected class, $y_k = 1$.

So, there exists at least one n-tuple classifier $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ capable of classifying the set X in every possible class vector $\mathbf{y}$ for both cases ($0^n : 0^n : \ldots : 0^n$ classified as either 1 or -1). Since each of those cases comprehend half of every $2^{\text{card}(X)}$ possible classifications and their intersection is empty, there exists at least one n-tuple classifier $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ capable of classifying X in all $2^{\text{card}(X)}$ possible manners. As a direct consequence, $\mathcal{W}_{N,n}$ shatters X and the VC dimension of this model is bounded below by $\text{card}(X) = N2^n - (N-1) = N(2^n - 1) + 1$. It is worth noting that this result proves wrong the conjecture proposed by BRADSHAW [28], which supposed the VC dimension of the traditional WiSARD n-tuple classifier would be slightly smaller. □

4.2.3 The Upper Bound

The VC dimension upper bound of an n-tuple classifying model is obtained through a linear algebra approach. In Section 4.2.1, it was advanced that a system of linear inequalities should be derived from the score formula, presented in Equation 2.4. The relation between the values of scores s_{-1} and s_1 plays a vital role on determining the suitable class for a given input, and this is the main intuition behind the development of that system of inequalities.

Proposition 1. *An n-tuple classifying model $\mathcal{W}_{N,n}$ shatters a set of feature matrices $Z, \text{card}(Z) = \ell$, if and only if given a generic class vector $\mathbf{y} \in \{-1,1\}^\ell$, there exists a set of memory matrix elements $m_{c,i,j} \in \{0,1\}$ that solves*

$$\sum_{i=1}^{N} \left(1 - \sum_{j=2}^{2^n} z_{k,i,j}\right)(m_{1,i,1} - m_{-1,i,1}) + \sum_{i=1}^{N}\sum_{j=2}^{2^n} z_{k,i,j}\,(m_{1,i,j} - m_{-1,i,j}) = y_k\mu_k - \frac{1}{2}, \tag{4.12}$$

for all $k \in \{1, \ldots, \ell\}$, where $\mu_k \in \left\{\frac{1}{2}, \frac{3}{2}, \ldots, \frac{2N+1}{2}\right\}$, and $z_{k,i,j}$ and y_k are defined as in Section 4.2.1.

Proof. Let ν_k be the difference between scores $s_1(\mathbf{Z}_k)$ and $s_{-1}(\mathbf{Z}_k)$. That is,

$$\begin{aligned} s_1(\mathbf{Z}_k) - s_{-1}(\mathbf{Z}_k) &= \nu_k \\ \sum_{i=1}^{N}\sum_{j=1}^{2^n} z_{k,i,j}\left(m_{1,i,j} - m_{-1,i,j}\right) &= \nu_k. \end{aligned} \tag{4.13}$$

ν_k can assume any integer value at interval $[-N, N]$. If it is negative, then it means that the classifier should choose class -1. Otherwise, it should choose 1 (even if there is a draw).

By defining a slack variable $\mu_k \in \left\{\frac{1}{2}, \frac{3}{2}, \ldots, \frac{2N+1}{2}\right\}$, one can rewrite ν_k as

$$\nu_k = y_k\mu_k - \frac{1}{2}. \tag{4.14}$$

This way, one gets the difference of the scores as a linear function of y_k. This can be checked by verifying that when $y_k = 1$, $\nu_k \in \{0, 1, \ldots, N\}$, i.e., any possible non-negative score difference between discriminators with N memory nodes. On the other hand, when $y_k = -1$, $\nu_k \in \{-1, -2, \ldots, -N-1\} \supset \{-1, -2, \ldots, -N\}$, i.e., any possible negative score difference between discriminators with N memory nodes. Although Identity 4.14 allows ν_k to be $-N-1$, it does not truly occur. It poses no problem nevertheless, as every value of $\nu_k \in [-N, N]$ could successfully be represented as a linear function of y_k, as it was intended.

As mentioned in Section 2.1.4, a feature matrix $\mathbf{Z}_k$ has the property of having a single element equal to 1 in each row. The remaining elements of $\mathbf{Z}_k$ are all equal to 0. Given this property, it is straightforward that any element of $\mathbf{z}_{k,i}$ can be defined as a function of the remaining elements through the identity

$$z_{k,i,q} = 1 - \sum_{\substack{j=1 \\ j\neq q}}^{2^n} z_{k,i,j}. \tag{4.15}$$

Combining Equations 4.13, 4.14 and 4.15 (for $q = 1$ without loss of generality),

$$\begin{aligned} \sum_{i=1}^{N}\left(1 - \sum_{j=2}^{2^n} z_{k,i,j}\right)\left(m_{1,i,1} - m_{-1,i,1}\right) + \\ \sum_{i=1}^{N}\sum_{j=2}^{2^n} z_{k,i,j}\left(m_{1,i,j} - m_{-1,i,j}\right) = y_k\mu_k - \frac{1}{2}. \end{aligned} \tag{4.16}$$

So, $\mathbf{M}_{-1}$ and $\mathbf{M}_1$ must be such that Equation 4.16 holds true for an n-tuple classifier $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ to correctly categorize a feature matrix $\mathbf{Z}_k$. As an immediate consequence, if there exist $\mathbf{M}_{-1}$ and $\mathbf{M}_1$ such that Equation 4.16 holds true

for every $k \in \{1, \ldots, \ell\}$, then $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ accurately assigns every feature matrix $\mathbf{Z}_k \in Z$ to its corresponding class $y_k \in \mathbf{y}$.

Finally, if those same memory matrices satisfy Equation 4.16 for every $k \in \{1, \ldots, \ell\}$ and for any generic class vector $\mathbf{y} \in \{-1, 1\}^\ell$, then for every 2^ℓ possible instances of $\mathbf{y}$, $W_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ can precisely classify every $\mathbf{Z}_k \in Z$. In other words, model $\mathcal{W}_{N,n}$ shatters Z. □

Next, it is intended to show that if the amount ℓ of equations of the present system (see Equation 4.12) is larger than a given upper bound, then there are no memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1$ that satisfy it for every possible set of classes $\mathbf{y} \in \{-1, 1\}^\ell$.

Proposition 2. *There exist no memory matrices* $\mathbf{M}_{-1}$ *and* $\mathbf{M}_1$ *for which system of Equations 4.12 holds true for a generic set of classes* $\mathbf{y} \in \{-1, 1\}^\ell$*, if the amount of equations* $\ell > N\left(2^n - 1\right) + 1$*.*

Proof. System of Equations 4.12 is linear, and so it can be expressed in the form $\mathbf{Ax} = \mathbf{b}$. According to Rouché-Capelli theorem [117], a system of linear equations has a solution if and only if the rank of its coefficient matrix $\mathbf{A}$ is equal to the rank of its augmented matrix $[\mathbf{A} \,|\, \mathbf{b}]$. Because many columns of the coefficient matrix of System 4.12 can be produced through linear combinations of others, eliminations must be done to determine its rank.

At first, the system of equations has $N2^{n+1}$ variables and so the rank of its coefficient matrix is at most that same value. Next, the coefficients of all variables $m_{-1,i,j}$ can be eliminated, for their columns can be combined to those of $m_{1,i,j}$. Further eliminations can be done, because

$$1 - \sum_{j=2}^{2^n} z_{k,i,j} = 1 - \sum_{j=2}^{2^n} z_{k,1,j} + \sum_{j=2}^{2^n} z_{k,1,j} - \sum_{j=2}^{2^n} z_{k,i,j}. \tag{4.17}$$

Identity 4.17 implies that the coefficient matrix columns related to $m_{1,i,1}, i \geq 2$ can be combined to those related to $m_{1,1,1}$, $m_{1,1,j}$ and $m_{1,i,j}$ $(j \geq 2)$. In summary, two significant elimination processes were made to determine that there are at most $N2^{n+1} - N2^n - (N-1) = N\left(2^n - 1\right) + 1$ linearly independent columns. Therefore, when the system of Equations 4.12 has more than $N\left(2^n - 1\right) + 1$ equations, the rank of its coefficient matrix is at most $N\left(2^n - 1\right) + 1$.

Because there is no prior linear relation between class vector $\mathbf{y}$ and the ordered set of feature matrices Z, then the rank of the augmented matrix is greater than the one of the coefficient matrix if their corresponding system have more than $N\left(2^n - 1\right) + 1$ equations. Hence, there is no solution for such system of linear equations. □

Due to results of Propositions 1 and 2, it is straightforward to determine an upper bound for the VC dimension of the traditional n-tuple classifier.

Lemma 2. *The VC dimension, d_{VC}, of a model $\mathcal{W}_{N,n}$, as defined in Section 4.2.1, is bounded above by $N(2^n - 1) + 1$.*

Proof. There is no solution for system of Equations 4.12 if it has at least $N(2^n - 1)+$ 2 equations. It implies that $\mathcal{W}_{N,n}$ does not shatter any set of feature matrices Z, $\text{card}(Z) = N(2^n - 1) + 2$. □

Theorem 1. *The VC dimension, d_{VC}, of a model $\mathcal{W}_{N,n}$, as defined in Section 4.2.1, is given by $N(2^n - 1) + 1$.*

Proof. The VC dimension of $\mathcal{W}_{N,n}$ is bounded above and below by $N(2^n - 1) + 1$. Then, it is exactly $N(2^n - 1) + 1$. □

4.3 Determination of the VC dimension of bleaching n-tuple classifier

The bleaching technique provided an advantage to the n-tuple classifier by mitigating its proneness to saturation. Results on experiments conducted with this architecture indicated a noticeable improvement to what could be achieved by the traditional learning model [9, 11, 13–16]. Historically, the n-tuple classifier accuracy tended to worsen if the amount of data to be learned by the system were large enough. The capability of learning from massive data without getting saturated showed that the classifier could achieve higher accuracies (with lower variance) as more data was presented to it.

The model, however, lacked a theoretical foundation. A study on its generalization capacity could bring a better comprehension on how the network works and also could provide insights on how it could be further improved.

4.3.1 Preliminaries

The proof of the VC dimension of the min-max bleaching n-tuple classifier is split into two parts, the proofs of its lower and upper bounds, as depicted in the flowchart of Figure 4.10. The lower bound is immediately obtained from the fact that the min-max bleaching n-tuple classifier is a generalization of traditional WiSARD.

To prove the upper bound, a new score measure must be defined. The discriminators yield scores that depend on the bleaching threshold. So, if a proof procedure were made and it were similar to the one done for the traditional model, the VC dimension of the bleaching architecture would be dependent on a fixed threshold

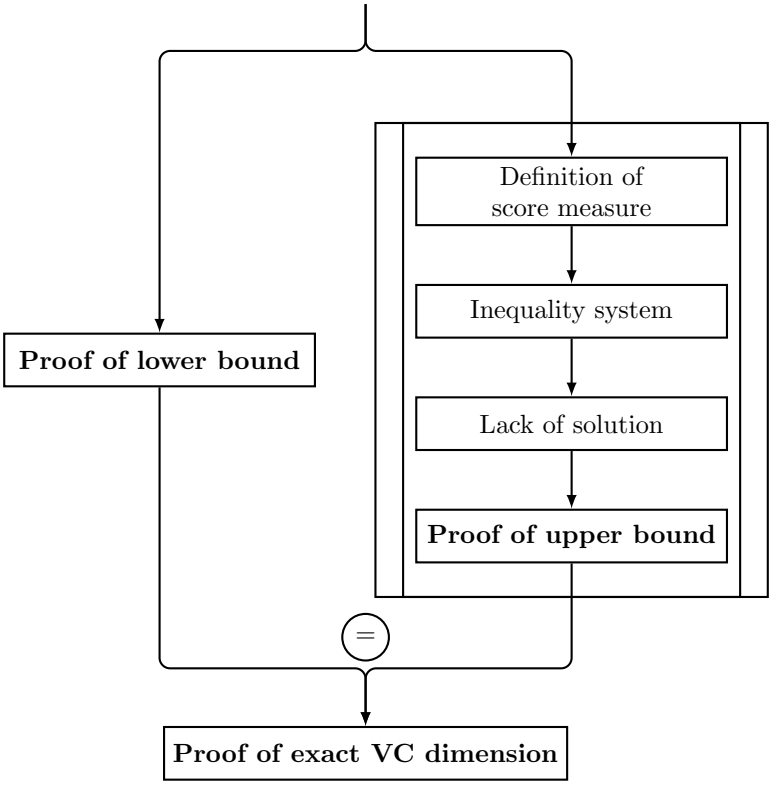


Figure 4.10: Proof flowchart for determination of the VC dimension of min-max bleaching n-tuple classifier.

value. But the classification process does not depend on this threshold, given its potential dynamic nature. That means, two distinct input patterns can achieve their classification with very different values of the bleaching threshold. So, it is imperative to have a score measure that does not depend on that threshold. Afterwards, it is important to prove that this defined score does really reproduce the intended outcome of the class selection criterion of min-max bleaching n-tuple classifier, as introduced in Section 2.2.2.

The use of this score measure leads to a system of linear equations, similar to the one of Proposition 1. *By a similar procedure to that of Proposition 2, one deduces that the system of equations has no solution if it has more equations than the intended upper bound of the VC dimension. Therefore, this upper bound is attained.* Again the lower and upper bounds are the same, and as so the VC dimension of the min-max bleaching n-tuple classifier has an exact value.

As previously mentioned in Section 2.2.3, the notation used for the proof of the VC dimension of the min-max bleaching n-tuple classifier is almost the same as that for the traditional model. Every notation needed is already introduced in Sections 2.2.3 and 4.2.1. This way, the notation included in Section 2.2.3 is revisited herein, in order to supersede some definitions set in Section 4.2.1. This notation is condensed in Table 4.3 for future reference.

Table 4.3: Notation for min-max bleaching n-tuple classifier.

$\mathcal{B}_{N,n}$	Bleaching N-node n-tuple classifying model
$B_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$	Bleaching N-node n-tuple classifier with memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1$
β	Bleaching threshold
$s_{-1}^{(\beta)}(\mathbf{Z}_k)$	Score of $\mathcal{D}_{-1}$ for $\mathbf{Z}_k$ and threshold β
$s_1^{(\beta)}(\mathbf{Z}_k)$	Score of $\mathcal{D}_1$ for $\mathbf{Z}_k$ and threshold β
$\mathbf{s}^{(\beta)}(\mathbf{Z}_k)$	Score vector $\mathbf{s}^{(\beta)}(\mathbf{Z}_k) = \left[s_{-1}^{(\beta)}(\mathbf{Z}_k)\,, s_1(\mathbf{Z}_k)\right]^{\mathrm{T}}$
$s_{-1}(\mathbf{Z}_k)$	β-independent score of $\mathcal{D}_{-1}$ for $\mathbf{Z}_k$
$s_1(\mathbf{Z}_k)$	β-independent score of $\mathcal{D}_1$ for $\mathbf{Z}_k$
$\mathbf{s}(\mathbf{Z}_k)$	Score vector $\mathbf{s}(\mathbf{Z}_k) = [s_{-1}(\mathbf{Z}_k)\,, s_1(\mathbf{Z}_k)]^{\mathrm{T}}$

Let $\mathbf{M}_{-1}$ and $\mathbf{M}_1 \in \mathbb{N}^{N\times 2^n}$ be memory matrices, and let $m_{-1,i,j}$ and $m_{1,i,j}$ be respectively the elements of their i-th row and j-th column. Let $\mathcal{B}_{N,n}$ be the family of functions that represents the biclass min-max bleaching N-node n-tuple classifying model. Let $B_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1) \in \mathcal{B}_{N,n}$ be a function $B_{N,n} : \{0,1\}^{N\times 2^n} \to \{-1, 1\}$ that given a feature matrix $\mathbf{Z}_k$ returns a class $c \in \{-1, 1\}$. $B_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ represents an n-tuple classifier whose learned knowledge is stored in the memory matrices $\mathbf{M}_{-1}$ and $\mathbf{M}_1$.

Let there also be $\mathbf{s}^{(\beta)}(\mathbf{Z}_k) = \left[s_{-1}^{(\beta)}(\mathbf{Z}_k)\,, s_1^{(\beta)}(\mathbf{Z}_k)\right]^{\mathrm{T}}$ a β-dependent score vector composed of $s_{-1}^{(\beta)}(\mathbf{Z}_k)$ and $s_1^{(\beta)}(\mathbf{Z}_k)$, the scores yielded by $\mathcal{D}_{-1}$ and $\mathcal{D}_1$ over $\mathbf{Z}_k$, respectively, given a bleaching threshold $\beta \in \mathbb{N}$. Those scores are defined according to Equation 2.8. Finally, $\mathbf{s}(\mathbf{Z}_k)$, $s_{-1}(\mathbf{Z}_k)$ and $s_1(\mathbf{Z}_k)$ are defined in Definition 6 and they respectively express the β-independent score vector, and the β-independent scores yielded by $\mathcal{D}_{-1}$ and $\mathcal{D}_1$.

4.3.2 The Proof Itself

The proof of the VC dimension of the min-max bleaching n-tuple classifying model starts by proving its lower bound.

Corollary 1. *The VC dimension, d_{VC}, of a model $\mathcal{B}_{N,n}$, as defined in Section 4.3.1, is bounded below by $N\,(2^n - 1) + 1$.*

Proof. The bleaching n-tuple classifier is a generalization of traditional WiSARD, for the former works exactly like the latter if its memory matrices only store binary values. Consequently, d_{VC} is bounded below by $N\,(2^n - 1) + 1$, the VC dimension of the traditional n-tuple classifier as indicated by Theorem 1. □

The proof of the upper bound of the VC dimension of the bleaching learning model needs the introduction of a score measure that independs from the bleaching

threshold β for the reasons exposed in Section 4.3.1.

Definiton 5. Let $\mathbf{\Lambda}_{-1} = [\lambda_{-1,i,j}]_{N\times 2^n}$ and $\mathbf{\Lambda}_1 = [\lambda_{1,i,j}]_{N\times 2^n}$ be matrices with same dimensions of $\mathbf{M}_{-1}$ and $\mathbf{M}_1$, respectively. For a given class $c \in \{-1, 1\}$, the elements of $\mathbf{\Lambda}_c$ are defined as $\lambda_{c,i,j} = -N^{-m_{c,i,j}}$.

Definiton 6. Let $\mathbf{s}(\mathbf{Z}_k) = [s_{-1}(\mathbf{Z}_k)\,, s_1(\mathbf{Z}_k)]^{\mathrm{T}}$ be a β-independent score vector. For a given class $c \in \{-1, 1\}$,

$$s_c(\mathbf{Z}_k) = \sum_{i=1}^{N}\sum_{j=1}^{2^n} \lambda_{c,i,j} z_{k,i,j} = -\sum_{i=1}^{N}\sum_{j=1}^{2^n} N^{-m_{c,i,j}} z_{k,i,j}. \tag{4.18}$$

Corollary 2.

$$B_{N,n}(\mathbf{Z}_k; \mathbf{M}_{-1}, \mathbf{M}_1) = \underset{c\in\{-1,1\}}{\operatorname{argmax}}\, s_c(\mathbf{Z}_k)\,. \tag{4.19}$$

Proof. Let $v_{-1,i}$ and $v_{1,i} \in \mathbb{N}$ respectively be the values of the accessed position of i-th node of $\mathcal{D}_{-1}$ and $\mathcal{D}_1$, hereinafter denoted as node scores. Suppose, without loss of generality, that $v_{-1,i} \geq v_{-1,j}$ and $v_{1,i} \geq v_{1,j}$ if $i < j$, i.e. nodes of both discriminators are sorted, so as the highest accessed positions belong to the ones with lowest indices.

Suppose also, without loss of generality, that $B_{N,n}(\mathbf{Z}_k; \mathbf{M}_{-1}, \mathbf{M}_1) = 1$, i.e. 1 was the class chosen by the classifier. So, according to the mathematical formulation given in Section 2.2.3, there is a bleaching threshold $\beta = \underset{\beta'}{\operatorname{argmin}}\, s_1^{(\beta')} > s_{-1}^{(\beta')}$.

For a bleaching n-tuple classifier to produce such scores, there must exist an $N' \leq N$, such that the learning machine has the following node scores: (i) $v_{1,i} \geq \beta + 1$ and $v_{-1,i} \geq \beta$, for every $i < N'$; (ii) $v_{1,N'} \geq \beta + 1$; (iii) $v_{-1,N'} = \beta$; and (iv) $v_{1,i} = v_{-1,i} \leq \beta$, for every $i > N'$. A classifier with these node scores opts for class 1 (as intended), because β is the smallest threshold for which there is a tie-break, with $s_1^{(\beta)} = N'$ and $s_{-1}^{(\beta)} \leq N' - 1$.

Because it was supposed, without loss of generality, that the classifier would select class 1, then β-independent score s_1 must be greater than s_{-1} for the corollary statement to be proven true. As

$$s_1 \geq -\sum_{i=N'+1}^{N} N^{-v_{1,i}} - N'N^{-\beta-1}, \tag{4.20}$$

$$s_{-1} < -\sum_{i=N'+1}^{N} N^{-v_{-1,i}} - N^{-\beta} \tag{4.21}$$

and

$$v_{1,i} = v_{-1,i} \leq \beta, \text{ for every } i > N', \tag{4.22}$$

then

$$s_1 - s_{-1} > -N'N^{-\beta-1} + N^{-\beta} \geq -N^{-\beta} + N^{-\beta} = 0. \tag{4.23}$$

That is, $s_1 > s_{-1}$.

Supposing the selected class to be -1 would lead to an analogous proof, that would result in $s_{-1} < s_1$, as expected. Finally, for a draw scenario, every node score $s_{1,i}$ should be equal to $s_{-1,i}$, for every i. This way, β-independent scores s_1 and s_{-1} would also be equal.

Summarizing, when $s_1(\mathbf{Z}_k) > s_{-1}(\mathbf{Z}_k)$, the network opts for class 1; if $s_{-1}(\mathbf{Z}_k) > s_1(\mathbf{Z}_k)$, it chooses -1; and if both β-independent scores are identical, the classifier outputs that there is a draw. Thus, it confirms the identity proposed by this corollary. □

The use of β-independent scores makes the function $B_{N,n}(\cdot; \mathbf{M}_{-1}, \mathbf{M}_1)$ equivalent to $W_{N,n}(\cdot; \mathbf{\Lambda}_{-1}, \mathbf{\Lambda}_1)$ (compare Definition 6 and Corollary 2 to Equations 2.4 and 2.5). Consequently, a system of linear equations with similar characteristics to Equation 4.12 can be derived from the relation between the β-independent scores, as mentioned in Section 4.3.1.

Lemma 3. *The VC dimension, d_{VC}, of a model $\mathcal{B}_{N,n}$, as defined in Section 4.3.1, is bounded above by $N(2^n - 1) + 1$.*

Proof. Analogously to Proposition 1, the formulas of β-independent score measures $s_{-1}(\cdot)$ and $s_1(\cdot)$ lead to a system of linear equations, whose coefficient matrix is the same. The column that represents the right hand side of the equations is also linearly independent from those of the coefficient matrix. Therefore, there lacks a solution for such system if it has at least $N(2^n - 1) + 2$ equations. Then, $\mathcal{B}_{N,n}$ does not shatter any set of feature matrices $Z, \text{card}(Z) = N(2^n - 1) + 2$. □

Theorem 2. *The VC dimension, d_{VC}, of a model $\mathcal{B}_{N,n}$, as defined in Section 4.3.1, is given by $N(2^n - 1) + 1$.*

Proof. The VC dimension of $\mathcal{B}_{N,n}$ is bounded above and below by $N(2^n - 1) + 1$. Then, it is exactly $N(2^n - 1) + 1$. □

4.4 Discussion

Calculations on the VC dimensions of WiSARD and bleaching n-tuple classifiers show that their values are the same, indicating that there is at least one bleaching technique that mitigates the saturation and does little to no harm to the network generalization capacity. Their values, however, are higher than what would be expected from previous experimental works. For instance, the WiSARD network

employed in the universal configuration of Table 3.7 has discriminators with more than 95 nodes, each one addressed by n-tuples of length 88. The VC dimension of this learning machine is approximately $3 \cdot 10^{28}$, which is far greater than the size of the largest dataset it was subjected to (less than $1.3 \cdot 10^6$ elements). The determination of the VC dimension of traditional and bleaching n-tuple classifiers was performed for dichotomous classifications. Despite the polychotomous nature of part-of-speech tagging, the analysis here given provides an insight on the learning capacity of those learning machines in a practical application. Yet, classically in the literature multiclass classification is often executed through algorithms that use dichotomous classifications, e.g., multiclass support vector machines (SVMs) [118].

The VC dimensions of both models suggest a similarity between them and other learning machines whose input is transformed into a feature vector, which is affected by the model effective parameters to produce the system output. For instance, extreme learning machines (ELMs) [119, 120] have pseudorandomly generated weights that connect its input layer to its hidden one. These weights are analogous to the addressing function introduced in Section 2.1.4. Besides, the content of the hidden layer neurons are affected by an weighted sum, which produces the ELM output. A similar process happens in WiSARD and bleaching n-tuple classifiers, where the feature vector generated by the addressing function is subjected to the system memory matrices in order to yield the network response. A comparison between the VC dimensions of those models also indicates a similarity between them. The VC dimension of ELMs is determined by the number of weights connecting its hidden to its output layer [121]. In a similar fashion, the number of addressable positions of an n-tuple classifier is tightly related to its VC dimension.

Despite their similarities, the weightless recognition methods differ in some points from their weighted counterparts. The latter ones rely on optimization methods and update all their weights at a training procedure, while the former ones only change the memory matrix elements that are addressed when the network is being trained. This way, WiSARD and bleaching n-tuple classifiers can generalize pretty well even if the amount of training data is far smaller than their VC dimension. The same does not happen in ELMs [121]. Actually, both weightless learning systems seem to have fewer effective parameters than their VC dimension imply. They have a pay-only-for-what-you-use policy, where the only positions to be updated are the ones that reveal to be relevant during the training procedure.

One might get a glimpse of the impact of this policy by calculating the VC dimension of a particular subset of WiSARD and bleaching N-node n-tuple classifying models. If a WiSARD network is exposed to a training dataset D composed of ω_1 observations associated to class 1 and ω_{-1} to class -1, then every node $d_{1,i}$ of discriminator $\mathcal{D}_1$ should have at most ω_1 positions storing 1. Analogously, every node

$d_{-1,i}$ of $\mathcal{D}_{-1}$ should have at most ω_{-1} positions storing 1. This way, by appending inequalities

$$\sum_{j=1}^{2^n} m_{c,i,j} \leq \omega_c,\ \forall i \in \{1,\ldots,N\} \text{ and } c \in \{-1,1\} \tag{4.24}$$

to the system introduced in Proposition 1, one could obtain a capacity measure lower than the calculated VC dimension, which would better depict how that policy affects WiSARD generalization capacity. The bleaching n-tuple classifier has a very characteristic property. If a learning machine of this sort is exposed to that same training dataset, then the sum of the contents of every node $d_{c,i}$ of a discriminator $\mathcal{D}_c$ is always ω_c. So, to attain the desired measure, it would be necessary to append equations

$$\sum_{j=1}^{2^n} m_{c,i,j} = \omega_c,\ \forall i \in \{1,\ldots,N\} \text{ and } c \in \{-1,1\} \tag{4.25}$$

to its equivalent linear system. But, because the system relies on the modified memory matrices $\mathbf{\Lambda}_{-1}$ and $\mathbf{\Lambda}_1$, and not on $\mathbf{M}_{-1}$ and $\mathbf{M}_1$, appending those equations would introduce a nonlinearity to the system that should toughen its resolution. That task becomes easier if those equations are replaced by

$$-2^n N^{-2^{-n}\omega_c} \leq \sum_{j=1}^{2^n} \lambda_{c,i,j} \leq -2^n + 1 - N^{-\omega_c}, \forall i \in \{1,\ldots,N\} \text{ and } c \in \{-1,1\}\,. \tag{4.26}$$

However, this might pose another problem, as it is quite improbable to find an exact solution. So, this measure would have to be expressed by lower and upper bounds.

The bounds presented in Equation 4.26 represent the most extreme forms in which the content of a memory matrix may be distributed. Its left hand side corresponds to a memory matrix where every position store the same amount, $2^{-n}\omega_c$. Conversely, its right hand side corresponds to one where a single memory position store ω_c and the $2^n - 1$ others store 0.

Finally, Equation 4.25 leads to the conclusion that the VC dimension of the bleaching n-tuple classifier can be lower than that of the traditional model. Some memory contents do not correspond to a bleaching n-tuple classifier that is trained like it was presented in Section 2.2.1. If one restricts the family of functions a bleaching n-tuple classifier can characterize to only those that a training procedure could provide, then Equation 4.25 holds true, and so does

$$\sum_{j=1}^{2^n} m_{c,i,j} - m_{c,1,j} = 0,\ \forall i \in \{2,\ldots,N\} \text{ and } c \in \{-1,1\}\,. \tag{4.27}$$

This way, further constraints that do not depend on the training data can be

added to the linear inequality system derived during the proof of the upper bound for the VC dimension of the bleaching n-tuple classifier, leading to a VC dimension lower than that of WiSARD. However, this restriction might only be of aid in the calculation of the VC dimension of the bleaching model if a new tie-breaking policy is designed, such that discriminator scores can be produced through linear combinations of the contents of its memory matrices.

Although there is room for improvement in the determination of the VC dimension of bleaching n-tuple classifier, this was not done in this work. This thesis intended to show the potential of the bleaching n-tuple classifier and introduce a theoretical background for this technique concerning its generalization capacity. Comparisons with weighted learning machines and studies on tie-breaking policies are immediate directions to be taken to advance the theoretical research on the bleaching n-tuple classifier.

Chapter 5

Conclusion

This thesis delivered an exact value for the VC dimension of the traditional WiSARD n-tuple classifier and also supplied a theoretical background for the bleaching n-tuple classifier, by introducing a mathematical study on its generalization capacity and leaving some insights on how these weightless models can be compared to weighted learning machines. This work also explored the capabilities of the bleaching n-tuple classifier by revisiting an application that required the handle of a large amount of data, mWANN-Tagger, a multilingual part-of-speech tagger initially proposed in [10] and improved in [11] and [12]. The thesis also provided a universal parameter configuration for mWANN-Tagger as a means to guarantee its applicability to multilingual part-of-speech tagging, given that the tagger could then be promptly employed in a new corpus with no need of fine tuning of parameters.

5.1 Summary of the Thesis

Here is offered a brief recap of what was presented and what was achieved in each chapter of this thesis.

5.1.1 Chapter 1

Chapter 1 introduced the aims and contents of this thesis. It served as a motivation for the reader, showing the reasons that led to this research and how it was related to other published works.

5.1.2 Chapter 2

In Chapter 2, the traditional WiSARD n-tuple classifier was described. Its architecture was detailed, as well as its training and recognition procedures. A mathematical formulation for this learning machine similar to the one provided by STECK [8] was

also settled. The saturation problem was mentioned, along with the technique proposed by GRIECO *et al.* [9] to mitigate it.

The description of the bleaching n-tuple classifier was also divided in three parts. Its architecture and training procedures were explained almost straightforwardly, because it was only a matter of explaining that the memory positions could store any positive number, and that they needed to be incremented by 1 every time they were accessed during the training phase, instead of only set to 1 as soon as they were accessed for the first time. The bleaching threshold was introduced, as well as its function in the recognition procedure of the bleaching n-tuple classifier. A discussion on tie-breaking policies was made and the min-max bleaching n-tuple classifier was defined, since it was necessary for the settling of a mathematical formulation for that model.

5.1.3 Chapter 3

mWANN-Tagger had the advantage of using a weightless architecture that works well with a big amount of data, so it could train several large corpora in a quite agile fashion. However, it had some parameters that needed to be finely tuned and so, it could represent a step back in the search for a multilingual part-of-speech tagger. This chapter aimed to show a relation between the lexical diversity of a corpus and the most suited parameter configuration for this textual base.

For a better comprehension of the content of this chapter, a brief introduction to quantitative linguistics was given. A discussion on the diverse forms a same piece of information can be encoded into words was provided and metrics to quantify it were proposed. It was then chosen to employ the parameters of the Zipf-Mandelbrot distribution, as a means to store in two small numbers, α and β, the knowledge about how a language encodes information in sentence.

In this chapter, mWANN-Tagger [10–12] was revisited. Its architecture and how it encodes words in a sentence into bit arrays were explained with the aid of an example. Its parameters were reintroduced and tests were made to verify if there was any relation between the best suited parameter configuration for a given corpus and parameters α and β of the Zipf-Mandelbrot distribution that describes its lexical diversity. This test was performed for eight corpora in languages that encode information into sentences in quite different ways, but no clear relation was found whatsoever.

It was then noted that despite the differences between the languages, their parameter configurations looked very similar, except for a few cases. A cross-linguistic analysis was conducted on those corpora in order to attest if there was actually a configuration best suited for a given corpus, or if all of them were interchangeable

and mWANN-Tagger would perform well with any one of them. The sensitivity analysis showed that the configurations were actually interchangeable and that a single parameter configuration could be employed in all those corpora. In light of this, a universal parameter configuration was then devised.

That single common configuration was subjected to several tests to check its potential for universality. A sensitivity analysis was conducted on that configuration, where each one of its parameters was slightly changed and mWANN-Tagger accuracy was collected. The best accuracies achieved in this analysis were those produced by the common configuration with no change in its parameters, indicating its potential for universality. That configuration was then put to test on six other corpora in languages that ranged from very isolating to polysynthetic ones. Another sensitivity analysis was conducted and again the best accuracies achieved were those of the common configuration for almost every corpus. That was only not the case for some parameters when employing the Pirahã dataset. However, that corpus was extremely small when compared to the others and that discrepancy could be caused by it. Finally, the performance of mWANN-Tagger with the universal configuration is compared to those of state-of-the-art multilingual part-of-speech taggers in the treebanks of Universal Dependencies project (v. 1.4). This comparison showed that mWANN-Tagger can outperform some state-of-the-art part-of-speech taggers if a better word representation is provided to the tagger.

The chapter ended with a discussion, where it was mentioned that a single universal parameter configuration was found for mWANN-Tagger and that it was subjected to several tests that indicate its actual potential for universality. It was also commented that mWANN-Tagger still has room for improvement and that better word representations, like character-level ones, might enhance the tagger performance.

5.1.4 Chapter 4

The chapter started by reminding the reader of the problem posed by memory saturation and how BRADSHAW [28] used learning theory to try to find a way to lessen its effects. As a counterpoint, the bleaching technique proved to be capable of mitigating the saturation of WiSARD through a series of works where it was exposed to large loads of data, but no theoretical studies were carried out on this architecture of the n-tuple classifier. The aim of Chapter 4 was to provide a theoretical background for the bleaching n-tuple classifier linked to its generalization capacity.

Like in Chapter 3, some basic concepts were given in Section 4.1 for a better comprehension of the study there described. A brief introduction on the ERM principle and some key definitions were offered. The idea of consistency is presented and it is explained that for the ERM principle to be consistent in a learning ma-

chine, its VC dimension should be finite. BRADSHAW [28] argued that the ERM principle is always consistent with finite sets of functions, such as those produced by traditional n-tuple classifiers. Section 4.1 ended by informing that the bleaching n-tuple classifier yields an infinite set of functions, so it was pertinent to assess the VC dimension of that learning machine.

The following sections presented the calculations of the VC dimensions of the traditional and the min-max bleaching n-tuple classifiers (defined in Section 2.2.2). For each variant, the lower and upper bounds of their VC dimensions were determined. In both cases the bounds proved to be equal and thus exact values were achieved for the VC dimensions of both systems. Furthermore, their values were also identical, indicating that the benefits brought by the bleaching technique did not imply in some harm on the generalization capacity of the WiSARD network.

Exact VC dimensions were obtained for both n-tuple classifiers, as it was intended. Their values were notably big nevertheless. A discussion followed the calculations, where a parallel was drawn with weighted learning machines, particularly ELMs. A comparison taking into account their architecture and VC dimension revealed similarities between those models and n-tuple classifiers. Next, it was reminded that the way the n-tuple method trains quite differently from its weighted counterparts, and therefrom proposals to achieve a lower (and closer to reality) VC dimension for the bleaching n-tuple classifier were offered.

5.1.5 Chapter 5

This chapter serves as a summary of the work described in this thesis, with a special focus on its contributions. It also indicates some directions that can be taken to further advance the research here presented.

5.2 Final Remarks

The intention to build an efficient POS-tagger that were versatile enough to work with several different languages led to the choice for a weightless architecture [10, 11]. However, this brought the need for the fine tuning of some parameters, which could impair its intended efficiency. This necessity drove the research towards the quantitative linguistics field, where it was expected that measures concerning a given corpus lexical diversity could help in the choice of the most suitable parameter configuration for that textual base. No direct relation was found between those measures and the parameters of mWANN-Tagger, but a universal parameter configuration was found instead [12]. Although it seemed counterintuitive, a possible explanation can be that the information given by the word itself could complement that given by its

vicinity and vice versa. In other words, even if an isolating language could rely more on the context of a word to disambiguate its tag, the word itself might have enough information to complement that knowledge in order to aid the tagger in its task. Analogously, the vicinity of a word may help mWANN-Tagger accurately tag them, even if the content of the corpus employed is written in a very synthetic language.

On the theoretical basis, this thesis provided the exact values of the VC dimension of WiSARD and of min-max bleaching n-tuple classifiers. Because they are identical, it was concluded that the bleaching technique enhances the WiSARD model doing little to no harm to its generalization capacity. This work also drew a parallel with other pattern recognition machines, showing similarities and differences between these and the weightless systems. This paper argued as well on how the pay-only-for-what-you-use policy of the n-tuple classifiers could be interpreted as an advantage, for it would imply that the model generalization capacity should grow according to the dataset employed for training.

5.3 Future Works

This thesis leaves some directions for enhancements in mWANN-Tagger architecture. The tagger may benefit from a better word representation. A character-level representation offers more information about the word being tagged, which could lead to higher accuracies. It would also be desirable that the acquired word representation were more compact than the current one. This would imply in a lighter architecture for mWANN-Tagger and, consequently, more efficient training and tagging procedures – one of the main reasons for opting for weightless learning systems. The universal configuration could also be improved if it were subjected to larger corpora on polysynthetic languages. The work on MIT Pirahã Corpus showed that mWANN-Tagger actually produced a relatively high accuracy when using that configuration. However, that corpus was quite small and experiments on larger corpora are important to get more conclusive results. Unfortunately, by the time this thesis was written no other annotated corpus in a polysynthetic language was found.

The study of the generalization capacity of other bleaching tie-breaking policies and the investigation on which one is best suited for a given task is an almost direct consequence to the theoretical research offered in this thesis. A deeper analysis on the impact of the training dataset size on the generalization capacity of WiSARD and bleaching n-tuple classifiers is also suggested as further improvement to this work, as well as how these weightless models relate to their weighted counterparts. Because of their multidiscriminator natures, the traditional and bleaching recognition machines could also benefit from studies on their generalization capacities, by employing multiclass extensions to the VC dimension.

Bibliography

[1] MÜLLER, T., SCHMID, H., SCHÜTZE, H. "Efficient Higher-Order CRFs for Morphological Tagging". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013.

[2] NGUYEN, D. Q., NGUYEN, D. Q., PHAM, D. D., et al. "A robust transformation-based learning approach using ripple down rules for part-of-speech tagging", *AI Communications*, v. 29, n. 3, pp. 409–422, April 2016. ISSN: 1875-8452. doi: 10.3233/AIC-150698.

[3] MORE, A., TSARFATY, R. "Data-Driven Morphological Analysis and Disambiguation for Morphologically Rich Languages and Universal Dependencies". In: *Proceedings of COLING 2016*, December 2016.

[4] PLANK, B., SØGAARD, A., GOLDBERG, Y. "Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics (ACL), 2016. doi: 10.18653/v1/p16-2067.

[5] ALEKSANDER, I., STONHAM, T. J. "Guide to pattern recognition using random-access memories", *IEE Journal on Computers and Digital Techniques*, v. 2, n. 1, pp. 29–40, 1979. doi: 10.1049/el:19770110.

[6] ALEKSANDER, I., THOMAS, W. V., BOWDEN, P. A. "WISARD: A Radical Step Foward in Image Recognition", *Sensor Review*, v. 4, pp. 120–124, July 1984.

[7] BLEDSOE, W. W., BROWNING, I. "Pattern Recognition and Reading by Machine". In: *Papers Presented at the December 1-3, 1959, Eastern Joint IRE-AIEE-ACM Computer Conference*, IRE-AIEE-ACM '59 (Eastern), p. 225–232, New York, NY, USA, 1959. ACM.

[8] STECK, G. P. "Stochastic Model for the Browning-Bledsoe Pattern Recognition Scheme", *IRE Transactions on Electronic Computers*, v. EC-11,

n. 2, pp. 274–282, April 1962. ISSN: 0367-9950. doi: 10.1109/TEC.1962.5219360.

[9] GRIECO, B. P. A., LIMA, P. M. V., GREGORIO, M. D., et al. "Producing pattern examples from "mental" images", *Neurocomputing*, v. 73, n. 7–9, pp. 1057–1064, March 2010.

[10] CARNEIRO, H. C. C. *A Função do Índice de Síntese das Linguagens na Classificação Gramatical com Redes Neurais Sem Peso*. M.Sc. Thesis, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brazil, 2012.

[11] CARNEIRO, H. C. C., FRANÇA, F. M. G., LIMA, P. M. V. "Multilingual Part-of-speech Tagging with Weightless Neural Networks", *Neural Networks*, v. 66, n. C, pp. 11–21, June 2015. ISSN: 0893-6080. doi: 10.1016/j.neunet.2015.02.012.

[12] CARNEIRO, H. C. C., PEDREIRA, C. E., FRANÇA, F. M. G., et al. "A universal multilingual weightless neural network tagger via quantitative linguistics", *Neural Networks*, v. 91, pp. 85–101, July 2017. ISSN: 0893-6080. doi: 10.1016/j.neunet.2017.04.011.

[13] CARDOSO, D. O., CARVALHO, D. S., ALVES, D. S. F., et al. "Financial credit analysis via a clustering weightless neural classifier", *Neurocomputing*, v. 183, pp. 70–78, March 2016. doi: 10.1016/j.neucom.2015.06.105.

[14] DE SOUZA, D. F. P., FRANÇA, F. M. G., LIMA, P. M. V. "Spatio-temporal Pattern Classification with KernelCanvas and WiSARD". In: *2014 Brazilian Conference on Intelligent Systems*. Institute of Electrical and Electronics Engineers (IEEE), October 2014. doi: 10.1109/bracis.2014.49.

[15] NASCIMENTO, D. N., DE CARVALHO, R. L., MORA-CAMINO, F., et al. "A WiSARD-based multi-term memory framework for online tracking of objects". In: *Proceedings of the 23st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning ESANN-2015*, p. 19–24, 2015.

[16] DE SOUZA, D. F. P., FRANÇA, F. M. G., LIMA, P. M. V. "Real-Time Music Tracking Based on a Weightless Neural Network". In: *2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*. Institute of Electrical and Electronics Engineers (IEEE), July 2015. doi: 10.1109/cisis.2015.84.

[17] SNYDER, B., NASEEM, T., EISENSTEIN, J., et al. "Unsupervised multilingual learning for POS tagging". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, p. 1041–1050, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[18] NASEEM, T., SNYDER, B., EISENSTEIN, J., et al. "Multilingual Part-Of-Speech Tagging: Two Unsupervised Approaches", *Journal of Artificial Intelligence Research*, v. 36, pp. 341–385, 2009.

[19] DAS, D., PETROV, S. "Unsupervised Part-of-speech Tagging with Bilingual Graph-based Projections". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, p. 600–609, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN: 978-1-932432-87-9.

[20] PETROV, S., DAS, D., MCDONALD, R. "A Universal Part-of-Speech Tagset". In: *LREC–Language Resources Evaluation Conference*, May 2012.

[21] TÄCKSTRÖM, O., DAS, D., PETROV, S., et al. "Token and Type Constraints for Cross-Lingual Part-of-Speech Tagging", *Transactions of the Association for Computational Linguistics*, v. 1, n. March, pp. 1–12, 2013.

[22] NIVRE, J., AGIĆ, Ž., AHRENBERG, L., et al. "Universal Dependencies 1.4". 2016. Available in: <`http://hdl.handle.net/11234/1-1827`>. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University in Prague.

[23] NIVRE, J., DE MARNEFFE, M.-C., GINTER, F., et al. "Universal Dependencies v1: A Multilingual Treebank Collection". In: Chair), N. C. C., Choukri, K., Declerck, T., et al. (Eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, May 2016. European Language Resources Association (ELRA). ISBN: 978-2-9517408-9-1.

[24] ZHANG, Y., GADDY, D., BARZILAY, R., et al. "Ten Pairs to Tag - Multilingual POS Tagging via Coarse Mapping between Embeddings". In: *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, p. 1307–1317, 2016.

[25] SAPIR, E. *Language: An Introduction to the Study of Speech*. New York, Harcourt, Brace and Company, 1921.

[26] VAPNIK, V. N., CHERVONENKIS, A. Y. "On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities", *Theory of Probability and its Applications*, v. 16, n. 2, pp. 264–280, 1971.

[27] VAPNIK, V. N. *Statistical Learning Theory*. Wiley, 1998.

[28] BRADSHAW, N. P. *An Analysis of Learning in Weightless Neural Systems*. Ph.D. Thesis, Imperial College, September 1996.

[29] BRADSHAW, N. P., ALEKSANDER, I. "Improving the Generalization of the N-Tuple Classifier using the Effective VC Dimension", *Electronic Letters*, v. 32, n. 20, pp. 1904–1905, 1996.

[30] RATNAPARKHI, A. "A Maximum Entropy Model for Part-Of-Speech Tagging". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, April 1996.

[31] TOUTANOVA, K., KLEIN, D., MANNING, C. D., et al. "Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network". In: *Proceedings of HLT-NAACL 2003*, pp. 252–259, 2003.

[32] DOS SANTOS, C. N., ZADROZNY, B. "Learning Character-level Representations for Part-of-Speech Tagging". In: *Proceedings of ICML*, 2014.

[33] SCHNABEL, T., SCHÜTZE, H. "FLORS: Fast and Simple Domain Adaptation for Part-of-Speech Tagging", *Transactions of the Association for Computational Linguistics*, v. 2, pp. 15–26, 2014.

[34] LABEAU, M., LÖSER, K., ALLAUZEN, A. "Non-lexical neural architecture for fine-grained POS Tagging". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (ACL), 2015. doi: 10.18653/v1/d15-1025.

[35] YIN, W., SCHNABEL, T., SCHÜTZE, H. "Online Updating of Word Representations for Part-of-Speech Taggging". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[36] YANG, Y., EISENSTEIN, J. "Unsupervised Multi-Domain Adaptation with Feature Embeddings". In: *Proceedings of the 2015 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 672–682, Denver, Colorado, May–June 2015. Association for Computational Linguistics.

[37] ZIPF, G. K. *Selected Studies of the Principle of Relative Frequency in Language.* Harvard University Press, 1932.

[38] ZIPF, G. K. *Human Behavior and the Principle of Least Effort.* Addison-Wesley, Reading MA (USA), 1949.

[39] MANDELBROT, B. B. "An information theory of the statistical structure of language", *Communication Theory*, p. 503–512, 1953.

[40] BEST, K.-H. "Word class frequencies in contemporary German short prose texts", *Journal of Quantitative Linguistics*, v. 1, n. 2, pp. 144–147, January 1994. doi: 10.1080/09296179408590008.

[41] ZIEGLER, A. "Word class frequencies in Brazilian-Portuguese press texts", *Journal of Quantitative Linguistics*, v. 5, n. 3, pp. 269–280, December 1998. doi: 10.1080/09296179808590136.

[42] UHLÍŘOVÁ, L. "On Language Modelling in Automatic Speech Recognition", *Journal of Quantitative Linguistics*, v. 7, n. 3, pp. 209–216, 2000. doi: 10.1076/jqul.7.3.209.4113.

[43] BEST, K.-H. "Zur Gesetzmäßigkeit der Wortverteilung in deutschen Texten", *Glottometrics*, v. 1, pp. 1–26, 2001.

[44] WIMMER, G., ALTMANN, G. "Some statistical investigations concerning word classes", *Glottometrics*, v. 1, pp. 109–123, 2001.

[45] FAN, F., ALTMANN, G. "On meaning diversification in English", *Glottometrics*, v. 17, pp. 66–78, 2008.

[46] LIU, H. "Probability Distribution of Dependencies Based on a Chinese Dependency Treebank", *Journal of Quantitative Linguistics*, v. 16, n. 3, pp. 256–273, August 2009. doi: 10.1080/09296170902975742.

[47] TUZZI, A., POPESCU, I.-I., ALTMANN, G. "Parts-of-speech diversification in Italian texts", *Glottometrics*, v. 19, pp. 42–48, 2009.

[48] VINCZE, V. "The Relationship of Dependency Relations and Parts of Speech in Hungarian", *Journal of Quantitative Linguistics*, v. 22, n. 1, pp. 44–54, 2015. doi: 10.1080/09296174.2014.974458.

[49] WANG, L. "Part-of-speech studies in Chinese", *Journal of Quantitative Linguistics*, v. 23, n. 3, pp. 235–255, June 2016. doi: 10.1080/09296174.2016.1169851.

[50] JAYAWEERA, A. J. P. M. P., DIAS, N. G. J. "Unknown words analysis in POS tagging of Sinhala language". In: *2014 14th International Conference on Advances in ICT for Emerging Regions (ICTer)*. Institute of Electrical and Electronics Engineers (IEEE), December 2014. doi: 10.1109/icter.2014.7083928.

[51] LO, R. T.-W., HE, B., OUNIS, I. "Automatically Building a Stopword List for an Information Retrieval System." *Journal on Digital Information Management*, v. 3, n. 1, pp. 3–8, 2005.

[52] HOFFMANN, B., LIFSHITS, M., LIFSHITS, Y., et al. "Maximal Intersection Queries in Randomized Input Models", *Theory of Computing Systems*, v. 46, n. 1, pp. 104–119, oct 2008. doi: 10.1007/s00224-008-9154-6.

[53] BLEDSOE, W. W., BISSON, C. L. "Improved Memory Matrices for the n-Tuple Pattern Recognition Method", *IRE Transactions on Electronic Computers*, v. EC-11, n. 3, pp. 414–415, June 1962.

[54] ROY, R. J., SHERMAN, J. "Two Viewpoints of k-Tuple Pattern Recognition", *IEEE Transactions on Systems Science and Cybernetics*, v. 3, n. 2, pp. 117–120, 1967. doi: 10.1109/tssc.1967.300092.

[55] ULLMANN, J. R. "Experiments with the n-Tuple Method of Pattern Recognition", *IEEE Trans. Comput.*, v. 18, n. 12, pp. 1135–1137, December 1969.

[56] ULLMANN, J. R. "Reduction of the storage requirements of Bledsoe and Browning's n-tuple method of pattern recognition", *Pattern Recognition*, v. 3, n. 3, pp. 297–306, 1971.

[57] STONHAM, T. J. "Improved Hamming-distance analysis for digital learning networks", *Electronics Letters*, v. 13, n. 6, pp. 155, 1977. doi: 10.1049/el:19770110.

[58] TARLING, R., ROHWER, R. "Efficient use of training data in the n-tuple recognition method", *Electronics Letters*, v. 29, n. 24, pp. 2093, 1993. doi: 10.1049/el:19931398.

[59] MITCHELL, R. J., BISHOP, J. M., MINCHINTON, P. R. "Optimising memory usage in n-tuple neural networks", *Mathematics and Computers in Simulation*, v. 40, n. 5-6, pp. 549–563, may 1996. doi: 10.1016/0378-4754(95)00006-2.

[60] ROHWER, R., MORCINIEC, M. "A Theoretical and Experimental Account of n-Tuple Classifier Performance", *Neural Computation*, v. 8, n. 3, pp. 629–642, apr 1996. doi: 10.1162/neco.1996.8.3.629.

[61] BRADSHAW, N. P. "The effective VC dimension of the n-tuple classifier". In: Gerstner, W., Germond, A., Hasler, M., et al. (Eds.), *Artificial Neural Networks — ICANN'97: 7th International Conference, Lausanne, Switzerland, October 8–10, 1997 Proceedings*, p. 511–516, Berlin, Heidelberg, Springer Berlin Heidelberg, 1997. doi: 10.1007/BFb0020206.

[62] GREGORIO, M. D. *On the reversibility of multi-discriminator systems*. Relatório Técnico 125/97, Istituto di Cibernetica–CNR, 1997.

[63] ROHWER, R., MORCINIEC, M. "The Generalisation Cost of RAMnets". In: Mozer, M., Jordan, M., Petsche, T. (Eds.), *Advances in Neural Information Processing Systems 9*, MIT Press, p. 253–259, 1997.

[64] ROHWER, R., MORCINIEC, M. "The Theoretical and Experimental Status of the n-Tuple Classifier", *Neural Networks*, v. 11, n. 1, pp. 1–14, 1998.

[65] JØRGENSEN, T. M., LINNEBERG, C. "Theoretical analysis and improved decision criteria for the n-tuple classifier", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 21, n. 4, pp. 336–347, April 1999. doi: 10.1109/34.761264.

[66] LINNEBERG, C., JØRGENSEN, T. M. "Cross-validation techniques for n-tuple-based neural networks". In: Lindblad, T., Padgett, M. L., Kinser, J. M. (Eds.), *Ninth Workshop on Virtual Intelligence/Dynamic Neural Networks*. SPIE-Intl Soc Optical Eng, mar 1999. doi: 10.1117/12.343045.

[67] WICKERT, I., FRANÇA, F. M. G. "AUTOWISARD: Unsupervised Modes for the WISARD". In: Mira, J., Prieto, A. (Eds.), *Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence*, v. 2084, *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, p. 435–441, 2001.

[68] AZHAR, H. B., DIMOND, K. "A Stochastic Search Algorithm to Optimize an N-tuple Classifier by Selecting Its Inputs". In: Campilho, A., Kamel, M.

(Eds.), *Image Analysis and Recognition: International Conference ICIAR 2004, Porto, Portugal, September 29- October 1, 2004, Proceedings, Part I*, p. 556–563, Berlin, Heidelberg, Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-540-30125-7_69.

[69] CARVALHO, D. S., CARNEIRO, H. C. C., FRANÇA, F. M. G., et al. "B-bleaching: Agile Overtraining Avoidance in the WISARD Weightless Neural Classifier". In: *Proceedings of the 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning ESANN-2013*, p. 515–520, 2013.

[70] CARNEIRO, H. C. C., FRANÇA, F. M. G., LIMA, P. M. V. "WANN-Tagger: A Weightless Artificial Neural Network Tagger for the Portuguese Language". In: Filipe, J., Kacprzyk, J. (Eds.), *ICFC-ICNC 2010 - Proceedings of the International Conference on Fuzzy Computation and International Conference on Neural Computation, [parts of the International Joint Conference on Computational Intelligence IJCCI 2010], Valencia, Spain, October 24-26, 2010*, p. 330–335. SciTePress, 2010.

[71] GREENBERG, J. H. "A quantitative approach to the morphological typology of language", *International Journal of American Linguistics*, v. 26, n. 3, pp. 178–194, 1960.

[72] DRYER, M. S. "Prefixing vs. Suffixing in Inflectional Morphology". In: Dryer, M. S., Haspelmath, M. (Eds.), *The World Atlas of Language Structures Online*, Leipzig, Max Planck Institute for Evolutionary Anthropology, 2013. Available in: <`http://wals.info/chapter/26`>.

[73] PIANTADOSI, S. T. "Zipf's word frequency law in natural language: A critical review and future directions", *Psychonomic Bulletin & Review*, v. 21, n. 5, pp. 1112–1130, 2014. ISSN: 1531-5320. doi: 10.3758/s13423-014-0585-6.

[74] POPESCU, I.-I., ALTMANN, G. "Hapax Legomena and Language Typology", *Journal of Quantitative Linguistics*, v. 15, n. 4, pp. 370–378, nov 2008. doi: 10.1080/09296170802326699.

[75] POPESCU, I.-I., ALTMANN, G. "Zipf's mean and language typology", *Glottometrics*, v. 16, pp. 31–37, 2008.

[76] BENTZ, C., KIELA, D., HILL, F., et al. "Zipf's law and the grammar of languages: A quantitative study of Old and Modern English parallel texts", *Corpus Linguistics and Linguistic Theory*, v. 10, n. 2, jan 2014. doi: 10.1515/cllt-2014-0009.

[77] BAAYEN, R. H. *Word frequency distributions*. Kluwer Academic, 2001. ISBN: 9780792370178.

[78] BENTZ, C., KIELA, D. "Zipf's law across languages of the world: Towards a quantitative measure of lexical diversity". In: *The Evolution of Language*. World Scientific Pub Co Pte Lt, apr 2014. doi: 10.1142/9789814603638_0056.

[79] IZSÁK, F. "Maximum likelihood estimation for constrained parameters of multinomial distributions—Application to Zipf–Mandelbrot models", *Computational Statistics & Data Analysis*, v. 51, n. 3, pp. 1575–1583, 2006. ISSN: 0167-9473. doi: 10.1016/j.csda.2006.05.008.

[80] FRANCIS, W. N., KUČERA, H. *A Standard Corpus of Present-Day Edited American English, for use with Digital Computers*. Relatório técnico, Department of Linguistics, Brown University, Providence, Rhode Island, US, 1964.

[81] SKUT, W., KRENN, B., BRANTS, T., et al. "An annotation scheme for free word order languages". In: *Proceedings of the 5th conference on Applied natural language processing*, ANLC '97, p. 88–95. Association for Computational Linguistics, 1997.

[82] BOSCO, C., LOMBARDO, V., VASSALLO, D., et al. "Building a Treebank for Italian: a Data-driven Annotation Schema". In: *Proceedings of the 2nd International Conference on Language Resources and Evaluation LREC-2000*, p. 99–105, 2000.

[83] HINRICHS, E. W., BARTELS, J., KAWATA, Y., et al. "The VERBMOBIL Treebanks". In: *KONVENS 2000 / Sprachkommunikation, Vorträge der gemeinsamen Veranstaltung 5. Konferenz zur Verarbeitung natürlicher Sprache (KONVENS), 6. ITG-Fachtagung "Sprachkommunikation"*, p. 107–112. VDE-Verlag GmbH, 2000.

[84] XIA, F., PALMER, M., XUE, N., et al. "Developing Guidelines and Ensuring Consistency for Chinese Text Annotation". In: *Proceedings of the 2nd International Conference on Language Resources and Evaluation LREC-2000*, 2000.

[85] AFONSO, S., BICK, E., HABER, R., et al. ""Floresta Sintá(c)tica": a treebank for Portuguese". In: *Proceedings of the 3rd International Conference on Language Resources and Evaluation LREC-2002*, p. 1698–1703, 2002.

[86] BOGUSLAVSKY, I., CHARDIN, I., GRIGORIEVA, S., et al. "Development of a Dependency Treebank for Russian and its possible Applications in NLP". In: *Proceedings of the 3rd International Conference on Language Resources and Evaluation LREC-2002*, p. 852–856, 2002.

[87] OFLAZER, K., SAY, B., HAKKANI-TÜR, D. Z., et al. "Building a Turkish treebank", *Treebanks*, p. 261–277, 2003.

[88] BLAKE, N. *1066–1476*, v. 2, *The Cambridge History of the English Language*. Cambridge University Press, 1992.

[89] HOGG, R. M. *The Beginnings to 1066*, v. 1, *The Cambridge History of the English Language*. Cambridge University Press, 1992.

[90] BAUGH, A. C., CABLE, T. *A History of the English Language*. Prentice Hall, 2002.

[91] CARR, C. T. *Nominal compounds in Germanic*. N. 41, St. Andrews University Publications. Oxford University Press, 1939.

[92] NEEF, M. "A case study in declarative morphology: German case inflection". In: Kehrein, W., Wiese, R. (Eds.), *Phonology and Morphology of the Germanic Languages*, Max Niemeyer Verlag, Tübingen, 1998.

[93] MAIDEN, M. *A linguistic history of Italian*. Longman Publishing Group, 1995.

[94] LOVEDAY, L. J. *Language Contact in Japan: A Sociolinguistic History*. Oxford University Press, 1996.

[95] LI, C. N., THOMPSON, S. A. *Mandarin Chinese: a Functional Reference Grammar*. Berkeley, University of California Press, 1981.

[96] NORMAN, J. L. *Chinese*. Cambridge University Press, 1988.

[97] CHASE, G. D. "The Form of Nominal Compounds in Latin", *Harvard Studies in Classical Philology*, v. 11, pp. 61–72, 1900.

[98] WILLIAMS, E. B. *From Latin to Portuguese: Historical phonology and morphology of the Portuguese language*. University of Pennsylvania Press, 1938.

[99] TOWNSEND, C. *Russian Word-formation*. Columbus, OH, USA, Slavica, 1975.

[100] LEWIS, G. *Turkish Grammar*. Oxford University Press, 2001.

[101] SIMOV, K., OSENOVA, P., SIMOV, A., et al. "Design and Implementation of the Bulgarian HPSG-based Treebank", *Research on Language and Computation*, v. 2, n. 4, pp. 495–522, 2004. doi: 10.1007/s11168-004-7427-z.

[102] HAVERINEN, K., NYBLOM, J., VILJANEN, T., et al. "Building the essential resources for Finnish: the Turku Dependency Treebank", *Language Resources and Evaluation*, v. 48, n. 3, pp. 493–531, 2014. doi: 10.1007/s10579-013-9244-1.

[103] ABEILLÉ, A., CLÉMENT, L., TOUSSENEL, F. "Building a Treebank for French". In: Abeillé, A. (Ed.), *Treebanks: Building and Using Parsed Corpora*, p. 165–187, Dordrecht, Springer Netherlands, 2003. ISBN: 978-94-010-0201-1. doi: 10.1007/978-94-010-0201-1_10.

[104] BAMMAN, D., CRANE, G. "The design and use of a Latin dependency treebank". In: *Proceedings of the Fifth Workshop on Treebanks and Linguistic Theories (TLT2006)*, p. 67–78, 2006.

[105] FUTRELL, R., STEARNS, L., EVERETT, D. L., et al. "A Corpus Investigation of Syntactic Embedding in Pirahã", *PLoS ONE*, v. 11, n. 3, pp. 1–20, 03 2016. doi: 10.1371/journal.pone.0145289.

[106] NGUYEN, P.-T., VU, X.-L., NGUYEN, T.-M.-H., et al. "Building a Large Syntactically-annotated Corpus of Vietnamese". In: *Proceedings of the Third Linguistic Annotation Workshop*, ACL-IJCNLP '09, p. 182–185, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[107] TOMIĆ, O. M. *Balkan Sprachbund Morpho-Syntactic Features*. Dordrecht, Springer Netherlands, 2006. doi: 10.1007/1-4020-4488-7.

[108] KARLSSON, F. *Finnish: An Essential Grammar*. Essential Grammars. Taylor & Francis, 2013. ISBN: 9781134070534.

[109] GREVISSE, M., GOOSSE, A. *Le bon usage: grammaire française*. Duculot, 1993. ISBN: 9782801110454.

[110] BENNETT, C. E. *A Latin Grammar*. Allyn and Bacon, 1895.

[111] FRANK, M. C., EVERETT, D. L., FEDORENKO, E., et al. "Number as a cognitive technology: Evidence from Pirahã language and cognition", *Cognition*, v. 108, n. 3, pp. 819–824, 2008. ISSN: 0010-0277. doi: 10.1016/j.cognition.2008.04.007.

[112] ENFIELD, N. J. "Areal linguistics and Mainland Southeast Asia", *Annual Review of Anthropology*, v. 34, n. 1, pp. 181–206, October 2005. doi: 10.1146/annurev.anthro.34.081804.120406.

[113] AL-RFOU, R., PEROZZI, B., SKIENA, S. "Polyglot: Distributed Word Representations for Multilingual NLP". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 183–192, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[114] ABU-MOSTAFA, Y. N., MAGDON-ISMAIL, M., LIN, H.-T. *Learning From Data*. AMLBook, 2012.

[115] SAUER, N. "On the density of families of sets", *Journal of Combinatorial Theory, Series A*, v. 13, n. 1, pp. 145–147, 1972.

[116] SHELAH, S. "A combinatorial problem; stability and order for models and theories in infinitary languages." *Pacific Journal of Mathematics*, v. 41, n. 1, pp. 247–261, 1972.

[117] MEYER, C. D. *Matrix Analysis and Applied Linear Algebra*. Philadelphia, PA, Society for Industrial and Applied Mathematics, 2000.

[118] HSU, C.-W., LIN, C.-J. "A Comparison of Methods for Multi-class Support Vector Machines", *IEEE Transactions on Neural Networks*, v. 13, n. 2, pp. 415–425, Mar 2002. doi: 10.1109/72.991427.

[119] HUANG, G.-B. "Learning capability and storage capacity of two-hidden-layer feedforward networks", *IEEE Transactions on Neural Networks*, v. 14, n. 2, pp. 274–281, Mar 2003. doi: 10.1109/TNN.2003.809401.

[120] HUANG, G.-B., WANG, D. H., LAN, Y. "Extreme learning machines: a survey", *International Journal of Machine Learning and Cybernetics*, v. 2, n. 2, pp. 107–122, 2011. doi: 10.1007/s13042-011-0019-y.

[121] LIU, X., GAO, C., LI, P. "A comparative analysis of support vector machines and extreme learning machines", *Neural Networks*, v. 33, pp. 58–66, 2012. doi: 10.1016/j.neunet.2012.04.002.

Appendix A

Publications related to the thesis

- CARNEIRO, HUGO C.C.; FRANÇA, FELIPE M.G. ; LIMA, PRISCILA M.V. . Multilingual part-of-speech tagging with weightless neural networks. NEURAL NETWORKS, v. 66, p. 11-21, 2015.
- CARNEIRO, HUGO C.C.; PEDREIRA, CARLOS E. ; FRANÇA, FELIPE M.G. ; LIMA, PRISCILA M.V. . A universal multilingual weightless neural network tagger via quantitative linguistics. NEURAL NETWORKS, v. 91, p. 85-101, 2017.
- CARNEIRO, HUGO C.C.; PEDREIRA, CARLOS E. ; FRANÇA, FELIPE M.G. ; LIMA, PRISCILA M.V. . The exact VC dimension of the WiSARD n-tuple classifier. Submitted to IEEE Transactions on Neural Networks and Learning Systems, 2017.

Neural Networks 66 (2015) 11–21

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Multilingual part-of-speech tagging with weightless neural networks

Hugo C.C. Carneiro [a,*], Felipe M.G. França [a], Priscila M.V. Lima [b]

[a] Systems Engineering and Computer Science Program/COPPE, Universidade Federal do Rio de Janeiro (UFRJ) - Caixa Postal 68511, Cidade Universitária, Rio de Janeiro, Rio de Janeiro 21941-972, Brazil

[b] Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais (NCE), Universidade Federal do Rio de Janeiro (UFRJ) - Av. Athos da Silveira Ramos, 274 - Edifício do Centro de Ciências Matemáticas e da Natureza, Bloco E, Cidade Universitária, Rio de Janeiro, Rio de Janeiro 21941-916, Brazil

ARTICLE INFO

Article history:
Received 12 April 2014
Received in revised form 17 February 2015
Accepted 22 February 2015
Available online 2 March 2015

Keywords:
Weightless neural networks
Part-of-speech tagging

ABSTRACT

Training part-of-speech taggers (POS-taggers) requires iterative time-consuming convergence-dependable steps, which involve either expectation maximization or weight balancing processes, depending on whether the tagger uses stochastic or neural approaches, respectively. Due to the complexity of these steps, multilingual part-of-speech tagging can be an intractable task, where as the number of languages increases so does the time demanded by these steps. WiSARD (**Wi**lkie, **S**tonham and **A**leksander's **R**ecognition **D**evice), a weightless artificial neural network architecture that proved to be both robust and efficient in classification tasks, has been previously used in order to turn the training phase faster. WiSARD is a RAM-based system that requires only one memory writing operation to train each sentence. Additionally, the mechanism is capable of learning new tagged sentences during the classification phase, on an incremental basis. Nevertheless, parameters such as RAM size, context window, and probability bit mapping, make the multilingual part-of-speech tagging task hard. This article proposes mWANN-Tagger (**m**ultilingual **W**eightless **A**rtificial **N**eural **N**etwork **tagger**), a WiSARD POS-tagger. This tagger is proposed due to its one-pass learning capability. It allows language-specific parameter configurations to be thoroughly searched in quite an agile fashion. Experimental evaluation indicates that mWANN-Tagger either outperforms or matches state-of-art methods in accuracy with very low standard deviation, i.e., lower than 0.25%. Experimental results also suggest that the vast majority of the languages can benefit from this architecture.

1. Introduction

Part-of-speech tagging (POS-tagging) is a common task in natural language processing. It requires high accuracy since its result is commonly used as input (or as part of the input) to other tasks, e.g., syntactic parsing and machine translation. Multilingual POS-tagging presents a further challenge. Not only its accuracy must be high in every language, but also the tagger used must have an agile language-independent architecture. Nowadays, two different techniques are used: (i) several POS-taggers are trained independently, which can create some overhead, or (ii) cross-lingual POS-taggers are employed, which use previously annotated relations between words of different *corpora* (composed of texts in different languages) in order to remove tagging ambiguities (Naseem, Snyder, Eisenstein, & Barzilay, 2009; Snyder, Naseem, Eisenstein, & Barzilay, 2008, 2009). In the first case, once a new tagger is needed for a particular language, there is no technique to speed up the parameter tuning procedure. In both strategies, the architecture of the tagger is not truly language-independent. This article proposes a tagger with both a language-independent architecture and the ability to train taggers for new languages with little time spent on parameter tuning procedures.

Neural network models have proven useful in solving natural language processing tasks (Caridakis, Karpouzis, Drosopoulos, & Kollias, 2012; Hinoshita, Arie, Tani, Okuno, & Ogata, 2011; Klein, Kamp, Palm, & Doya, 2010). Neural-based taggers have been proposed since Schmid (1994), some of which employed the neuro-symbolic paradigm, such as Ma, Murata, Uchimoto, and Isahara (2000) and Marques, Bader, Rocio, and Hölldobler (2007). More recently, a weightless neural-based tagger was proposed in Carneiro, França, and Lima (2010). Despite the variety of techniques and parameters adjustment employed, it is observed that every neural tagger created ever since has only been used for monolingual part-of-speech tagging. This work explores the weightless neural

* Corresponding author.
E-mail addresses: hcesar@cos.ufrj.br (H.C.C. Carneiro), felipe@cos.ufrj.br (F.M.G. França), priscilamvl@gmail.com (P.M.V. Lima).

http://dx.doi.org/10.1016/j.neunet.2015.02.012
0893-6080/

Neural Networks 91 (2017) 85–101

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

A universal multilingual weightless neural network tagger via quantitative linguistics

Hugo C.C. Carneiro [a,*], Carlos E. Pedreira [a], Felipe M.G. França [a], Priscila M.V. Lima [b]

[a] Systems Engineering and Computer Science Program/COPPE, Universidade Federal do Rio de Janeiro (UFRJ) - Caixa Postal 68511, Cidade Universitária, Rio de Janeiro, Rio de Janeiro 21941-972, Brazil

[b] Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais (NCE), Universidade Federal do Rio de Janeiro (UFRJ) - Av. Athos da Silveira Ramos, 274 - Edifício do Centro de Ciências Matemáticas e da Natureza, Bloco E, Cidade Universitária, Rio de Janeiro, Rio de Janeiro 21941-916, Brazil

ARTICLE INFO

Article history:
Received 25 November 2016
Received in revised form 4 March 2017
Accepted 18 April 2017
Available online 26 April 2017

Keywords:
Part-of-speech tagging
Weightless neural networks
Zipf's law
Lexical diversity

ABSTRACT

In the last decade, given the availability of corpora in several distinct languages, research on multilingual part-of-speech tagging started to grow. Amongst the novelties there is mWANN-Tagger (**m**ultilingual **w**eightless **a**rtificial **n**eural **n**etwork **tagger**), a weightless neural part-of-speech tagger capable of being used for mostly-suffix-oriented languages. The tagger was subjected to corpora in eight languages of quite distinct natures and had a remarkable accuracy with very low sample deviation in every one of them, indicating the robustness of weightless neural systems for part-of-speech tagging tasks. However, mWANN-Tagger needed to be tuned for every new corpus, since each one required a different parameter configuration. For mWANN-Tagger to be truly multilingual, it should be usable for any new language with no need of parameter tuning. This article proposes a study that aims to find a relation between the lexical diversity of a language and the parameter configuration that would produce the best performing mWANN-Tagger instance. Preliminary analyses suggested that a single parameter configuration may be applied to the eight aforementioned languages. The mWANN-Tagger instance produced by this configuration was as accurate as the language-dependent ones obtained through tuning. Afterwards, the weightless neural tagger was further subjected to new corpora in languages that range from very isolating to polysynthetic ones. The best performing instances of mWANN-Tagger are again the ones produced by the universal parameter configuration. Hence, mWANN-Tagger can be applied to new corpora with no need of parameter tuning, making it a universal multilingual part-of-speech tagger. Further experiments with Universal Dependencies treebanks reveal that mWANN-Tagger may be extended and that it has potential to outperform most state-of-the-art part-of-speech taggers if better word representations are provided.

1. Introduction

Part-of-speech tagging (POS-tagging) is the basis for several other natural language processing tasks. It is important that its accuracy is as high as possible, so to avoid that potential mistakes propagate in cascade to the following tasks. There are two major issues that prevent part-of-speech tagging from being a fully straightforward process, they are homonymy and unknown words. The former consists of a word having at least two different meanings (or parts of speech) while the latter are words that need to be tagged but were not seen during the training phase.

Those issues arise from the basic nature of languages and how they encode information into words. Some languages have a small lexicon, mostly with words of atomic meaning, and build sentences using many of them in a very fixed order. Other languages prefer to have a freer word order, but their words tend to have more complex meanings and new ones can always be coined by appending affixes to other existing words.

The languages of the former group are known as isolating and the ones of the latter as synthetic (Sapir, 1921). A same language can encode distinct pieces of information in very different ways. This way, languages can be classified in a syntheticity spectrum that range from fully isolating languages at one end to fully synthetic at the other. It is important to know where a language lies

* Corresponding author.
E-mail addresses: hcesar@cos.ufrj.br (H.C.C. Carneiro), pedreira56@gmail.com (C.E. Pedreira), felipe@cos.ufrj.br (F.M.G. França), priscilamvl@gmail.com (P.M.V. Lima).

http://dx.doi.org/10.1016/j.neunet.2017.04.011
0893-6080/

The Exact VC Dimension of the WiSARD n-tuple Classifier

Hugo C. C. Carneiro, Carlos E. Pedreira, Felipe M. G. França and Priscila M. V. Lima

***Abstract*—WiSARD n-tuple classifier is a multicategorical weightless neural network. Each category is represented by a structure called discriminator. It has N nodes, composed of memory positions, where the model stores its learned knowledge. They are addressed by n-tuples, and so their stored content can be retrieved. This architecture allows WiSARD to learn a pattern in a single pass. In other words, the system does not depend on convergence. However, its fast training was counterbalanced by the saturation problem, which arose when the network was exposed to large amount of data. Previous studies were done concerning the generalization capability of the learning system, in order to better understand it and come up with solutions to its saturation. They did not find the exact value of the VC dimension of the traditional n-tuple classifier, but found tight bounds for it nevertheless. Recently, the bleaching technique was proposed as a means to avoid saturation. Applications with large amount of data showed that the bleaching n-tuple classifier is able to prevent it, achieving high accuracies with low variance. Despite its competitive performance, no theoretical studies had been conducted with this extension yet. This paper aims to push forward the research on the theoretical foundations of the bleaching technique and how it improves the generalization capability of the n-tuple classifier. This article presents the calculation of the exact VC dimension of the traditional model, which is linearly proportional to the amount of nodes and exponentially to their addressing tuple length, precisely $N\left(2^n - 1\right) + 1$. It also introduces the VC dimension of the bleaching model, whose value is exactly the same as that of the traditional one, demonstrating that the bleaching technique is an enhancement to the n-tuple method that does little to no harm to its generalization capability.**

***Index Terms*—**

I. Introduction

The n-tuple classifier is a weightless neural model, initially proposed in 1959 [1] and formally defined in 1962 [2]. It is a one-pass learning multicategory pattern recognizer, whose learned knowledge is stored on memory matrices. A practical implementation made in 1982 [3], [4], the **Wi**lkie, **S**tonham and **A**leksander **R**ecognition **D**evice (**WiSARD**), showed that it was actually possible to assemble the n-tuple classifier.

Theoretical researches and architecture improvements were performed on that learning model [5]–[24]. One of the main issues raised by those studies was how to find a means to mitigate saturation, a common problem of WiSARD n-tuple classifier. As the recognition machine is fed with (especially if noisy) data, it starts filling up every position of the memory matrices, deteriorating its capability of discriminating patterns.

Among the studies intended to lessen the effects of saturation, it is worth mentioning the work of Bradshaw [12], where lower and upper bounds for the VC dimension of WiSARD n-tuple classifier were calculated. No exact value for this measure was found. The research made by Bradshaw [12] provided a fertile ground for further analyses on this field. Unfortunately, the solution to the saturation problem there proposed was not that successful, as it relied on convergence and had no guarantee that it would actually happen whatsoever.

In 2010 [23] an actual way of mitigating saturation was devised, called the bleaching technique. It allowed the network to be exposed to loads of data (noisy and unnoisy) and yet keep the fidedignity of its pattern discrimination capability. This improvement considerably enhanced both accuracy and precision of traditional WiSARD n-tuple classifier applications with no performance harm on their training procedures and just a small bit on the classification step [25]–[29].

That technique granted the WiSARD model competitiveness with trending learning systems, by achieving high accuracies with low variance in experimental works. However, there lacks a theoretical background for bleaching. This paper aims to provide a mathematical foundation for that extension of the weightless classifier, by analyzing the generalization capacity of both traditional and bleaching recognition schemes.

The paper structure is divided as follows. Section II introduces some basic definitions of the VC theory and presents WiSARD and bleaching n-tuple classifiers. A formal mathematical definition of both models is also provided. Their VC dimensions are calculated in Sections III and IV. The results thereof are then discussed in Section V, where some conclusions are drawn and comparisons with weighted learning schemes are made. Section VI summarizes the work presented in this paper and proposes future works.

II. Basic concepts

Some VC theory basic concepts and a brief introduction to both architectures of the WiSARD model are presented in this Section. They are provided as a cornerstone for the calculations yet to come.

A. VC theory definitions

To measure the capacity of a learning system, like the WiSARD network, the definitions of some VC theory basic

H. C. C. Carneiro, C. E. Pedreira and F. M. G. França are with Systems Engineering and Computer Science Program - PESC/COPPE, Universidade Federal do Rio de Janeiro (UFRJ) - Caixa Postal 68511, Cidade Universitária, Rio de Janeiro, RJ, 21941-972 Brazil - e-mails: hcesar@cos.ufrj.br, pedreira56@gmail.com and felipe@cos.ufrj.br.

P. M. V. Lima is with Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais - iNCE, Universidade Federal do Rio de Janeiro (UFRJ) - Av. Athos de Silveira Ramos, 274 - Edifício do Centro de Ciências Matemáticas e da Natureza, Bloco E, Cidade Universitária, Rio de Janeiro, RJ, 21941-916 Brazil - e-mail: priscilamvl@gmail.com.

Appendix B

Further publications as a D.Sc. student

- CARVALHO, D.S. ; CARNEIRO, H.C.C. ; FRANCA, F.M.G. ; LIMA, P.M.V. . B-bleaching: Agile Overtraining Avoidance in the WiSARD Weightless Neural Classifier. In: 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2013, Bruges. Proceedings. of ESANN 2013. Brussels: i6doc.com, 2013. v. 1. p. 515-520.

- SOUZA, DIEGO F.P. DE ; CARNEIRO, HUGO C.C. ; FRANCA, FELIPE M.G. ; LIMA, PRISCILA M.V. . Rock-Paper-Scissors WiSARD. In: 2013 BRICS Congress on Computational Intelligence & 11th Brazilian Congress on Computational Intelligence (BRICSCCI & CBIC), 2013, Ipojuca. 2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence. v. 1. p. 178-183.

- ALVES, DANIEL S.F. ; CARDOSO, DOUGLAS O. ; CARNEIRO, HUGO C.C. ; FRANCA, FELIPE M.G. ; LIMA, PRISCILA M.V. . An Empirical Study of the Influence of Data Structures on the Performance of VG-RAM Classifiers. In: 2013 BRICS Congress on Computational Intelligence & 11th Brazilian Congress on Computational Intelligence (BRICSCCI & CBIC), 2013, Ipojuca. 2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence. v. 1. p. 388-393.

- VIDAL, F.S. ; CARNEIRO, H.C.C. ; ROSA, P.F.F. ; FRANCA, F.M.G. . Identificação de emoções a partir de expressões faciais com redes neurais sem peso. In: Simpósio Brasileiro de Automação Inteligente, 2013, Fortaleza. Anais do XI Simpósio Brasileiro de Automação Inteligente (SBAI 2013), 2013. p. 1-7.

- CARDOSO, D.O. ; CARVALHO, D.S. ; ALVES, D.S.F. ; SOUZA, D.F.P. ;

CARNEIRO, H.C.C. ; PEDREIRA, C.E. ; LIMA, P.M.V. ; FRANCA, F.M.G. . Credit Analysis with a clustering RAM-based neural classifier. In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2014, Bruges. Proceedings of ESANN 2014. Brussels: i6doc.com, 2014. p. 517-522.

- COUTINHO, P.V.S. ; CARNEIRO, H.C.C. ; CARVALHO, D.S. ; FRANCA, F.M.G. . Extracting rules from DRASiW's "mental images". In: European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 2014, Bruges. Proceedings of ESANN 2014. Brussels: i6doc.com, 2014. p. 529-534.

- CARDOSO, DOUGLAS O. ; CARVALHO, DANILO S. ; ALVES, DANIEL S.F. ; SOUZA, DIEGO F.P. ; CARNEIRO, HUGO C.C. ; PEDREIRA, CARLOS E. ; LIMA, PRISCILA M.V. ; FRANÇA, FELIPE M.G. . Financial credit analysis via a clustering weightless neural classifier. NEUROCOMPUTING, v. 183, p. 70-78, 2016.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

B-bleaching: Agile Overtraining Avoidance in the WiSARD Weightless Neural Classifier

Danilo S. Carvalho[1], Hugo C. C. Carneiro[1], Felipe M. G. França[1], Priscila M. V. Lima[2] *

1- Universidade Federal do Rio de Janeiro - PESC/COPPE
Rio de Janeiro, Brazil

2- Universidade Federal Rural do Rio de Janeiro - PPGMMC/DEMAT
Seropédica, Brazil

Abstract. Weightless neural networks constitute a still not fully explored Machine Learning paradigm, even if its first model, WiSARD, is considered. Bleaching, an improvement on WiSARD's learning mechanism was recently proposed in order to avoid overtraining. Although presenting very good results in different application domains, the original sequential bleaching and its confidence modulation mechanisms still offer room for improvement. This paper presents a new variation of the bleaching mechanism and compares the three strategies performance on a complex domain, that of multilingual grammatical categorization. Experiments considered both number of iterations and accuracy. Results show that binary bleaching allows for a considerable improvement to number of iterations whilst not introducing loss of accuracy.

1 Introduction

As the areas of application of Artificial Intelligence expand, so do the demand for speed and accuracy in classification and training techniques. Moreover, many situations require that training be interleaved with classification, in an online learning fashion. Though not a recently proposed paradigm, weightless neural networks (WNNs) are still not fully explored [1]. WNNs first model, WiSARD [2], possesses the ability of performing online training. However, it often suffers from overtraining after a not so big set of examples. WiSARD's learning mechanism has been recently improved by the addition of a process called *bleaching* [3]. The original sequential bleaching and its confidence modulation mechanisms presented promising results in different application domains [3] [4]. There is still, however, room for improvement.

The following is how the remainder of the text is organised. Background knowledge on WiSARD and bleaching is briefly reviewed in Section 2. Section 3 presents a new variation of the bleaching mechanism and Section 4 provides a quantitative comparison of the three strategies performance. The problem of multilingual grammatical categorization of ambiguous words was the chosen complex domain chosen for the comparison. Section 5 provides some concluding remarks as well as points to future research steps.

*This work was partially supported by CAPES, CNPq and FAPERJ Brazilian research agencies.

Rock-paper-scissors WiSARD

Diego F. P. de Souza
Hugo C. C. Carneiro
and Felipe M. G. França
System Engineering and Computer Science Program - COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil
Email: diegosouza@cos.ufrj.br
hcesar@cos.ufrj.br
felipe@cos.ufrj.br

Priscila M. V. Lima
Department of Mathematics - ICE
Universidade Federal Rural do Rio de Janeiro
Seropédica, Brazil
Email: priscilamvl@gmail.com

***Abstract*—This paper presents some strategies used for creating intelligent players of rock-paper-scissors using WiSARD weightless neural networks and results obtained therewith. These strategies included: (*i*) a new approach for encoding of the input data; (*ii*) three new training algorithms that allow the reclassification of the input patterns over time; (*iii*) a method for dealing with incomplete information in the input array; and (*iv*) a bluffing strategy. Experiments show that, in a tournament of intelligent agents, WiSARD-based agents were ranked among the 200 best players, one of them achieving 9th place for about three weeks.**

***Keywords*—Weightless neural networks, game, bots, intelligent agents, adaptiveness**

I. Introduction

A common requirement for the development of intelligent agents is the need to adapt their behavior according to the actions of other agents. As seen in [1], such a behavior is especially important for the gaming industry, in which it is desirable to create players adaptable to the strategy of each opponent, increasing this way the realism and difficulty of the game. In general, strategy games, whether electronic or not, tend to provide a good environment for the study and development of such agents. Rock-paper-scissors [2], [3], in special, constitutes a quite interesting game, because its simplicity would be analogous to situations in which agents do not have a wide range of action possibilities. Furthermore, this game analogy has been successfully applied to represent several communities of subpopulations [4], [5], [6], [7]. In this work, we create adaptive agents who can play this game, predicting its opponent strategy and defeating it.

This work assumes that every move performed in a rock-paper-scissors game is influenced by the recent results in the game and, therefore, can be inferred by an adaptive intelligent agent. As it is easily seen, in cases where at least one of the two players uses randomness as a strategy, the number of wins of both players will be on average equal to one third of the number of rounds played, thereby generating a result very close to a draw. Several machine learning techniques have been adopted to provide agents with this adaptiveness [8], [9], but weightless artificial neural networks (WANNs) had not yet been applied in this context.

Because it is a mechanism that combines a high potential adaptiveness and an extremely simple architecture [10], [11], [12], the WiSARD neural network has been chosen as the basic paradigm in the proposal of rock-paper-scissors players/agents. The network shall receive as input the game history of last H rounds and thus try to predict the next move of its opponent.

The major problem of the application of that model to this context, is that the game strategies of the opponents tend to change over time. Thus it is necessary to readjust the knowledge of the network according to opponent strategy. For this reason, this work uses variations during the network training so as to allow that its classification for a particular input pattern can evolve with time.

The remainder of this paper is organized as follows. After this introduction, the architecture and functioning of the WiSARD neural model are summarized in Section II. Sections III and IV present, respectively, the strategies used to create the agents and experimental results. Section V concludes the text pointing at future steps of this research.

II. The WiSARD Neural Model

Technological advances on the area of integrated circuits that occurred in the '70s, enabled Wilkie, Stonham and Aleksander to create a general-purpose device, composed of small units of RAM (Random Access Memory). This device, called WiSARD [10], [12], was capable of recognizing and classifying different patterns which were presented to it, with a certain degree of generalization.

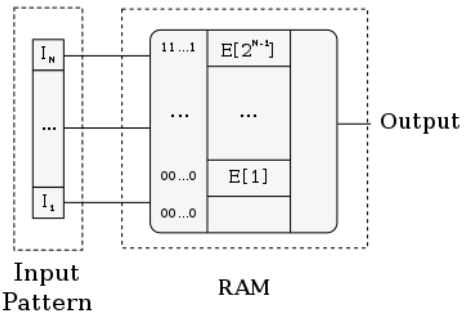


Fig. 1. A RAM node with an input pattern of N bits.

An empirical study of the influence of data structures on the performance of VG-RAM classifiers

Daniel S. F. Alves, Douglas O. Cardoso,
Hugo C. C. Carneiro and Felipe M. G. França
PESC - COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brazil

Priscila M. V. Lima
DEMAT - ICE
Universidade Federal Rural do Rio de Janeiro
Seropédica, Brazil

***Abstract*—This work investigates the effect of different data structures on the performance and accuracy of VG-RAM-based classifiers. This weightless neural model is based on RAM nodes having very large address input, what suggests the use of special data structures in order to deal with space and time computational costs. Four different data structures are explored, including the classical one used in recent VG-RAM related literature, resulting in a novel and accurate yet fast setup.**

***Keywords*—VG-RAM, weightless neural networks, data structures.**

I. Introduction

One of the most common activities in machine learning is the classification of data. Among other techniques, artificial neural networks are often used for this purpose. A particular subset of this group, weightless artificial neural networks (WANNs) [1]–[3], distinguishes itself by approaching the biological inspiration from a different point of view than that of McCullogh and Pitts [4]. WANNs are based on the collective response of artificial neurons inspired by the functionality of the dendritic tree, a massive decoding structure found on most neuron cells, that are interpreted as memory modules, or nodes.

The relation between memory nodes and its internal functionality influences the features of a particular WANN model. This work was focused on the Virtual Generalizing Random Access Memory (VG-RAM) [1], [3], [5] whose properties include a great generalization capacity of stored data without the expense of writing extra memory locations, as needed by similar memory modules. But this comes at the cost of extra runtime and higher implementation complexity. How these characteristics vary depends on the operation of the internals of the VG-RAM node. This work proposes to study some of the possibilities to this and compare their observed executions.

The remainder of the paper is organized as follows. Section II shows the basic version of a VG-RAM node and the generic VG-RAM classifier architecture. Section III presents the alternative versions proposed in this paper. The introduced VG-RAM classifier architecture was used to experiment on the new versions proposed and to compare with the original version. Such experiments and comparison are discussed in section IV. Section V concludes the paper pointing to future topics of investigation.

II. Classification using VG-RAM

Random access memory (RAM) modules are one of the most basic building blocks in modern computing. In a simple manner, the way a RAM module works can be described by two actions: write(x, y), which records the value y in the location x; and read(x), which returns the last value recorded in x. We will consider that each location is addressed exclusively by a value of m bits, where m is a parameter of each RAM instance. From this description it is possible to see a RAM module as an associative array [6], another very common structure in computer systems.

GRAM [1] (G stands for generalizing) extends the traditional RAM with the idea of *spreading*: when the action write(x, y) is performed, not only the location x is written with y but this value is spread to neighbouring locations according to the Hamming distance to x, similarly to a circular wave. The spreading in a direction may stop because: (i) all related locations were recorded, (ii) a location written before with a value that is not y was reached or (iii) a given maximum distance threshold was exceeded.

The generalization capability of a GRAM node could also be provided without spreading. Whenever a read(x) targets an unwritten location, it returns the value of the closest written location. The proximity criteria adopted consists of the Hamming distance. This is how VG-RAM [1], [7](V stands for *virtual*) nodes work. Virtual means that the node only stores the written locations and its values, while still behaving like the GRAM.

Using any VG-RAM conception, a network can be built for classification. While there are advanced architectures such as the GNU and MAGNUS [7], the most traditional structure of a neural network of VG-RAM nodes is similar to that of a WiSARD discriminator [1], [3], an array of nodes whose results are combined to define a result for the network. However, it has been shown that a simple layer of VG-RAM nodes can achieve performance similar to that of the WiSARD network with a smaller storage penalty, albeit a larger time penalty [7].

The way a RAM node works was described by write(x, y) and read(x). Showing how a VG-RAM classifier works can be done in a similar manner, using train(a, b) and classify(a), which we present next (algorithms 1 and 2, respectively), as pseudocode, considering that the classifier has n VG-RAM nodes of m-bits addresses.

IDENTIFICAÇÃO DE EMOÇÕES A PARTIR DE EXPRESSÕES FACIAIS COM REDES NEURAIS SEM PESOS

Fábio Silveira Vidal* † , Hugo Cesar de Castro Carneiro‡ , Paulo Fernando Ferreira Rosa* , Felipe Maia Galvão França‡

*Instituto Militar de Engenharia
Praça General Tibúrcio, 80 - Urca
Rio de Janeiro, RJ, Brasil

†Instituto Federal do Tocantins
Distrito Agroindustrial, BR 153, KM480
Paraíso do Tocantins, TO, Brasil

‡Universidade Federal do Rio de Janeiro
Av. Horácio Macedo, 2030, Prédio do Centro de Tecnologia, Bloco H - Cidade Universitária
Rio de Janeiro, RJ, Brasil

Email: vidalfs@gmail.com, hcesar@cos.ufrj.br, rpauloime@gmail.com, felipe@cos.ufrj.br

Abstract— This paper describes a study of neural networks without weights application for recognizing emotions. Task is performed from facial expressions image analysis. Eight situations were considered: anger, contempt, disgust, fear, happiness, neutral, sadness, and surprise. Experiments were made from TFEID (Taiwanese Facial Expression Image Database). Results shown that presented method is suitable to solve the recognizing emotions problem. Various training configurations were tested. Best returned hit rate above 95%.

Keywords— Computer Vision, Pattern Recognition, Emotion Recognition, Neural Network without Weights, Natural Interfaces

Resumo— Este trabalho descreve um estudo da aplicação de redes neurais sem pesos para o reconhecimento de emoções. A tarefa é feita a partir de análise de imagens de expressões faciais. Oito situações foram consideradas: raiva, desprezo, nojo, medo, felicidade, neutro, tristeza e surpresa. Os experimentos foram feitos a partir da base de dados TFEID (*Taiwanese Facial Expression Image Database*). Os resultados mostraram que a metodologia é adequada para a resolução do problema. Várias configurações de treinamento foram testadas. As melhores retornaram taxa de acerto acima de 95%.

Keywords— Visão Computacional, Reconhecimento de Padrões, Reconhecimento de Emoções, Redes Neurais sem Pesos, Interfaces Naturais

1 Introdução

O desenvolvimento tecnológico e o crescimento de poder de processamento proveem diferentes formas de interação com o computador. O reconhecimento automático de emoções viabiliza que uma máquina possa reagir de forma mais adequada ao estado emocional do usuário ou de pessoas em torno de uma máquina. Esta é uma forma de desenvolver interfaces naturais e pró-ativas, sendo uma alternativa ao uso do tradicional método de respostas a comandos por meio de hardwares como teclado, *mouse* ou *joysticks*.

O reconhecimento automático de expressões faciais vem ganhando importância para o desenvolvimento de interfaces interativas, podendo contribuir em diversas aplicações como: ambientes virtuais de aprendizagem, sistemas de segurança, casas inteligentes ou aplicações que visem acessibilidade a pessoas portadoras de necessidades especiais. O objetivo é fazer com que um sistema aja de acordo com reações de usuários ou de indivíduos a serem observados.

Uma solução de reconhecimento de expressões faciais pode passar por 3 (três) etapas: (1) localização da face; (2) segmentação de regiões de interesse; e (3) identificação da emoção. O escopo deste trabalho foca apenas a última etapa. Nos experimentos, as duas primeiras foram feitas de modo manual.

Vários métodos de reconhecimento de padrões podem ser adotados para identificar a emoção de uma pessoa, como Lógica Fuzzy, Redes Neurais Artificiais, *Support Vector Machine* (SVM), ou Redes Bayesianas. Neste trabalho, utilizou-se uma Rede Neural sem Pesos. Para tal escolha, considerou-se o processo de treinamento, o qual é mais rápido e mais simplificado que outros modelos de redes neurais, como o o MLP (*Multilayer Perceptron*).

Nas duas próximas seções, alguns trabalhos relacionados e opções de base de dados de imagens são referenciados. Em seguida, são descritos os processos de pré-processamento das imagens e o treinamento da rede neural sem pesos. Por fim, apresenta-se a validação e os resultados, e são feitas as considerações finais.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Credit analysis with a clustering RAM-based neural classifier

Douglas O. Cardoso[1], Danilo S. Carvalho[1], Daniel S. F. Alves[1],
Diego F. P. Souza[1], Hugo C. C. Carneiro[1],
Carlos E. Pedreira[1], Priscila M. V. Lima[2] and Felipe M. G. França[1] *

1 - COPPE, 2 - iNCE, Universidade Federal do Rio de Janeiro, BRAZIL

Abstract. Datasets with a large amount of noisy data are quite common in real-world classification problems. Robustness is an important characteristic of state-of-the-art classifiers that use error minimization techniques, thus requiring a long time to converge. This paper presents ClusWiSARD, a clustering customization of the WiSARD weightless neural network model, applied to credit analysis, a non-trivial real-world problem. Experimental evidence show that ClusWiSARD is very competitive with Support Vector Machine (SVM) w.r.t. accuracy, with the difference of being capable of online learning. Nonetheless, it outperforms SVM in both training time, being two orders of magnitude faster, and test time, being slightly faster.

1 Introduction

Data with concept drift increases the complexity of classifiers, since information learnt is likely to degrade classification performance over time, as both feature patterns and target functions can change. This is the case of credit analysis, a recurring problem in the banking business which can be summarized as deciding which requests for credit should be granted. The usual process involves the collection of data, which is used to determine the "quality" of the request, in other words, the risk of a borrower failing to pay his or her debts. The analysis of these data, however, is a complex problem.

One of the aspects which makes this problem considerably harder is the change of patterns over time. The movement of populations, changes in economy, natural catastrophes [1], general news [2], among other factors which may directly affect the relations pertinent to credit. Another aspect to be considered is the bias of the available data: only data about granted requests are usually stored. This means there is not enough information about the bad payers.

An automated learning and classification mechanism could offer a more precise solution, being able to analyse vast amounts of data on credit applications and consider subtle relations between the actual financial data and the borrower profile. Those methods would need to be efficient and robust in order to account for changes in the circumstances and sample biasing. Two classifying mechanisms which exhibit these characteristics are the WiSARD [3] artificial neural network model and the Support Vector Machine (SVM) [4], which we introduce

*This work was partially supported by GE, Inovax, and CAPES, CNPq and FAPERJ Brazilian research agencies.

ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.
Available from http://www.i6doc.com/fr/livre/?GCOI=28001100432440.

Extracting rules from DRASiW's "mental images"

Paulo V. S. Coutinho, Hugo C. C. Carneiro, Danilo S. Carvalho, Felipe M. G. França *

Universidade Federal do Rio de Janeiro - PESC/COPPE
Rio de Janeiro, Brazil

Abstract. DRASiW is an extension of the WiSARD weightless neural model that provides the ability of producing examples/prototypes, called "mental images", from learnt categories. This work introduces a novel way of performing rule extraction by applying the WiSARD/DRASiW RAM-based neural model upon a well-known machine learning benchmark. A functional exploration is offered in order to demonstrate how the new rule extraction mechanism behaves under different system configurations. Experimental results suggest that the rules conformance to data increases proportionally to the corresponding classifier accuracy. Furthermore, comparison with C4.5 decision tree algorithm shows that the DRASiW-based technique produces more compact sets of rules.

1 Introduction

DRASiW [1] [2] is an extension to the WiSARD weightless neural model capable of producing approximated examples of learnt categories, the so-called "mental images". This work introduces a novel way of performing rule extraction (production rules) from "mental images" produced by a WiSARD/DRASiW multidiscriminator trained with the *Iris* machine learning dataset [3, 4]. In order to demonstrate how the proposed rule extraction mechanism works, the system is stressed under different configurations. The remainder of the text is organised in the following manner. Background knowledge on WiSARD, DRASiW and bleaching is briefly reviewed in Section 2. The explanation of how rules can be extracted by the use of DRASiW is presented in Section 3. Section 4 provides experiments with a well-known benchmark and a detailed analysis in order to demonstrate the capabilities of DRASiW. Finally, in Section 5 some concluding points are drawn and some further research challenges are presented.

2 WiSARD and Bleaching

WiSARD (**Wi**lkie, **S**tonham and **A**leksander's **R**ecognition **D**evice) [5] is both a weightless neural network and an n-tuple classifier, i.e., its input is an array of bits. The basic structure of its architecture is a RAM (random access memory) discriminator. Each of these structures is assigned to a particular category, therefore a WiSARD network possesses as many discriminators as the number

*This work was partially supported by Inovax, and CAPES, CNPq and FAPERJ Brazilian research agencies.

Neurocomputing 183 (2016) 70–78

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Financial credit analysis via a clustering weightless neural classifier

Douglas O. Cardoso [a], Danilo S. Carvalho [a], Daniel S.F. Alves [a], Diego F.P. Souza [a,*], Hugo C.C. Carneiro [a], Carlos E. Pedreira [a], Priscila M.V. Lima [b], Felipe M.G. França [a]

[a] PESC/COPPE - Universidade Federal do Rio de Janeiro - UFRJ, Brazil
[b] NCE, Instituto Tércio Pacitti, Universidade Federal do Rio de Janeiro, Brazil

ARTICLE INFO

Article history:
Received 31 August 2014
Received in revised form
5 April 2015
Accepted 2 June 2015
Available online 9 January 2016

Keywords:
Bleaching
ClusWiSARD
Clustering
Concept drifting
Credit assignment
Online learning

ABSTRACT

Credit analysis is a real-world classification problem where it is quite common to find datasets with a large amount of noisy data. State-of-the-art classifiers that employ error minimisation techniques, on the other hand, require a long time to converge, in order to achieve robustness. This paper explores ClusWiSARD, a clustering customisation of the WiSARD weightless neural network model, applied to two different credit analysis real-world problems. Experimental evidence shows that ClusWiSARD is very competitive with Support Vector Machine (SVM) w.r.t. accuracy, with the advantage of being capable of online learning. ClusWiSARD outperforms SVM in training time, by two orders of magnitude, and is slightly faster in test time.

1. Introduction

Credit analysis represents the complex tasks of deciding which credit applicants present a good probability of returning the granted credit and which do not. This task depends on many different factors, such as economic and cultural circumstances, and is often delegated to human experts. Human judgement, however, may not use explicit rules that can be referenced as basis for decision making. That could lead to conflicting analysis of the same problem instance from different experts. In some countries, this is considered illegal. This question would justify the design of a machine learning system that is able to replace the decisions of experts, providing a single analysis standard.

Important pattern recognition challenges can be found in credit analysis. For example, data can be noisy or corrupted due to problems in data collection. Data could also embed temporal information, possibly useful to identify *concept drift*: movement of populations, changes in economy, natural catastrophes [1], general news [2], etc. These and other factors may affect the relations pertinent to credit assignment. Class imbalance is also expected, as credit applications labeled as “good” are more frequent than “bad” ones.

How observations were gathered and labeled is also noteworthy. Labelling could be done *a priori*, according to a risk appraisal system already in use. Alternatively, this could be performed after observing if payment of granted requests was duly realised. A system trained with data from the first case aims to reproduce the behaviour of the established classification system, instead of attempting to excel it. In the second case, training data is the product of a filtering process, implying in a reduction of information about the population.

Different machine learning techniques have been analysed in the context of this problem. As discriminated by Tsai [3], they may be classified in three smaller sets, which are: single classifiers, classifier ensemble and hybrid classifiers. The first one contains single supervised models, like Support Vector Machine (SVM) [4–8], Multilayer-Perceptron (MLP) [9,4,10], Decision Trees (DT) [11] and Genetic Algorithm/Programming (GA/GP) [12,13]. Regarding classification accuracy over the UCI dataset, which was also used in this work, some results obtained were 77.34% by Ong, Huang and Tzeng [13] with the use of GP and 77.09%/76.59% by Tsai [4] with SVM and MLP, respectively. These models have achieved at most an accuracy of 77.34% working as single classifiers. However, they may achieve much better results when grouped together, forming classifier ensembles. For instance, Ghodselahi [14] has obtained 81.42% with the use of a SVMs ensemble, and Hoffmann [15] has reached 84.90% with a GA-based SVM. Some other approaches used GA-based MLP [16] and GA-based SVM [17] in other financial credit analysis datasets. The third category, called Hybrid Classifiers, contains approaches mixing two or more techniques. For instance, combining clustering and single classifiers. Previously a work [3] compared many different approaches

* Corresponding author.
E-mail address: diegofpsouza@gmail.com (D.F.P. Souza).

http://dx.doi.org/10.1016/j.neucom.2015.06.105
0925-2312/