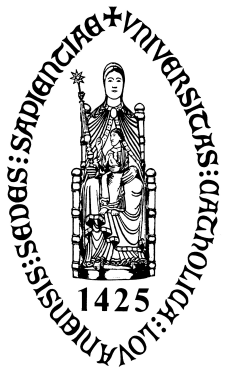
XAVIER GOAS AGUILILLA

MACHINE LEARNING FOR ANCIENT GREEK LINGUISTICS

A TENTATIVE METHODOLOGY AND APPLICATION TO THE CORPUS OF DOCUMENTARY PAPYRI



Master's thesis submitted in partial fulfilment of the requirements for the degree of

MASTER OF ARTS IN DE TAAL- EN LETTERKUNDE

Supervisor: dr. T. Van Hal
KU Leuven
Faculty of Arts
Department of Greek Studies

Leuven, 2013

E-MAIL:
xavier.goas@student.kuleuven.be

ABSTRACT

SAMENVATTING

Deze masterproef bevat **39686** tekens.

PREFACE

ACKNOWLEDGEMENTS

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 THESIS STATEMENT

The following work is concerned with the application of techniques from the field of artificial intelligence and machine learning to an ancient Greek corpus, namely that of the documentary papyri.

It intends to prove that it is possible to use these techniques, which are generally applied to English and other modern languages, to generate linguistic annotation for this corpus in an accurate and efficient manner.

In order to achieve this, I have implemented the architecture proposed in Collobert & Weston 2008 and Collobert et al. 2011. This architecture relies on cutting-edge techniques in machine learning and offers the possibility of using data devoid of linguistic annotation as part of the training corpus. After implementing this architecture, I have taken a sample from fully tagged texts and used it to check the general accuracy of the system. I have done the same for the papyri, albeit manually and using a smaller sample due to the fact that there is no preexisting annotation for the corpus of documentary papyri.

1.2 MOTIVATION

The study of the language of the papyri has in the past thirty years seen little evolution until the recent appearance of Evans and Obbink's *The Language of the Papyri* [Evans and Obbink, 2010], which has placed the subject in the spotlight again. Twentieth-century scholarship on the topic, though still useful for those interested in the study of the papyri for historical purposes, is either antiquated, limited in scope or incomplete (see *infra*). Despite this, the papyri are useful source material for the history and evolution of the Greek language, as they contain not only official texts but private documents as well, whose linguistic features and peculiarities have the potential to foster new insights into the nature of colloquial Greek. Such a corpus could be a boon to scholars interested in the Greek of the papyri, as it would facilitate, for instance, the creation of linguistically sound grammars and lexica.

Furthermore, applying the same methodology to other Greek texts offers new linguistic perspectives. The largest corpus of ancient Greek, the Thesaurus Linguae Graecae, contains more than a hundred million words. A system is in place which offers morphological and lemmatic analysis in a rudimentary form, but this cannot serve as a full linguistic corpus. Using manual methods to tag this corpus is rather prohibitive

due to its size, but using computational methods, we can make an attempt at offering a relatively complete linguistic corpus for ancient Greek.

I also wish to demonstrate the potential of a methodology based on computational techniques for the study of the classical languages. The field has seen a move towards digitalisation in the past half century, but a lot of potential is left untapped. Steps in the right direction are currently being made, but we can drive progress much farther. The classicist breed does not number many specimens; and though the true classics, the Homers, the Platos, the Virgils, have been subjected to thorough analysis for millennia, the amount of texts in need of scholarly attention remains large. Demonstrating the potential of computational methods for Greek linguistics will hopefully serve as further proof of their potential for other branches of classics, such as stylometry, authorship verification, textual criticism, and more.

1.3 GOALS AND OBJECTIVES

1.4 CONTRIBUTIONS

1.5 OUTLINE

I begin by giving an overview of the background for this thesis. First the historical and linguistic background of the question is handled, expounding on previous efforts to study the grammar of the papyri and to apply the techniques of corpus linguistics to the Greek language in general. Second comes a technical section, describing a set of core concepts and techniques which are relevant to the task at hand and are necessary to understand the underpinnings of the applied methodology.

This is followed by an analysis of the objectives and requirements, as well as a general (that is, only described in broad strokes, without providing full details of the algorithms and implementation) methodology. Correspondingly, a chapter is dedicated to the general algorithmic design and structure of our program, followed by a chapter delving into the actual implementation, which provides more details on the choice of programming language, the availability of the source code, the technical requirements for running our code, etc.

We then offer an interpretation and a critical assessment of our program's output. Special attention is, of course, given to the linguistic accuracy of our results. A final chapter is dedicated to an overview of further possibilities for research in this field and of possible applications outside papyrology and to the field of Greek linguistics in general.

2 BACKGROUND

2.1 HISTORICAL BACKGROUND

2.1.1 The language of the papyri

The papyri began to be studied linguistically not by papyrologists and historians, but rather by Bible scholars and grammarians interested in their relevance in the development began to koinê Greek, particularly that of the New Testament. G. N. Hatzidakis, W. Crönert, K. Dieterich, A. Deissmann, and A. Thumb pioneered the field in the late nineteenth and early twentieth century, spurring a resurgence of scholarship on the topic; an excellent overview of pre-1970s research may be found in Mandilaras [1973] and Gignac [1976 and 1981].

During this period, Mayser begon work on the earliest compendious grammar of the papyri; it limits itself to the Ptolemaic era but explores it at length and in great detail. The work consists of a part on phonology and morphology, made up of three slimmer volumes, and a part on syntax, encompassing three larger volumes. Its composition seems to have been exhausting: it took Mayser thirty-six years to finish volumes I.2 through II.3, with I.1 only completed in 1970 by Hans Schmoll, at which point the entire series was given a second edition.

When casually browsing through some of its chapters (though casual is hardly the word one would associate with the *Grammatik*) it is remarkable to see that Mayser brings an abundance of material to the table for each grammatical observation he makes, however small it may be. For instance, the section on diminutives essentially consists of pages upon pages of examples categorised by their endings.

This is its great strength as a reference work - whenever one is faced with an unusual grammatical phenomenon in any papyrus, consulting Mayser is bound to clarify the matter; or rather, it was, for the work is now inevitably dated. The volumes published during Mayser's lifetime only include papyri up to their date of publication; only the first tome by Schmoll includes papyri up to 1968. It is still a largely useful resource, but it is in urgent need of refreshment.

After Mayser set the standard for the Ptolemaic papyri, a grammar of the post-Ptolemaic papyri was the new *desideratum* in papyrology. The work had been embarked on by Salonius, Ljungvik, Kapsomenos, and Palmer, only to be interrupted or thwarted by circumstance or lack of resources. Salonius [1927], for instance, only managed to write an introduction on the sources, though he offered valuable comments on the matter of deciding how close to spoken language a piece of writing is. Ljungvik [1932] contains select studies on some points of syntax.

It is in the 1930's that we see attempts to create a grammar of the papyri that would be the equivalent of Mayser for the post-Ptolemaic period. Kapsomenos published a series of critical notes [1938, 1957] on the subject; though he attempted at a work on the scale of the *Grammatik*, he found the resources sorely lacking, as the existing editions of papyrus texts could not form the basis for a systematic grammatical study. The other was Palmer, who had embarked on similar project and had already set out a methodology [1934]; the war interrupted his efforts, and he published what he had already completed, a treatise on the suffixes in word formation [1945].

A new work of some magnitude presents itself two decades later with B. G. Mandilaras' *The verb in the Greek non-literary papyri* [1973]. Though it does not aim to be a grammar of the papyri, it does offer a thorough and satisfactory treatment of the verbal system as manifest in the papyri. Further efforts essentially do not appear until the publication of Gignac's grammar. It is essentially treading in the footsteps of Mayser, only with further methodological refinement and a more limited, though still sufficiently exhaustive, array of examples. The author, for reasons unknown to me, only managed to complete two of the three projected volumes, on phonology and on morphology. The volume on syntax is thus absent, a gap only partly filled by Mandilaras' *The verb in the Greek non-literary papyri*.

Finally, there is the aforementioned *The Language of the Papyri* [Evans and Obbink, 2010], which does not aim to be a work on the same scale as the aforementioned. It is a collection of articles on various topics, the whole of which is meant to illuminate new avenues for future research. A particularly relevant chapter for this thesis is the last one by Porter and O'Donnell [Porter and O'Donnell, 2010], who set out to create a linguistic corpus for a selection of papyri; their tagging approach, however, is manual, and their target corpus limited. The authors also are the creators of `http://www.opentext.org/`, a project aiming for the development of annotated Greek corpora and tools to analyse them; sadly, no progress seems to have been made since 2005.

2.1.2 Corpus linguistics

A[1] corpus or text corpus is a large, structured collection of texts designed for the statistical testing of linguistic hypotheses. The core methodological concepts of this mode of analysis may be found in the concordance, a tool first created by biblical scholars in the Middle Ages as an aid in exegesis. Among literary scholars, the concordance also enjoyed use, although to a lesser degree; the eighteenth century saw the creation of a concordance to Shakespeare.

The development of the concordance into the modern corpus was not primarily driven by the methods of biblical and literary scholars; rather, lexicography and pre-Chomskyan structural linguistics played a crucial role.

1 The following section is based *passim* on McCarthy and O'Keeffe [2010].

Samuel Johnson created his famous comprehensive dictionary of English by means of a manually composed corpus consisting of countless slips of paper detailing contemporary usage. A similar method was used in the 1880s for the Oxford English Dictionary project - a staggering three million slips formed the basis from which the dictionary was compiled.

1950s American structuralist linguistics was the other prong of progress; its heralding of linguistic data as a central given in the study of language supported by the ancient method of searching and indexing ensures its proponents may be called the forerunners of corpus linguistics.

Computer-generated concordances make their appearance in the late 1950s, initially relying on the clunky tools of the day - punch cards. A notable example is the Index Thomisticus, a concordance to the works of Thomas of Aquino created by the late Roberto Busa S.J. which only saw completion after thirty years of hard work; the printed version spans 56 volumes and is a testament to the diligence and industry of its author. The 1970s brought strides forward in technology, with the creation of computerised systems to replace catalogue indexing cards, a change that greatly benefited bibliography and archivistics.

It is only in the 1980s and 1990s that are marked the arrival of fully developed corpora in the modern sense of the word; for though the basic concepts of corpus linguistics were already widely used, they could not be applied on a large scale without the adequate tools. The rise of the desktop computer and the Internet as well as the seemingly ever-rising pace of technological development ensured the accessibility of digital tools. The old tools - punch cards, mainframes, tape recorders and the like - were gladly cast aside in favour of the new data carriers.

The perpetual increase of computing power equally demonstrated the limits of large-scale corpora; while lexicographical projects that had as their purpose to document the greatest number of possible usages could keep increasing the size of their corpora, the size of others went down as they whittled the data down to a specific set of uses of language.

The possible applications of the techniques of corpus linguistics are diverse and numerous; for they allow for a radical enlargement in scope while remaining empirical, and remove arduous manual labour from the equation. Corpus linguistics can be an end to itself; it can, however, assert an important role in broader research. McCarthy and O'Keeffe [2010, p. 7] mention areas such language teaching and learning, discourse analysis, literary stylistics, forensic linguistics, pragmatics, speech technology, sociolinguistics and health communication, among others.

The term 'corpus' has a slightly different usage in classical philology: they designate a structured collection of texts, but that collection is not primarily intended for the testing of linguistic hypotheses. Instead, we have, for instance, the ancient corpus Tibullianum, or modern-day collection, for instance the Corpus Papyrorum Judaicarum, etc. We are

primarily interested in the digital techniques used to create linguistic corpora; so let us first take a look at the progress of the digital classics.

2.1.3 The digital classics

Classical philology, despite its status as one of the oldest and most conservative scientific disciplines still in existence today, has in the last fifty years found itself at the front lines of the digital humanities movement. Incipient efforts in the fifties and sixties, mainly stylometric and lexical studies and the development of concordances, demonstrated the relevance of informatics in the classics, an evolution that was at first met with some skepticism, but later fully embraced.

The efforts began with the aforementioned Index Thomisticus, the first computer-based corpus in a classical language; but the first true impetus was the foundation of the Thesaurus Linguae Graecae project in 1972, a monumental project with as its goal the stocking of all Greek texts from the Homeric epics to the fall of Constantinople. Over the years, many functions have been added to this ever more powerful tool; and even in the beginning stages of its development, the TLG garnered praise.

The usefulness of the tool in its current form cannot be overstated: not only does it contain a well-formatted and easily accessible gigantic collection of text editions whose scope and dimensions exceed those of nearly any university library; it also offers all of these texts in a format that allows for lexical, morphological and proximity searches, as well as including a full version of the Liddell & Scott and Lewis & Short dictionaries. The TLG has become a staple of the digital classics.

Despite this, the TLG is becoming more and more dated as technology progresses. While recent years have seen the rise of Unicode as the standard for encoding ancient Greek, the TLG still uses beta code, a transliteration system designed to only use the ASCII character set, and the texts are stored using an obsolete text-streaming format from 1974, which divides the text in blocks of eight kilobytes and marks the division between segments.

A digitised version of the Liddell-Scott-Jones lexicon has been added to the TLG's web interface, but the texts themselves have not undergone extensive tagging, only lemmatisation. Searching through the database can be done by searching for specific forms of a lemma, or by searching for all forms of a lemma, but this is essentially the limit of the search tool's power; it is not possible to perform a query for all possible lemmata associated with a particular form, i.e. we cannot find all forms which are, for example, an active perfect indicative.

In the wake of the TLG, several notable projects have emerged: Brepols' Library of Latin Texts is trying hard to be for Latin texts what the TLG is for Greek texts; the Packard Humanities Institute has released CD's containing a selection of classical Latin works. In more recent times, the Perseus Project has enjoyed great popularity because of the attractive combination of an excellent selection of classical texts with translations, good accessibility and a set of interesting textual tools, the entire

package carrying a very interesting price tag for the average user — it is free to use, and for the greatest part, open source as well.

The databases I have mentioned are quite general in scope; but within the domain of classical philology, other specialised projects exist. Within the field of papyrology, for instance, the digital revolution has taken a firm foothold. Starting with several separate databases, the field has experienced a tendency towards convergence and integration of the available resources, as exemplarised by the papyri.info website, maintained by Columbia University, that integrates the main papyrological databases into a single database.

A great feature of this database is the shell in which all data is wrapped; they are compliant with the EpiDoc standard, a subset of XML based on the TEI standard and developed specifically for epigraphical and papyrological texts. One may access the database's resources through the Papyrological Navigator and suggest corrections and readings through the Papyrological Editor. What's more, all data is freely accessible under the Creative Commons License, crowd-sourced, regularly updated, and can be downloaded for easier searching and tweaking.

In other words, papyri.info has brought the open-source mentality from the computer world into the classics. For our purposes, this open setup is desirable, as the database is not fit for them as it is, but can with some effort be molded into a useful tool.

2.1.4 Natural language processing

Natural language processing (henceforth NLP) is a subdiscipline in computer science concerned with the interaction between natural human language and computers. Its history well and truly starts in the fifties, with a basic concept which has played a great role in natural language processing, and computer science in general, the Turing test. This test, put forth by Alan Turing in his seminal paper *Computing Machinery and Intelligence* [Turing, 1950], evaluates whether a machine is intelligent or not by placing a human in conversation with another human and a machine; if the first human cannot tell the other human and the machine apart, the machine passes the test.

Machine translation systems entered development, though progress soon stalled because of technical limitations and because of methodological obstacles: such systems were dependent on complex rulesets written by programmers that allowed for very little flexibility. Because of the slow return on investments made, funding for artificial intelligence in general and machine translation specifically was drastically reduced throughout the late sixties and the seventies.

A resurgence followed: thanks to advances in computational power and the decline of Chomskyan linguistics in natural language processing, which had been the dominant theoretical vantage point in the preceding thirty years, the eighties were marked by the introduction of statistical machine translation, which is fundamentally based on the tenets of corpus linguistics. Modern natural language processing is

therefore situated on the crossroads between various fields: artificial intelligence, computer science, statistics, and corpus and computational linguistics. It looks to be an exciting field for the coming years as its techniques are under constant improvement and ever more present in our daily lives.

Most NLP software is designed explicitly with living languages in mind; English, being a world language and the international *lingua franca*, has enjoyed most of the attention, but other major languages have enjoyed some attention, too. Ancient languages, however, are neglected, presumably due to their often high complexity and the extensive study and analysis to which they have been submitted by skilled scholars. Yet most texts have not been integrated in annotated corpora; and though databases such as the Perseus project contain large swathes of morphologically and sometimes syntactically annotated text, the process has been driven largely by manual labour; to give an exhaustive list is not appropriate here, but another such example which is relevant is the PROIEL project [*PROIEL: Pragmatic Resources in Old Indo-European Languages*], which is also a treebank, i.e. a database of syntactically annotated sentences. It contains data for Herodotus and the New Testament.

2.2 CONCEPTS AND TECHNIQUES

2.2.1 Mathematics

While there is, of course, no room in this thesis for an extended course in mathematics, it is necessary to have some background in order to understand the techniques used for the design and implementation of the architecture. What follows is a very brief overview of important concepts in set theory, probability, calculus, linear algebra, statistics and formal language theory.

Set theory

Consider an object o and a set A. We write 'object o is an element of set A' as $o \in A$. Sets themselves are also objects and can belong to other sets. Consider two sets, A and B. We write that A is a subset B as $A \subseteq B$; this implies that B contains all elements in A and does not exclude the possibility of $A = B$; if A is a proper subset of B, i.e. all elements of A are in B but not all elements of B are in A, we write $A \subset B$. Mirroring these symbols from right to left gives us the symbols for supersets and strict supersets, respectively. The empty set, which contains no elements, is written as $\varnothing$.

There exist different binary operators on sets (i.e. operators on two sets) which return another set. The most common operators are:

- the union of sets, written as $A \cap B$, denotes a set which contains all elements which are in A and B;

- the intersection of sets, written as $A \cup B$, denotes a set which contains all elements which are in both A and B;
- the difference of sets, written as $A \setminus B$, denotes a set which contains every element from A excluding those which are also in B.
- the Cartesian product of sets, written as $A \times B$, is the set containing all ordered pairs of elements from A and B.

Probability

A probability is a measure for the likelihood of an event for an experiment, which we intuitively understand to be an action whose outcome we want to observe. Such events are then elements of a set containing all possible outcomes of an experiment; an event can be a point or subset of that set. We call this set the sample space, denoted S. We denote the probability of an event E as P(E).

Axiomatically, we can define probability as follows:

1. For every event E, $P(E) \geqslant 0$; no event can have a negative probability.
2. $P(S) = 1$; that is to say, every experiment has an event.
3. For any sequence of disjoint events A_i (that is to say, there is no overlap between events), the probability of any one of these events occurring is the sum of their respective probabilities.

A few other important properties of probabilities and events are the following:

1. The complement of an event A, which is the union of all elements of S which are not an element of A, is denoted A^C; $P(A^C$ is ther probability of this event occurring and is equal to $1 - P(A)$.
2. For any event E, $0 \leqslant P(E) \leqslant 1$.
3. Given any two events A and B, $A \subset B \rightarrow P(A) \leqslant P(B)$.

Given probabilities of a number of events, we can establish relationships between these probabilities and compute related probabilities using the rules certain rules. A classic rule is the **multiplication rule**, which states that if we perform an experiment in k parts and the i^{th} part of the experiment has n_i possible outcomes, and the outcomes of prior parts of the experiment do not affect latter ones, the probability of any specific sequence of partial outcome will be the product of all outcome counts n_i with i ranging from 1 to k.

Set theory is important when we want to know the probability of an event E which can be constructed from a set of sets A_i using set operators. The third axiom of probability has already given us the solution for disjoint events; events may also overlap, and in this case,

we need a more sophisticated formula. For the union of any n events A_i, the following holds:

$$h_w(x) = w_1 x + w_0 \tag{1}$$

- Axioms
- Probabilities and sets
- Bayes' rule

Calculus

- Derivatives
- Numerical methods

Linear algebra

- Vector spaces
- Jacobian matrices

Statistics

- **Regression**

  Regression is a classic technique from statistics, often visualised as 'fitting a line to a set of points'. Classification is a related technique which uses regression to classify new data points. This is essentially what any probabilistic model for natural language processing does, in one form or another. What follows is an overview of various types of regression and corresponding methods for classification.

  We start with **univariate linear regression**. Given a set of n points in the plane, we want to find a hypothesis that best corresponds to the location of these points, and we want this hypothesis to be a linear function. This function is then of the form:

  $$h_w(x) = w_1 x + w_0 \tag{2}$$

  Unless all points are collinear, it is of course impossible to find a function of this form that gives a correct mapping for each point. The best we can do is find the values of w_o and w_1 for which the empirical loss on the mappings is minimal. The traditional way of doing this is to define a function that computes the squares of the errors and sums it over all data points; this is called an L_2 **loss function**. We now want to find the values of w_0 and w_1 for which this function attains a minimum. We can find these minimal points by solving for the roots of the partial derivative functions of this loss function with respect to w_0 and w_1. This

problem is mathematically relatively simple and has a unique solution. This solution is valid for all loss functions of this type.

Problems arise when we are trying to create a nonlinear model. In this case, the minimum loss equations frequently do not have a unique solution. We can of course still model the problem algebraically, and the goal is the same: finding the roots of the partial derivative function. Now, however, we need to use a more sophisticated method: **gradient descent**. We can visualise this technique as 'descending a hill'; the 'hill' is the graphical representation of the root of the system of partial derivatives, and by 'descending' this hill, i.e. by iteratively picking values which bring us closer to the bottom part of the valley next to the hill, which corresponds to the minimal point of the function, eventually convergence will be reached on the minimum and we will found the correct weights for our function. The difference by which we change the value at each iteration is called the **step** or **learning rate** and determines how fast we will converge; it may be either a fixed constant or a mutable value which can increase or decay according to the current state of our descent.

Multivariate linear regression poses a similar problem; only this time the function is not dependent on a single variable, but on two or more. Such a function is a bit more complex, but we can find a solution to the regression problem using analogous techniques. Suppose the function has n variables. Each example x_j must be a vector with n values. At this point, we are looking at a function of the following form:

$$h_w(x_j) = w_0 + w_1 x_{j,1} + w_2 x_{j,2} + ... + w_n x_{j,n} = w_0 + \sum_i w_i x_{j,i} \tag{3}$$

We want to simplify this to make algebraic manipulations easier. We therefore prepend an extra component $x_{j,0} = 1$ to the vector x_j; now using vector notation we can simplify the previous equation to:

$$h_w(x_j) = \sum_i w_i x_{j,i} = w \cdot x_j \tag{4}$$

What we are now looking for is a vector w containg the weights of our function which minimises the empirical loss, as in univariate linear regression. We can equivalently use gradient descent; only now, of course, the computational cost of that technique will be higher. A common problem can now appear: **overfitting**, that is, giving an irrelevant dimension of the vector w too much weight due to chance errors in the computation. This can be compensated by taking into account the complexity of the hypothesis; a statistical equivalent to Ockham's razor, if you will.

- **Classification**

We can define an analogous process for classification; only now the function must not fit to the data itself but must create a **decision boundary** between data points. If there exists a linear function which satisfies this property for a given data set , we call the bounding line or surface generated by this function a **linear separator**, and the data set **linearly separable**. The hypothesis function is now of the form:

$$h_w(x) = 1 \text{ if } w \cdot x \geqslant 0, 0 \text{ otherwise.} \tag{5}$$

We can view this as a function $\texttt{threshold}(w \cdot x)$ which is equal to 1 only if $w \cdot x = 0$ Note that while the separating function is linear, the hypothesis function is not, and in fact has the distinctly unappealing property of not being differentiable. We can therefore not apply the technique of gradient descent here. Furthermore, this type of function has exactly two outputs: 1 or 0. For our purposes, we need subtler methods of classification. This type of hypothesis function is therefore not fit for our purposes, but it does give a good idea of what classification is.

The best option is replacing the hard threshold function with the sigmoid or logistic function, which offers a good approximation and is differentiable at every point. This function is of the form:

$$g(x) = \frac{1}{1 + e^{-x}} \tag{6}$$

Such that our new hypothesis function is:

$$h_w(x) = g(w \cdot x) = \frac{1}{1 + e^{-w \cdot x}} \tag{7}$$

If we use this function, we are performing **linear classification with logistic regression**.

LOGISTIC REGRESSION AND THE CHAIN RULE

- Clustering

Formal language theory

- Languages and strings
- Regular languages
- Context-free grammar and languages
- The Chomsky hierarchy

2.2.2 Natural language processing

N-grams

Hidden Markov Models and Maximum Entropy Models

2.2.3 Artificial intelligence and machine learning

What is machine learning?

- Supervised learning
- Unsupervised learning

Neural networks

An artificial neural network is massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use.

For the design of an architecture which allows us to solve the problems above, we have taken our cues largely from recent work in machine learning as applied to natural language processing. In particular, we follow the approach set out in Weston & Collobert 2008 and expanded in Weston et al. 2011, that is, the use of deep neural networks for joint training on our chosen corpus. This section is dedicated to a more expository overview of that architecture for the mathematical layman.

An artificial neural network is a massively parallel processing system constructed from simple interconnected processing units (called neurons) which has the ability to learn by experience and store this knowledge for later use. The term 'neural network' is due to the resemblance of this architecture to the most powerful biological processor known to exist, the human brain, which has a way of functioning which is broadly analogous to this process.

Artificial neural networks find their origin in a mathematical model dating from before the first wave of artificial intelligence in 1956, the McCulloch-Pitts Threshold Logical Unit, known also as the McCulloch-Pitts neuron. Warren McCulloch was a psychiatrist and neuroanatomist; Walter Pitts a mathematical prodigy. Both met at the University of Chicago, where a neural modeling community led by the mathematical physicist N. Rashevsky had been active in the years preceding the publication in 1943 of the seminal paper A logical calculus of the ideas immanent in nervous activity.

Formally, we can define a neuron as a triplet (v,g,w) where:

- v is an input function which takes a number of inputs and computes their sum;
- g is an activation function which is applied to the output of the input function and determines if the neuron 'fires';
- w is an output function which receives its input from the activation function and distributes it over a number of outputs.

Given its structure, we can also see a neuron as a composite function $F = w \circ g \circ v$. The combination of several of these units using directed links forms a neural network. A link connecting a node i to a node j transfers the output a_i of node i to node j scaled by a numeric weight $w_{i,j}$ associated with that specific link. This is the general model of a neural network; countless variations on this theme have been developed for different purposes, mainly by modifying the activation function and the interconnection of neurons.

The activation function g typically will be a hard threshold function (an example of this is the original McCulloch-Pitts neuron), which makes the neuron a perceptron, or a logistic (also known as a sigmoid) function, in which case we term the neuron a sigmoid perceptron. Both these functions are nonlinear; since each neuron itself represents a composition of functions, the neuron itself is a non-linear function; and since the entire network can also be seen as a composite function (since it takes an input and gives an output) the network can be viewed as a nonlinear function. Additionally, choosing to use a logistic function as an activation function offers mathematical possibilities, since it is differentiable. This offers similar possibilities as the use of the logistic function for regression (cf. *supra*).

The links between nodes can be configured in different ways, which each afford distinct advantages and disadvantages. Broadly, we can distinguish two models. The simplest model is the **feed-forward network**, which can be represented as an acyclic directed graph. The propagation of an input through this kind of network can be seen as a stream, with posterior (downstream) nodes accepting outputs from prior (upstream) nodes. This type of network is the most widely-used and is used in the architecture. A more complex type is the **recurrent network**, which feeds its output back to its input and thus contains a directed cycle; this type of network has interesting applications (for example in handwriting recognition), as they resemble the neural architecture of the brain more closely than feedforward networks do.

Feed-forward networks are often organised (to continue the stream analogy) in a kind of waterfall structure using layers. The input is the initial stream, the output is the final stream; in between, we may place hidden layers, which are composed of neurons which take inputs and outputs as any neuron does, but whose output is then immediately transferred to a different neuron. Throughout the network, we can equip the neurons in each layer with distinct activation functions and link weights and in this way mold the learning process of the network to our purpose.

Single-layer networks contain no hidden layers; the input is directly connected to the output. Therefore, the output is a linear combination of linear functions. This is undesirable in many cases. The main problem, demonstrated early on in the development of neural network theory, is the fact that such a network is unable to learn functions that are not linearly separable; one such function is the XOR function, which is a very simple logical operator. Despite this, such neural networks

are useful for many tasks, as they offer an efficient way of performing logistic regression and linear classification.

Our interest lies in multi-layer networks, however. Multi-layer networks contain one or more layer between the input and output layer, which are called hidden layers. By cascading the input through all these layers, we are in fact modeling a nonlinear function which consists of nested nonlinear soft threshold functions as used in logistic regression. The network can now be used to perform **nonlinear regression**. Different algorithms exist which can be used to train the network; the most important one is the **backpropagation algorithm**, which is the equivalent of the loss reduction techniques used in linear regression.

Suppose that a neural network models a vector-valued hypothesis function h_w which we want to fit to an example output vector y. We can create a L_2 loss function E by taking the error on this vector and squaring it. This function can be quite complex, but by taking partial derivatives of this function, we can consider the empirical loss on each output separately, like so:

$$\begin{aligned}\frac{\partial}{\partial w}E(w) &= \frac{\partial}{\partial w}|y - h_w(x)|^2 \\ &= \frac{\partial}{\partial w}\sum(y_k - a_k(x))^2 \\ &= \sum\frac{\partial}{\partial w}(y_k - a_k(x))^2\end{aligned} \tag{8}$$

If the output function is to have m outputs, instead of handling one large problem, we can subdivide the problem into m smaller problems. This approach works if the network has no hidden layers, but due to the fact that nothing is really known about the hidden layers if we only look at the output layer, a new approach is necessary. This is called backpropagation, a shorthand term for backward error propagation.

Deep learning

2.3 RELATED WORK

- Mimno and Wallach
- Dik and Whaling
- Lee
- the Open Philology Project
- Bamman
- Crane

3 ANALYSIS

3.1 OBJECTIVES

The tasks we aim to perform are the following:

- **tokenization**, that is, the splitting of individual words into separate 'tokens' which can be considered to form a single linguistic unit;
- **stemming**, the splitting up of words into their morphemic constituents for purposes of lemmatisation or reduction of the complexity of assigning part-of-speech tags;
- **part-of-speech tagging** involves labeling each individual word with relevant information indicating its syntactic role;
- **chunk parsing** or **shallow parsing** (as opposed to deep parsing) aims to label shallow syntactic constituents, such as individual phrases and clauses.

The performance of any NLP system on these tasks is determined by the quality of its underlying **language model**, which assigns probabilities to word sequences; thus, this is our primary task, and will provide the necessary data for the four tasks denoted above.

3.2 SCOPE OF THE PROBLEM

We summarise the major component problems of our task facing us.

A first major obstacle is the size and diversity of the corpus. The corpus of the papyri as presented on papyri.info contains more than 4.5 million words and spans nearly a millennium, by virtue of which it inevitably contains tens of thousands of unorthodox or corrupted word forms. This adds a great amount of complexity: not only must our model recognise 'normal' word forms, it must also be able to make inferences (albeit limited ones) about unknown forms with minimal hiccups.

- quantity: roughly 4.5 million words
- quality: varying from extremely corrupted text to nearly pristine documents
- representation: from 300BC to 800AD, representing an array of different discourse registers, but not including literary texts

- simplicity: offered in EpiDoc XML, which is more complex than plain text, but is relatively easily converted

- retrievability: extensive annotation using XML makes searching and retrieving text easy

The second major obstacle is the ancient Greek language itself. Though it has lost a great deal of morphological complexity in its evolution towards its current state, the Greek of Hellenistic, Roman and Byzantine times is still marginally more complex than a language like English, which is the target language for most research in natural language processing. Commonly used techniques in NLP are still applicable and have been used with success on other morphologically complex languages, but given the size of the tag set for ancient Greek morphology and syntax, it is wise to preprocess the corpus to reduce the amount of factors that must be held into account in the creation of our model.

Another important obstacle to the development of natural language processing tools for ancient Greek is the relative scarcity of training material: most resources do provide morphological analysis, but there are very few projects concerned with treebanks or databases of semantically annotated Greek. The Perseus project, for instance, has developed a dependency treebank for Latin and ancient Greek. It is an admirable effort, but limited in scope and containing mainly poetry, which is in itself valid training data, but certainly not sufficient training data if we want our system to be able to analyse large amounts of prose. What's more, the project seems to be lacking manpower and has lost steam since its inception, the last update dates from 2012, more than a year ago at the time of this writing.

Another interesting treebank is that hosted by the PROIEL [1] project, which aims to offer morphologically and syntactically annotated multilingual corpora for comparative purposes. The project, contrary to the Perseus treebank, seems to be alive and well at the time of this writing. This corpus contains data which can be of much help: large swathes of Herodotus, the New Testament, and the writings of the Byzantine historian George Sphrantzes are fully annotated, both morphologically and syntactically. Since the Greek of the papyri is syntactically similar to the Greek of these texts, we are afforded a good basis for our system.

Nevertheless, probabilistic natural language processing is by virtue of its underlying principles hungry for ever more data in order to achieve high performance. Therefore, we somehow need to create a larger foundation upon which we can construct a performant architecture. Here, we can take our cues from the field of machine learning, where the state-of-the-art approaches rely on massive amounts of unlabeled data which are then submitted to analysis. The exact approaches chosen are explained in detail in the next chapter.

1 Short for 'Pragmatic Resources for Indo-European Languages'

3.3 TOKENIZATION

Tokenization is also a tricky affair given the common usage in Greek of crasis and similar phenomena. This issue must also be resolved using a system which is as simple as possible. Resolving this ambiguity requires, in many instances, the use of external resources. Take the forms ἄνθρωπος and ἄνθρωπος , for instance. While this kind of crasis is not omnipresent in most Greek texts, it is certainly not inexistent. Solving this kind of overlap requires a system which is able to recognise accents and spirituses and is able to infer the elision of the definite pronoun ὁ as well, a task which is not easy to generalise. The founder of the Perseus project, Gregory Crane, wrote Morpheus in the 80s, which is a morphological analysis tool for ancient Greek. He has not made the source code for this tool publicly available (and will not do so in the foreseeable future) but an SQL dump is freely available from the Perseus project's website which contains an astounding one million word forms (REFERENCE), many of which are one-off cases exactly like this one. We will make gratuitous use of this tool for tokenising Greek text. A possible objection is that ambiguity can exist between several such forms; but for tokenisation, this is hardly relevant.

Another delicate point is punctuation: do we consider the Greek semicolon as delimiting a sentence or not? It seems to me that the best course of action here is to simply consider these as having a value a bit between the modern colon and semicolon as used in English and other modern languages. The reason for this is that semantic relations very frequently span over this symbol; therefore we would run the risk of achieving very low accuracy rates in this domain. On the other hand, this is also true for full sentences delimited by a period, but less frequently so. It would also not be computationally reasonable to check the previous sentence each time; this would cause massive overhead and effectively at least double the input size for our algorithm.

Once the text is tokenised, we may proceed to the next level, that is, morphological analysis.

4 DESIGN

4.1 SELECTING TRAINING DATA

4.2 CREATING A LANGUAGE MODEL

4.2.1 Network structure

4.2.2 Unsupervised learning

Genetic model generation

Increasing dictionary sizes

4.2.3 Supervised learning

4.2.4 Possible improvements

- affix features
- gazetteers
- cascading features
- ensemble voting
- parsing
- word representations
- general engineering

4.3 ANNOTATING THE CORPUS

4.3.1 Using the embeddings

4.3.2 Viterbi decoding

5 IMPLEMENTATION AND EXECUTION

5.1 LANGUAGE AND SOURCE CODE

5.1.1 Choice of language

- Haskell
- performant
- concise syntax
- interfaces with C for streamlining dense matrix-vector computations
- excellent Unicode support

5.1.2 Source code availability

- GitHub
- based off SENNA but with better Unicode support

5.2 THE FULL PROCESS

5.2.1 Preparing the training corpora

5.2.2 Training the model

5.2.3 Preparing the object corpus

6 INTERPRETATION

6.1 DATA ANALYSIS

7 ASSESSMENT

7.1 ACCURACY

8 FURTHER WORK

This chapter is meant as a brief evocation of the potential of a fully annotated corpus of the papyri for further research.

8.1 COLLABORATIVE EDITING

The IDP project is heading into crowdsourcing territory at full steam, and excluding our own work from this movement would be inserting a shrill note into this symphony. All data is placed on GitHub at https://github.com/sinopeus/tjufy, freely accessible and editable for all.

This opens promising avenues of inquiry that do not have a direct relation to this thesis. For instance, it can in the future be integrated into SoSOL and absorbed into the larger codebase for the IDP project if it does not create too much overhead for the current developers (the technical back end of the IDP seems to be labyrinthine and adding new layers of complexity might be off-putting). It could even grow into a separate project which itself could be linked to papyri.info as the HGV, APIS and Trismegistos currently are, by a common system of indexation.

Making the code publically available to all also has the advantage of the public eye inspecting the texts; using solely automatic analysis is bound to deliver an inaccurate result, however small that inaccuracy may be, as creating a NLP engine perfectly capable of understanding language would be the equivalent of creating a perfect artificial intelligence. Therefore, considering the size of the corpus, one must rely upon the intelligence of the community. In the same way open source software is often among the best of software due to public inspection, the potential for a crowdsourced corpus is immense.

8.2 CORPUS-BASED GRAMMARS AND LEXICA

Expanding our method to other texts might bring the benefit of comprehensive corpus-based grammars and lexica, which can integrate available data on the fly and create a self-updating and reliable web of grammatical knowledge. Instead of focusing mainly upon a few choice authors or laboriously trudging through the huge wealth of ancient Greek literature to linearly create lexica and grammars, all of it could be harnessed at once in a quantitatively precise and easily visualisable way.

http://www.digitalhumanities.org/dhq/vol/003/1/000033/000033.html

8.3 HISTORICAL AND VARIATIONAL LINGUISTICS

The language of the papyri has an important role to play in the historical linguistics of Greek; once a full annotation has been achieved, it could be possible to implement the same methods used for synchronic language processing to map language changes in a statistical way; it could be possible to estimate the transition probabilities for diachronic grammatical evolutions, which has the potential to create a picture of the evolution of Greek that would be both comprehensive and precise. It even has potential on a comparative level; given the long history and meandering evolutionary trajectory of the Greek language, one could observe from the data catalysts for language evolution in one direction or the other and apply that comparatively.

One might also win valuable insight into language diversity in Egypt; using the paraliterary data already available from the Trismegistos, linguistic phenomena and evolutions could be visualised on a map and give insight into the diatopic, diastratic and diaphasic variation of Egyptian koinê, much in the way of modern dialect survey maps but directly linked to the original texts.

8.4 TEXTUAL CRITICISM

Textual criticism, too, could benefit from improved access to linguistic data; dubious *passus* could be disambiguated by comparing them to similar instances in papyri from the same period and adapting constructions and words from them. This technique is harnessed by Mimno and Wallach [2009], who use the techniques of statistical NLP solely for these specific critical problems. Though textual criticism will for the foreseeable future still necessitate trained papyrologists, the need for a very in-depth knowledge of the corpus of papyri can be greatly reduced by calling upon data from other parts of the corpus to present a series of statistically possible solutions for textual issues.

8.5 NAMED ENTITY RECOGNITION

Named entity recognition is a subdiscipline in natural language processing which is concerned with the automatic extraction and localisation of all kinds of names from texts. It has been used extensively in literary texts with a view to discern the importance of certain characters throughout the text. The KU Leuven's long-standing Prosopographia Ptolemaica project, which aims to be a repository of all inhabitants of

Egypt between 300 and 30 B.C., could easily benefit from these techniques. The abundant manual labour that has gone into the project could be fed as training data to and then supplemented by a named entity recognition engine that could also categorise personal names by any criteria and establish contextual relations between them. To take a very rudimentary example, the name 'Alexander' could be retrieved in all texts and a cluster of related names generated, so that related individuals may be placed in a web of relations; or one could ask, by combining the already present linguistic annotation, to display all adjectives which accompany the name 'Alexander'.

It could even go further than this and also include other particular names, such as places, distances, monetary units, weights, and so on. Historians could create a comprehensive overview of, for instance, the inflation of Egyptian currency, or map out trade connections using a search for all mentions of currency, weight and places which are in proximity to each other.

9 CONCLUSION

BIBLIOGRAPHY

Evans, Trevor V. and Dirk D. Obbink

2010 *The Language of the Papyri* (eds.), Oxford University Press. (Cited on pp. 1, 4.)

Gignac, Francis Thomas

1976 *A Grammar of the Greek Papyri of the Roman and Byzantine Periods. I. Phonology.* Milano: Goliardica. (Cited on p. 3.)

1981 *A Grammar of the Greek Papyri of the Roman and Byzantine Periods. II. Morphology.* Milano: Goliardica. (Cited on p. 3.)

Haug, Dag Trygve Truslew *et al.*

PROIEL: Pragmatic Resources in Old Indo-European Languages, http://foni.uio.no:3000/, info at http://www.hf.uio.no/ifikk/proiel/. (Cited on p. 8.)

Kapsomenos, Stylianos G.

1938 *Voruntersuchungen zu einer Grammatik der Papyri der nachchristlichen Zeit*, Münchener Beiträge zur Papyrusforschung und antiken Rechtsgeschichte, München: C. H. Beck. (Cited on p. 4.)

1957 "Ἔρευναι εἰς τὴν γλῶσσαν τῶν ἑλληνικῶν παπύρων. Σείρα Πρώτη", *EEThess*, vii, pp. 225–372. (Cited on p. 4.)

Ljungvik, Herman

1932 *Beiträge zur Syntax der spätgriechischen Volkssprache.* Vol. 27, Skrifter utgivna av K. Humanistiska Vetenskaps-Samfundet i Uppsala, 3, Uppsala & Leipzig: Humanistiska Vetenskaps-Samfundet. (Cited on p. 3.)

Mandilaras, Basileios G.

1973 *The verb in the Greek non-literary papyri*, Athens: Hellenic Ministry of Culture and Sciences. (Cited on pp. 3, 4.)

McCarthy, Michael and Anne O'Keeffe

2010 "What are corpora and how have they evolved?", in *The Routledge Handbook of Corpus Linguistics*, ed. by Anne O'Keeffe and Michael McCarthy, London and New York: Routledge. (Cited on pp. 4, 5.)

Mimno, David and Hanna Wallach

2009 "Computational Papyrology (presentation)", in *Media in Transition 6: Stone and Papyrus, Storage and Transmission*, Cambridge, MA. (Cited on p. 30.)

Palmer, Leonard Robert

1934 "Prolegomena to a Grammar of the post-Ptolemaic Papyri", *J. Th. S.*, xxxv, pp. 170–5. (Cited on p. 4.)

1945 *A Grammar of the post-Ptolemaic Papyri: Accidence and Word-Formation*, London: Oxford University Press. (Cited on p. 4.)

Porter, S. E. and M. B. O'Donnell

2010 "Building and Examining Linguistic Phenomena in a Corpus of Representative Papyri", in *The Language of the Papyri*, Oxford University Press, pp. 287–311. (Cited on p. 4.)

Salonius, A. H.

1927 *Zur Sprache der griechischen Papyrusbriefe*, vol. i, Die Quellen, Akademische Buchhandlung. (Cited on p. 3.)

Turing, Alan Mathison

1950 "Computing Machinery and Intelligence", *Mind*, 59, 236 [Oct. 1950], pp. 433–460. (Cited on p. 7.)