ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ευφυής Σήμανση και Λημματισμός Αρχαίων Ελληνικών Κειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Λιόσσης Εμμανουήλ

Επιβλέπων: Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής

Αθήνα, Αύγουστος 2014

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ευφυής Σήμανση και Λημματισμός Αρχαίων Ελληνικών Κειμένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Λιόσσης Εμμανουήλ

Επιβλέπων: Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26^{ην} Αυγούστου 2014

.....	.....	.....
Α. Γ. Σταφυλοπάτης	Σ. Κόλλιας	Γ. Στάμου
Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Επ. Καθηγητής ΕΜΠ

Αθήνα, Αύγουστος 2014

Λιόσσης Μ. Εμμαουήλ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Εμμανουήλ Λιόσσης 2014

Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται μία γενική μέθοδο σημάνσεως λέξεων κειμένων και ευρέσεως των λημμάτων τους, με εφαρμογή τα αρχαία ελληνικά. Συγκεκριμένα, πρώτον, εκτιμάται το μέρος τού λόγου και η μορφολογία κάθε λέξεως στο κείμενο, δηλαδή αναγνωρίζεται η κλίση της. Δεύτερον, για κάθε μορφή λέξεως στο κείμενο εκτιμάται το λήμμα της, δηλαδή η κανονική της μορφή όπως την βρίσκουμε σ' ένα λεξικό. Η προσέγγιση δεν στηρίζεται σε αυθαιρέτους κανόνες αλλά σε ευφυείς μεθόδους μηχανικής μαθήσεως. Το σύστημα δεν είναι προσδεδεμένο στα αρχαία ελληνικά αλλά είναι σχεδιασμένο ώστε να μπορεί να εφαρμοσθεί σε όλες τις γλώσσες, ιδιαιτέρως σε αυτές που εμφανίζουν πλουσία μορφολογία όπου υπάρχει δυσκολία επεξεργασίας. Έχουν επιστρατευθεί οι κατάλληλες μεθοδολογίες τεχνολογίας λογισμικού ώστε να αντιμετωπίζεται η κάθε γλώσσα σε αφαιρετικό επίπεδο καθ' ενιαίο τρόπο. Το σύστημα αυτό είναι το πρώτο στρώμα στο οποίο μπορούν να βασισθούν υπηρεσίες όπως αναζήτηση υψηλής ποιότητος, μηχανική μετάφραση, συστήματα γνώσεων οντοτήτων και σημαντική αναζήτηση.

Λέξεις Κλειδιά

Σήμανση Μερών τού Λόγου, Λημματισμός, Επεξεργασία Φυσικής Γλώσσας (Natural Language Processing, NLP), Μέθοδοι Πυρήνων (Kernel Methods), Γενικευμένοι Πυρήνες Συμβολοσειρών (Generalized String Kernels), Μηχανές Διανυσμάτων Υποστηρίξεως (Support Vector Machines, SVM), Υπό Συνθήκη Μαρκοβιανά Πεδία (Conditional Random Fields, CRF).

Abstract

This thesis presents a general method for labeling words within texts and finding their lemmata. The method is applied to ancient greek. More specifically, first, the part of speech and the inflection type is estimated for each word. Second, for each word form found in the text, the corresponding lemma is estimated, that is, the canonical form of the word as it is typically found in a dictionary. The approach does not rely on arbitrary rules but uses intelligent methods and machine learning. The system is not bound to ancient greek but it is designed in order to be able to serve all languages, especially the ones with rich morphological features, which present the most processing difficulty. The appropriate software engineering methodologies have been employed in order to address each language in an abstract and uniform way. This system is the first layer where higher services can be built upon, such as high quality search, machine translation, entity knowledge systems and semantic search.

Keywords

Part of Speech Tagging (POS Tagging), Lemmatization, Natural Language Processing (NLP), Kernel Methods, Generalized String Kernels, Support Vector Machines (SVM), Conditional Random Fields (CRF).

Περιεχόμενα

1 Εισαγωγή ... 1
1.1 Γενική ιδέα ... 1
2 Επισκόπηση συστήματος και μεθοδολογίας ... 3
2.1 Αναπαράσταση λέξεων ... 3
2.2 Πρώτη βαθμίδα: Χαρακτηρισμός μεμονωμένων λέξεων ... 4
2.3 Δεύτερη βαθμίδα: Σήμανση λέξεων εντός προτάσεων και ανίχνευση λημμάτων ... 5
3 Η πλατφόρμα και οι μονάδες τού συστήματος ... 6
3.1 Η επιλογή τής πλατφόρμας ... 6
3.2 Συσκευασία τών μονάδων ... 7
3.3 Μονάδα Gramma.Vectors ... 8
3.4 Μονάδα Gramma.GenericContentModel ... 10
3.5 Μονάδα Gramma.Serialization ... 12
3.6 Μονάδα Gramma.Parallel ... 13
3.7 Μονάδα Gramma.Linq ... 13
3.8 Μονάδα Gramma.Windows ... 13
3.9 Μονάδα Gramma.Caching ... 14
3.10 Μονάδα Gramma.BetaImport ... 15
3.11 Μονάδα Gramma.Optimization ... 15
3.11.1 Μέθοδος Συζυγούς Κλίσεως ... 15
3.11.2 Στοχαστική Κατάβαση Κλίσεως ... 17
3.12 Μονάδα Gramma.Indexing ... 18
3.12.1 Λειτουργικότητα Edit Distance ... 19
3.12.2 Δένδρα ... 19
3.12.3 Ο πυρήνας ακολουθιών ... 20
3.13 Η μονάδα Gramma.Kernels ... 22
3.13.1 Εισαγωγικά για τις μεθόδους πυρήνος ... 22
3.13.2 Η αρχιτεκτονική τής μονάδος Gramma.Kernels ... 25
3.14 Η μονάδα Gramma.SVM ... 28
3.14.1 Ορισμός τού Ταξινομητού Διανυσμάτων Υποστηρίξεως ... 28

3.14.2 Αρχιτεκτονική μονάδος Gramma.SVM 34
3.15 Μονάδα Gramma.CRF 35
3.15.1 Γενικά Πιθανοτικά Δίκτυα 35
3.15.2 Conditional Random Fields 36
3.15.3 Linear Chain Conditional Random Fields 37
3.15.4 Αρχιτεκτονική τής μονάδος Gramma.CRF 39
3.16 Μονάδα Gramma.LanguageModel 41
3.16.1 Το γραμματικό μοντέλο 42
3.16.2 Χειρισμοί συμβολοσειρών 44
3.17 Η μονάδα Gramma.LanguageModel.Provision.Greek 46
3.18 Το βοηθητικό πρόγραμμα LXXCombiner 48
3.19 Η μονάδα Gramma.Inference 48
3.19.1 Άντληση εκπαιδευτικών δεδομένων 49
3.19.2 Δήλωση γλωσσών και αντιστοίχων εκπαιδευτικών πηγών 50
3.19.3 Το σύστημα τών συλλαβικών εντολών και οι γραμματικές κλάσεις 54
3.19.4 Πρώτο στάδιο: Χαρακτηριστικά λέξεων 57
3.19.5 Δεύτερο στάδιο: Το δίκτυο αποτιμήσεως προτάσεων 60
3.20 Η μονάδα Gramma.LanguageModel.TrainingSources.Greek 64
4 Εκπαίδευση, αποτίμηση και συμπεράσματα 66
4.1 Τα προγράμματα τού συστήματος 66
4.2 Εκπαίδευση με το πρόγραμμα Gramma.TrainingApplication 67
4.2.1 Επεξήγηση παραμέτρων και λειτουργίας 67
4.2.2 Οι πηγές εκπαιδεύσεως για τα Αρχαία Ελληνικά 72
4.2.3 Ρυθμίσεις εκπαιδεύσεως για τα Αρχαία Ελληνικά 72
4.2.4 Συνολική αποτίμηση αποδόσεως 73
4.3 Ανάλυση παραδειγμάτων με το πρόγραμμα Gramma.Evaluator 74
4.3.1 Ρυθμίσεις λειτουργίας 74
4.3.2 Διάκριση σημάνσεως μιας λέξεως από τα συμφραζόμενα 76
4.3.3 Διάκριση λήμματος από τα συμφραζόμενα 76
4.3.4 Επιλογή γένους επιθέτων 78
4.3.5 Αναγνώριση εντελώς αγνώστων λέξεων 80
4.3.6 Διφορούμενα γένη 82

4.4 Συμπεράσματα και μελλοντική εργασία 83
5 Βιβλιογραφία 84

1 Εισαγωγή

Σκοπός της διπλωματικής εργασίας είναι σήμανση μερών τού λόγου και ο λημματισμός αρχαίων ελληνικών κειμένων. Μέ τον όρο σήμανση (tagging) εννοούμε τον γραμματικό χαρακτηρισμό των λέξεων μέσα στο κείμενο, για παράδειγμα, «οὐσιαστικὸν, γένους θηλυκοῦ, αἰτιατικὴ πτῶσις, ἑνικὸς ἀριθμός». Με τον δε λημματισμό εννοούμε την αναγωγή μιας μορφής λέξεως στην κανονική της μορφή, το λεγόμενο «λήμμα», όπως αναγράφεται τυπικώς σ' ένα λεξικό. Επί παραδείγματι, για την μορφή «θαλασσῶν» το λήμμα είναι «θάλασσα», για την μορφή «διέλυσαν» το λήμμα είναι «διαλύω».

Ο διττός στόχος αυτός είναι ένα πρώτο στάδιο για πλήθος εφαρμογών, όπως η υψηλής ποιότητος αναζήτηση, η μηχανική μετάφραση, συστήματα οντολογίας καὶ εννοιολογική αναζήτηση.

Παρ' ότι τα αρχαία ελληνικά είναι ο στόχος τής εργασίας, η υλοποίηση δεν εξαρτάται από την γλώσσα. Το σύστημα εργάζεται κατά ένα αφαιρετικό τρόπο ώστε να μπορεί νά εφαρμοσθεί σε όλες τις γλώσσες, ειδικά αυτές που παρουσιάζουν πλούσια μορφολογία όπως τα αρχαία ελληνικά. Τα ειδικά χαρακτηριστικά της εκάστοτε γλώσσας απομονώνονται από το κυρίως σύστημα και προσαρμόζονται μέσω διεπαφής τύπου "inversion-of-control" κατά την ορολογία της τεχνολογίας λογισμικού. Στην διεπαφή αυτή περιγράφονται:

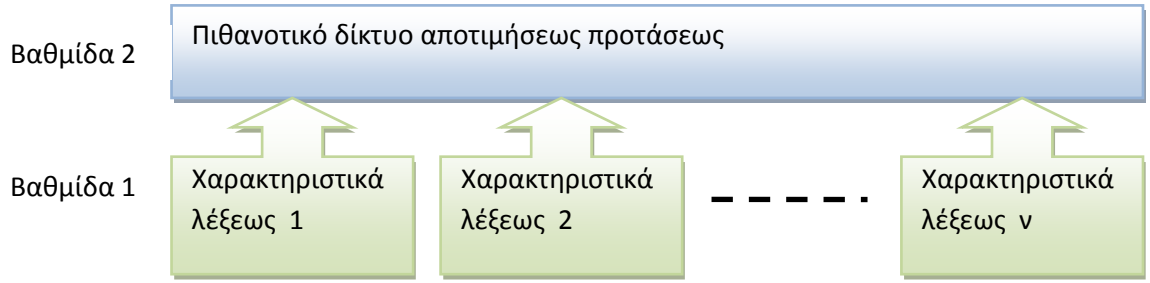
- Οι δυνατές γραμματικές κλάσεις και οι σημάνσεις τις οποίες μπορεί νά λάβει μία λέξη.
- Η κατάτμηση προτάσεως σε λέξεις.
- Ο κατάτμηση λέξεως σε συλλαβές και η επανασυναρμολόγηση συλλαβών σε λέξη.
- Προαιρετικά, ένα μέτρο αποστάσεως μεταξύ συλλαβών.

Ο σκοπός της αρχιτεκτονικής αυτής είναι να μειώσει την ανθρώπινη προσπάθεια για την περιγραφή της γλώσσας στα απολύτως απαραίτητα και να αφήσει την μηχανή να μάθει τους κανόνες μορφολογίας και τους συντακτικούς περιορισμούς τής γλώσσας αυτομάτως από εκπαιδευτική ύλη.

1.1 Γενική ιδέα

Προς την επίτευξη αυτών τών στόχων, η προσέγγιση γίνεται σε δύο βαθμίδες. Η προσέγγιση είναι κατά μίμηση της ανθρωπίνης αντιλήψεως, όπου πρώτα σχηματίζονται στον νου οι δυνατές εκδοχές για τον ρόλο κάθε λέξεως ξεχωριστά, έπειτα με τα συμφραζόμενα κάθε λέξεως ενισχύονται κάποιες εκδοχές και αποκλείονται άλλες. Όταν ακούμε ή διαβάζουμε μεμονωμένη λέξη, στον νού δημιουργούνται παράλληλες πιθανές εκδοχές υποθέσεις για το μέρος του λόγου οπου ανήκει ή ακόμη και για το ποια είναι η ίδια η λέξη μεταξύ ομοήχων ή ομοίων γραφών. Παραδείγματα: «πρᾶγμα» μπορεί να είναι στην ονομαστική ή την αιτιατική ή τήν κλητική. «φέρουσι» μπορεί να είναι τρίτο πρόσωπο πληθυντικού οριστικής ενεστώτος, αλλά μπορεί να είναι δοτική πληθυντικού μετοχής ενεστώτος, αρσενικό ή ουδέτερο γένος. Μπορεί ακόμη το γραφόμενο να είναι μορφή διαφορετικών πιθανών λέξεων. Για παράδειγμα, «μένει» μπορεί να είναι το τρίτο πρόσωπο ενικού οριστικής τού «μένω», αλλά μπορεί να είναι

η δοτική ενικού τής λέξεως «μένος». Έπειτα όμως οι πολλές αντικρουόμενες εκδοχές περιορίζονται και κατασταλάζουν όταν συντίθενται με τα συμφραζόμενα τής προτάσεως. Έτσι, όταν μεν διαβάζουμε «ἐξεκαύθη τῷ μένει» γνωρίζουμε ὅτι εἶναι ἡ λέξη «μένος», όταν διαβάζουμε δε «οὗτος μένει εἰς τὸν αἰῶνα» ξέρουμε ότι είναι το ρήμα «μένω».



Σχήμα 1-1: Γενικὴ αρχιτεκτονική τής μεθοδολογίας

Η γενική αρχιτεκτονική τού συστήματος, εικονιζομένη στο Σχήμα 1-1, ακολουθεί αυτήν την βασικήν ιδέα τών δύο βαθμίδων. Στην πρώτη βαθμίδα, κάθε λέξη εντός κειμένου αναλύεται μεμονωμένα και από αυτήν εξάγονται σήματα που εκφράζουν ενδεχόμενα να είναι ανήκει η λέξη σε κάποιο μέρος τού λόγου και να παράγεται η δεδομένη μορφή της από κάποιο κανόνα. Τα σήματα αυτά μπορεί να είναι στατιστικώς αλληλοεξαρτώμενα ή αλληλοσυγκρουόμενα, και τυπικώς είναι χιλιάδες ανά λέξη. Έπειτα, στην επομένη βαθμίδα, οι δέσμες τών χιλιάδων σημάτων για κάθε λέξη μιας προτάσεως συντίθενται ώστε νὰ διασαφηνισθεί σε σχέση με τα συμφραζόμενα το μέρος τού λόγου όπου ανήκει κάθε λέξη καθώς και το λήμμα της.

Το σύστημα τρέχει σε περιβάλλον .NET και έχει υλοποιηθεί εκ τού μηδενός χωρίς χρήση άλλων βιβλιοθηκών πλην αυτών τού συστήματος. Με αυτόν τον τρόπο δόθηκε ευκαιρία στην εμβάθυνση τών επιμέρους μεθόδων αλλά και ευχέρεια για καινοτομία όπου ήταν απαραίτητο για τις ανάγκες τού έργου.

Στην συνέχεια, ακολουθεί επισκόπηση τού συστήματος και τής μεθοδολογίας σε γενικό επίπεδο, έπειτα έκθεση τών μονάδων που αποτελούν τό σύστημα και τής σχετικής θεωρίας, παρουσίαση τών προγραμμάτων τού συστήματος, παραδείγματα εκτελέσεως και πειραματικά δεδομένα. Στο τέλος εξάγονται συμπεράσματα και ιδέες για μελλοντική εξέλιξη.

2 Επισκόπηση συστήματος και μεθοδολογίας

Πριν φθάσουμε στην ανάλυση τού συστήματος, θα ανατρέξουμε μίαν επισκόπηση.

2.1 Αναπαράσταση λέξεων

Η πρόκληση για το σύστημα είναι να μάθει τήν μορφολογία τής γλώσσας, δηλαδή τούς μηχανισμούς κλίσεώς της. Όπως θα δούμε, αυτό μεταφράζεται ως χιλιάδες ταξινομητές ανά λέξη. Αυτοί θα μπορούσαν να έχουν ως είσοδο την λέξη αναπαρισταμένη με συνήθη συμβολοσειρά. Η παρούσα εργασία όμως δανείζεται έμπνευση από άλλο γνωστικό τομέα, με αφορμή την αποκρυπτογράφηση τής Γραμμικής Γραφής Β.

Με τις εργασίες τού Ventris και την συνέργεια τού Chadwick αποκρυπτογραφήθηκε η Γραμμική Β το 1952 και βρέθηκε ότι ήταν ελληνικά [1]. Το ενδιαφέρον ήταν ότι αποκρυπτογραφήθηκε χωρίς παράβολο κείμενο σε γνωστή γλώσσα, εν αντιθέσει με τα αιγυπτιακά όπου το κλειδί ήταν η περίφημος στήλη τής Ροζέτας η οποία περιείχε το ίδιο κείμενο σε τρεις γραφές, δύο αιγυπτιακές και μία ελληνική. Μόνη βοήθεια στο τέλος ήταν το κυπριακό συλλαβάριο, γραφή η οποία επέζησε έως τους κλασικούς χρόνους, η οποία όπως απεδείχθη είλκε καταγωγή από την γραμμική Β. Η Γραμμική Β είναι μία συλλαβική γραφή, δηλαδή κάθε γράμμα συμβολίζει συλλαβή. Στο συμπέρασμα αυτό έφθασε αρχικώς η Alice Kober επειδή τα 87 περίπου γράμματα ήταν πάρα πολλά γιά να είναι φθόγγοι, πολύ λίγα για να είναι ιδεογράμματα πλήρων λέξεων. Η Kober επίσης βασίσθηκε στην συλλαβική υφή τής γραφής και κατάφερε να κάνει στατιστική ανάλυση τών καταλήξεων τών λέξεων καί συνέταξε τους πρώτους πίνακες «κλίσεων». Την σκυτάλη παρέλαβε ο Michael Ventris, ο οποίος σημπλήρωνε και εμπλούτιζε τους πίνακες αυτούς σε φάσεις 20 «αναδρομών». Στην τελευταία αναδρομή αποπειράθηκε με επιτυχία να προσδώσει φωνητική αξία στά συλλαβογράμματα αναζητώντας τοπωνύμια και αναφερόμενος στο κυπριακό συλλαβάριο. Το σημαντικό είναι ότι μέχρι την 20^{η} αναδρομή ο Ventris είχε την προκατάληψη ότι η γραφή είνα ι Ετρουσκική, μια γλώσσα πολύ μακράν της Ελληνικής αφού δεν ανήκει στην Ινδοευρωπαϊκήν οικογένεια. Όμως η συλλαβική φύση της γραφής με την στατιστική ανάλυση τον ωδήγησε με ασφάλεια στην αναγνώριση της μορφολογίας και τών φθόγγων τής Ελληνικής, με επιβεβαίωση από τον Chadwick.

Η παρούσα εργασία έλκει από τα παραπάνω την ιδέα τής αναπαραστάσεως μιας λέξεως με συλλαβές αντί γραμμάτων. Οι συλλαβές γίνονται ένα είδος «μεταγραμμάτων» για το σύστημα. Οι μορφολογικές αλλαγές μιας λέξεως κατά την κλίση αναπαριστώνται με πολύ συμπαγέστερο και δηλωτικώτερο τρόπο ως πάθη συλλαβών από ότι ως πάθη γραμμάτων. Έτσι, η συλλαβική αναπαράσταση οδηγεί ευκολώτερα σε στατιστικό συμπερασμό για την μορφολογία της γλώσσας, όπως ωδήγησε και τους Kobe, Ventris, Chadwick.

Έχει αναφερθεί ότι ο συλλαβισμός είναι ένα από τα καθήκοντα τού προσαρμογέως γλώσσας στο σύστημα. Για την προσαρμογή των αρχαίων ελληνικών, δεν ακολουθείται ο γνωστός συλλαβισμός κατά την γραμματική αλλά η κατάτμηση γίνεται με βάση την μεγίστη διευκόλυνση για στατιστικό συμπερασμό. Έχει λοιπόν προκριθεί η τμήση συλλαβών ώστε, αν

είναι δυνατόν, ν’ αρχίζουν από φωνήεν, και εάν είναι δυνατόν, να τελειώνουν σε σύμφωνο. Έτσι, η λέξη «ἀνθρώπων» συλλαβίζεται ὡς «ἀνθρ-ωπ-ων», αφού γίνει κανονικοποίηση τόνων.

Η επιλογή για συλλαβική αναπαράσταση ισχυροποιείται από την επιτυχία της σε έργα ταξινομήσεως κειμένου [2].

Η κανονικοποίηση τόνων, ένα ακόμη καθήκον τού προσαρμογέως γλώσσας, αφαιρεί την οξεία, την βαρεία και την περισπωμένη από την λέξη και αφήνει μόνο τα πνεύματα, εκτός αν η λέξη έχει έως τρία γράμματα, οπότε δεν αφαιρείται τίποτε. Αυτή η πολιτική φροντίζει και για τις περιπτώσεις «ἄνθρωπός τις», «οἷόν τε», «ῥῦσαί με» κλπ.

2.2 Πρώτη βαθμίδα: Χαρακτηρισμός μεμονωμένων λέξεων

Ανεφέρθη ότι η πρώτη βαθμίδα για κάθε λέξη παράγει χαρακτηριστικά σήματα. Αυτά αναπαριστώνται με δέσμες αριθμητικών τιμών. Όσο πιο θετική ειναι η τιμή, τόσο μεγαλύτερη πεποίθηση υπάρχει για την παρουσία ενός χαρακτηριστικού. Για κάθε χαρακτηριστικό λοιπόν υπάρχει ένας ταξινομητής. Ο ταξινομητής βλέπει το χαρακτηριστικό ως κλάση, ώστε, γιά κάθε προσφερομένη λέξη να αποφαίνεται αν η λέξη ανήκει στην κλάση ή όχι. Ο ταξινομητής δηλώνει την απόφασή του επιστρέφοντας την αναφερθείσα αριθμητική τιμή.

Τι είδους χαρακτηριστικά ή κλάσεις όμως αναγνωρίζουν οι ταξινομητές; Ένα χαρακτηριστικό αποτελείται από την γραμματική σήμανση τής λέξεως (πχ «ἐπίθετο, γένους θηλυκοῦ, αἰτιατικὴ πτῶσις, ἑνικὸς ἀριθμός») συνοδευομένη από την ακολουθία συλλαβικών εντολών οι οποίες όταν εφαρμοσθούν στην δεδομένη λέξη οδηγούν στο λήμμα της. Για το παράδειγμά μας, η σήμανση συνοδεύεται από μία ακολουθία που περιέχει μόνο μία εντολή: «αντικατάστησε την λήγουσα μὲ την συλλαβή “ος”». Σ’ αυτήν την κλάση λοιπόν ανήκουν οι μορφές «χρησίμην», «θετικήν», δηλαδή όλα τα θηλυκά επίθετα σε αιτιατική, τα οποία σχηματίζουν θηλυκό κατά την «πρώτην κλίσιν» και αρσενικό κατά την «δευτέραν κλίσιν». Κατά την εκπαίδευση, με αυτήν την δομή «κλάσεων», το σύνολο τών ταξινομητών μαθαίνει την μορφολογία τής γλώσσας.

Η απόφαση για την κλάση μιας λέξεως δεν είναι μονοσήμαντος. Μπορεί πολλοί διαφορετικοί ταξινομητές να αποφαίνονται θετικώς για την ίδια λέξη και να έχουν δίκιο. Για παράδειγμα, για τις λέξεις «πράγματα», «θέματα», «ἅρματα», «βρώματα» και τα παρόμοια, αποφαίνονται θετικώς τρεις ταξινομητές: για την κλάση «οὐσιαστικόν, γένους οὐδετέρου, πληθυντικὸς ἀριθμός , πτῶσις ὀνομαστική, ἀκολουθία ἐντολῶν: ἀφαίρεσις ληγούσης», για την αντίστοιχο κλάση «πτῶσις αἰτιατική» καὶ την κλάση «πτῶσις κλητική». Ακόμη, στο παράδειγμα της εισαγωγής, για την λέξη «μένει» απαντούν θετικώς δύο ταξινομητές, ένας για την περίπτωση που είναι «ρῆμα, χρόνος ἐνεστώς, ἔγκλισις ὁριστική, τρίτον πρόσωπον ἑνικοῦ» και ένας για «οὐσιαστικόν, γένους οὐδετέρου, ἑνικὸς ἀριθμός, πτῶσις δοτική». Στο τελευταίο παράδειγμα, παρατηρούμε ότι οι δύο θετικοί ταξινομητές όχι μόνο αποφαίνονται για διαφορετική γραμματική σήμανση αλλά προτείνουν και δύο διαφορετικά λήμματα, «μένω» και «μένος».

Όπως και ο άνθρωπος, από μόνη την λέξη η πρώτη αυτή βαθμίδα δεν μπορεί να συναγάγει σε ποια κλάση ανήκει η λέξη από αυτές που βρέθηκαν θετικές. Την διασαφήνιση αυτή αναλαμβάνει η δεύτερη βαθμίδα η οποία λαμβάνει υπ΄ όψιν τα συμφραζόμενα.

2.3 Δεύτερη βαθμίδα: Σήμανση λέξεων εντός προτάσεων και ανίχνευση λημμάτων

Πρώτον, σε αυτήν την βαθμίδα γίνεται η απόφανση για την γραμματική σήμανση κάθε λέξεως μέσα σε μία πρόταση. Οι πληροφορίες που συνδυάζονται είναι δύο είδών: Το πρώτο είδος είναι οι αριθμητικές τιμές «αποφάσεις» που καταφθάνουν από τους ταξινομητές τής πρώτης βαθμίδος για κάθε λέξη. Το δεύτερο είδος είναι η συσχέτιση τών άλλων λέξεων στην πρόταση.

Δεύτερον, αφού εκτιμηθεί η μία μοναδική σήμανση για κάθε λέξη, τότε το σύστημα ανατρέχει στην πρώτη βαθμίδα να βρεί ποιος ταξινομητής «ψήφισε» υπέρ τής εκλεγείσης γραμματικής σημάνσεως. Αν αυτοί οι ταξινομητές είναι πολλοί, επιλέγεται αυτός που συνεισέφερε περισσότερο στην «εκλογή» σημάνσεως. Τότε, επί τής λέξεως εφαρμόζεται η ακολουθία εντολών που βρίσκεται μέσα στην κλάση τού επιλεγμένου ταξινομητού, ώστε να παραχθεί το εκτιμώμενο λήμμα τής λέξεως.

3 Η πλατφόρμα και οι μονάδες τού συστήματος

Στην ενότητα αυτή θα αιτιολογηθεί η επιλογή πλατφόρμας και θα παρουσιαστούν οι μονάδες τού συστήματος από χαμηλά προς τα υψηλά, δηλαδή από τα προαπαιτούμενα προς τα εξαρτώμενα. Όπου χρειασθεί, θα γίνει σύντομη θεωρητική παρουσίαση και παραπομπές.

Οι μονάδες αυτές έχουν εκπονηθεί από του μηδενός για την εργασία αυτή, ωστε να δοθεί η ευκαιρία κατανοήσεως των επιμέρους αντικειμένων, αλλά και να είναι ευχερής η προσθήκη καινοτομιών για τις ανάγκες τού έργου.

3.1 Η επιλογή τής πλατφόρμας

Το σύστημα έχει εκπονηθεί σε .NET εκδόσεως 4.0, και είναι κυρίως γραμμένο σε γλώσσα C# μέσω Visual Studio 2010.

Ο γράφων έχει εμπειρία και στην Java, και θα μπορούσε να ακολουθήσει αυτήν την πλατφόρμα. Όμως η επιλογή τού .NET έγινε για τους εξής λόγους: Πρώτον, ήδη από το 2006 η πλατφόρμα .NET έχει συναρτησιακά χαρακτηριστικά (λάμδα όροι, extensions methods, closures, LINQ [3], parallel LINQ) ώστε να επιτρέπει συνοπτική και ευανάγνωστο δηλωτική περιγραφή τής λύσεως ενός προβλήματος αντί επιπόνου λεπτομερούς προστακτικής περιγραφής. Μερικά από τα παραπάνω χαρακτηριστικά εμφανίσθηκαν μόλις στην Java 8, η οποία δεν ήταν διαθέσιμη κατά την έναρξη τής εργασίας.

Δεύτερον και σημαντικώτερον, ο τρόπος που υλοποιεί η Java τα generics είναι εξαιρετικά επιζήμιος για την απόδοση, ειδικά σε εργασίες με έμφαση σε υπολογισμούς καθώς η παρούσα [4]. Όπως είναι γνωστό, για την Java ο πολυμορφισμός τύπων αφορά μόνο τον μεταγλωττιστή, ο οποίος αφαιρεί τον πολυμορφισμό κατά την παραγωγή κώδικος με μια διαδικασία γνωστή ως “type erasure”. Ο σκοπός τής στρατηγικής ήταν να μπορεί νεώτερος κώδικας να τρέχει σε απαράλλακτα παλαιότερα περιβάλλοντα Java. Έτσι, ο τελικός κώδικας χρησιμοποιεί τύπους χωρίς πολυμορφισμό και διενεργεί type casts παντού κατά τις δοσοληψίες αντικειμένων, όπως ακριβώς και προ τής υπάρξεως generics στην πλατφόρμα, προσθέτει και καλεί δε τα λεγόμενα “bridge methods” όπου χρειάζεται. Αν δε ο πολυμορφικός τύπος βασίζεται σε primitive types, πχ ArrayList<double>, πράγμα πολύ σύνηθες για υπολογιστικά έργα όπως το παρόν, τότε για κάθε δοσοληψία primitive τιμής επιπλέον γίνεται “boxing” και “unboxing”, δηλαδή εγκιβωτισμός τού primitive σε αντικείμενο αποθηκευμένο στον σωρό, ώστε στο παράδειγμα να έχουμε ArrayList<Double> και να γίνονται συνεχείς μετατροπές μεταξύ double και Double. Πίσω δηλαδή από ένα σωστό κώδικα, μέσα σε κρισίμους βρόχους που εκτελούνται εκατομμύρια φορές, υπάρχουν μεγάλα κρυφά κόστη όπως type casts, κλήση “bridge” μεθόδων οι οποίες μάλιστα είναι “virtual” εμποδίζοντας inlining, “boxing” και “unboxing”, πχ μεταξύ double και java.lang.Double, τα οποία επιφέρουν το άμεσο βάρος συχνής κατανομής αντικειμένων στον σωρό αλλά και έμμεσο λόγω υπερφορτώσεως τού garbage collector.

Η υλοποίηση τών generics στήν πλατφόρμα .NET είναι διαφορετική, όπου η πολυμορφικότης τών τύπων υποστηρίζεται και από το περιβάλλον εκτελέσεως. Με την στρατηγική αυτή,

υπάρχει συμβατότης μόνο προς τα πίσω, δηλαδή νέο περιβάλλον μπορεί να τρέξει παλαιό κώδικα χωρις generics, δεν μπορεί όμως παλαιό περιβάλλον να τρέξει νέο κώδικα με generics. Αυτή η συνθήκη κρίνεται λογική και μη περιοριστική για το παρόν έργο. Το κέρδος όμως είναι ότι δεν γίνονται ποτέ type castings ή “bridge methods”. Μάλιστα δεν χρειάζονται “boxing-unboxing” και για τα primitives, αλλά και για την ευρύτερη οικογένεια τών “value types” τού .NET, τιμών που κατανέμονται στην στοίβα, στην οποία οικογένεια ανήκουν και τα primitives. Η πρόσβαση στις τιμές γίνεται απ’ ευθείας.

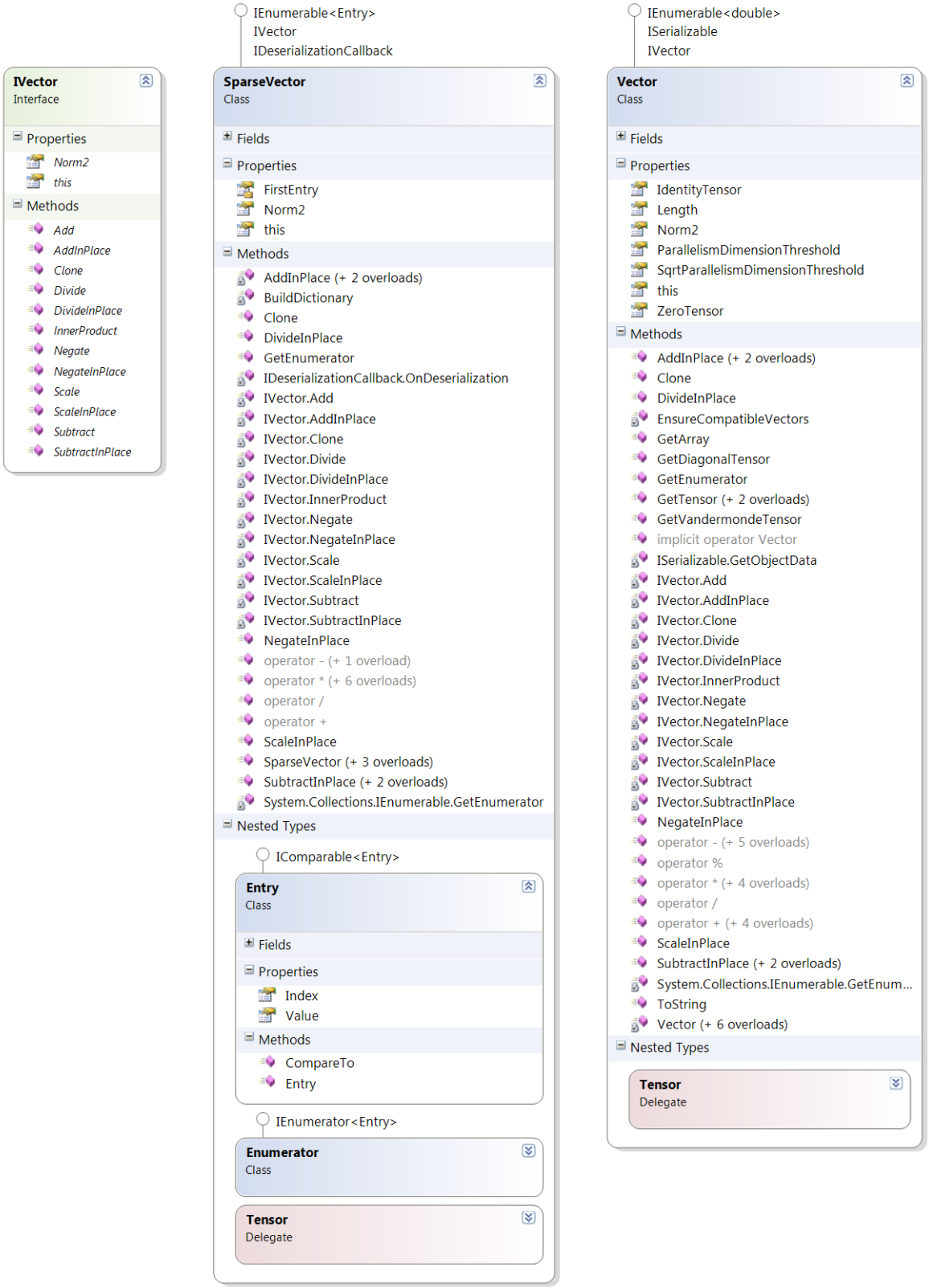
Το παρόν έργο είναι πολύ απαιτητικό σε υπολογιστική ισχύ, ώστε, με βάση τους διαθεσίμους υπολογιστικούς πόρους, τα παραπάνω χαρακτηριστικά κρίνουν το εφικτόν τού εγχειρήματος.

3.2 Συσκευασία τών μονάδων

Όλες οι μονάδες κατά κανόνα βρίσκονται μέσα σε namespace (αντίστοιχο τού Java package) με κοινή ρίζα το όνομα “Gramma” και είναι υλοποιημένες σε C# εκτός αν αναφέρεται άλλο. Οι ονομασίες τών μονάδων είναι οι ρίζες των namespaces που καταλαμβάνουν.

Όλες οι μονάδες ανοίγουν ως μέρος τού Visual Studio 2010 “solution” με όνομα Gramma.sln.

3.3 Μονάδα Gramma.Vectors



Σχήμα 3-1: Τα βασικά συστατικά τής μονάδος Gramma.Vectors.

Σ’αυτήν την μονάδα υλοποιούνται πυκνά και αραιά διανύσματα, των οποίων τα στοιχεία είναι τύπου double. Το πυκνό διάνυσμα είναι η κλάση Vector, το αραιό είναι το SparseVector, διαγραμμα τών οποίων φαίνεται στο Σχήμα 3-1. Αμφότερες έχουν εξοπλισθεί με operator overloading, ένα χαρακτηριστικό τού περιβάλλοντος .NET, ώστε η πρόσθεση, αφαίρεση, κλιμάκωση, εσωτερικό γινόμενο διανυσμάτων να δίνονται μέ φυσική μαθηματική γραφή ”+, -, *”. Οι τελεστές αυτοί έχουν και εκδοχές για μικτές πράξεις μεταξύ αραιών και πυκνών διανυσμάτων. Για παράδειγμα, ορίζεται εσωτερικό γινόμενο ενός πυκνού και ενός αραιού διανύσματος, και πρόσθεση ενός πυκνού και ενός αραιού διανύσματος με αποτέλεσμα πυκνό διάνυσμα. Και οι δύο όμως κλάσεις υποστηρίζουν μία κοινή διεπαφή IVector. Αυτή η διεπαφή, επαυξάνοντας τους αναφερθέντες τελεστές, ορίζει γενικές μεθόδους για αριθμητική διανυσμάτων στο αφαιρετικό επίπεδο τής διεπαφής, για παράδειγμα:

```
IVector AddInPlace(IVector summand);
```

Με την χρήση τής διεπαφής IVector μπορεί ένας αλγόριθμος να αφαιρέσει την λεπτομέρεια υλοποιήσεως των διανυσμάτων, ώστε να μπορεί να εργασθεί εξ ίσου με πυκνά και με αραιά διανύσματα και να προσαρμόζεται η επίδοσή του στην φύση τών δεδομένων χωρίς να αλλάζει ο κώδικας.

Μία ακόμη λειτουργία που προσφέρουν αμφότερα τα Vector και SparseVecetor είναι ο συμβολισμός «τανυστών», δηλαδή συναρτήσεων με όρισμα διάνυσμα και αποτέλεσμα διάνυσμα:

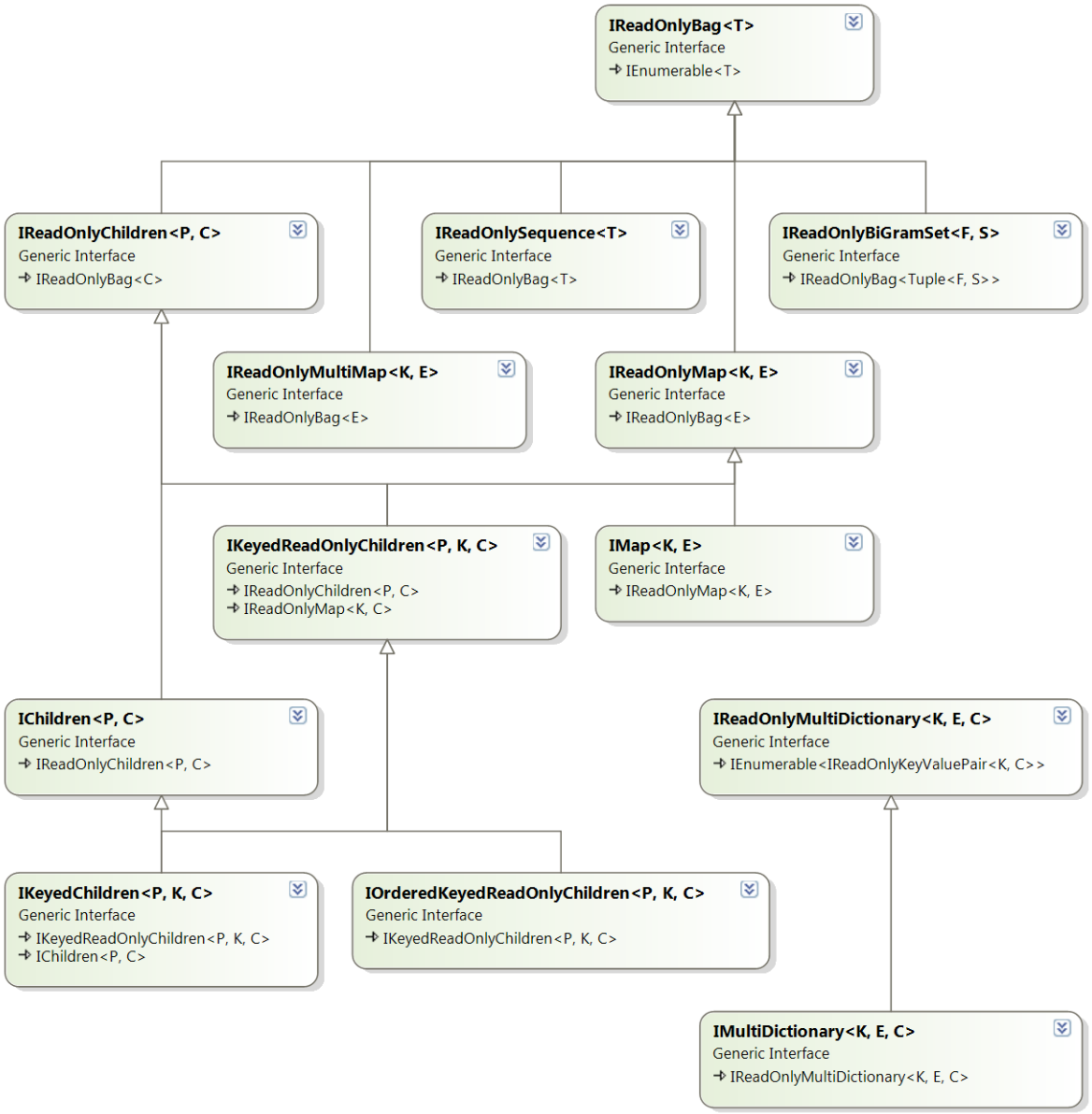
```
public delegate Vector Tensor(Vector input);
public delegate SparseVector Tensor(SparseVector input);
```

Με αυτόν τον τρόπο δηλούται αφαιρετικώς ως τανυστής τ_A ο πολλαπλασιασμός πίνακος A επί διάνυσμα, χωρίς να φανερώνεται η υλοποίηση:

$$\tau_A(x) = Ax \tag{3.3.1}$$

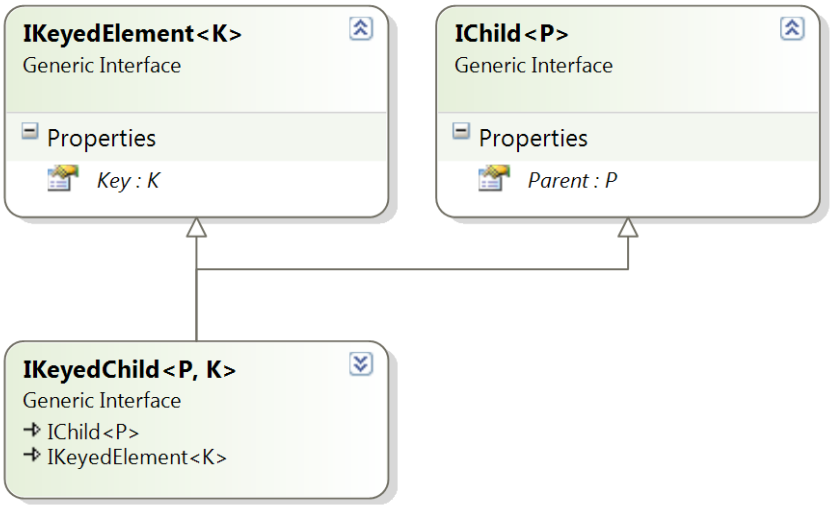
Έτσι, μπορεί η πράξη να αντιμετωπισθεί με ενιαίο τρόπο από αλγορίθμους, κρύβοντας ειδικές στρατηγικές αραιών δεδομένων ή παραλληλισμού ή ακόμη και μέσου αποθηκεύσεως, δηλαδή μπορεί τα δεδομένα τού πίνακος να είναι σε βάση δεδομένων, να ορίζονται αλγοριθμικώς ή να είναι στό «σύννεφο». Αυτό επιτρέπει και την δυνατότητα επιλύσεως πολύ μεγάλων προβλημάτων που εκφράζονται με πίνακες, όταν οι πίνακες αυτοί, ακόμη και αραιοί, δεν χωρούν στην μνήμη.

3.4 Μονάδα Gramma.GenericContentModel



Σχήμα 3-2: Οι διεπαφές συλλογών που προσφέρονται από την μονάδα Gramma.GenericContentModel.

Η μονάδα αυτή προσφέρει προηγμένες συλλογές για γενική χρήση. Στο Σχήμα 3-2 παρουσιάζονται οι διεπαφές τών συλλογών αυτών. Για κάθε μία διεπαφή ορίζεται και μία ομώνυμος κλάση, χωρίς το πρόθεμα “I”, που να την υλοποιεί. Ακολουθεί σύντομη περιγραφή διεπαφών:



Σχήμα 3-3: Διεπαφές προς υλοποίηση από αντικείμενα που προορίζονται να είναι μέλη ειδικών συλλογών τής μονάδος Gramma.GenericContentModel.

- **IReadOnlyBag<T>:** Ρίζα όλων σχεδών τών συλλογών στην μονάδα. Αναπαριστά ένα immutable σύνολο αντικειμένων τύπου T το οποίο μπορεί ν'απαριθμηθεί ή να παράσχει το πλήθος τών μελών του.
- **IReadOnlySequence<T>:** Κληρονόμος τής IReadOnlyBag<T>, με επιπλέον ιδιότητα ότι το σύνολο είναι διατεταγμένο. Τα στοιχεία μπορεί να είναι διπλά. Προσφέρεται random-access πρόσβαση.
- **IReadOnlyChildren<P, C>:** Κληρονόμος τής IReadOnlyBag<C>, με επιπλέον ιδιότητα ότι υλοποιεί την σχέση composite μεταξύ ενός πατέρα τύπου P και πολλών παιδιών τύπου C, με τον περιορισμό ότι ο τύπος παιδιού C πρέπει να υλοποιεί την διεπαφή IChild<P>, η οποία έχει ένα πεδίο "Parent" τύπου P, όπως φαίνεται στο σχήμα Σχήμα 3-3.
- **IChildren<P, C>:** Κληρονόμος τής IReadOnlyChildren<P, C>, ο οποίος όμως επιτρέπει την μεταβολή τού συνόλου μελών.
- **IReadOnlyMap<K, E>:** Κληρονόμος τής IReadOnlyBag<E>, με επιπλέον ιδιότητα ότι μπορεί να αναζητήσει κανείς μέλος τύπου E της συλλογής με βάση ένα κλειδί τύπου K. Το μέλος τύπου E έχει τον περιορισμό να υλοποιεί την διεπαφή τύπου IKeyedElement<K>, η οποία έχει ένα πεδίο "Key" τύπου K, όπως φαίνεται στο Σχήμα 3-3.
- **IMap<K, E>:** Κληρονόμος τής IReadOnlyMap<K, E>, η οποία όμως επιτρέπει την μεταβολή τού συνόλου μελών.
- **IKeyedReadOnlyChildren<P, K, C>:** Συνδυασμός τού IReadOnlyChildren<P, C> και τού IReadOnlyMap<K, C>. Δηλαδή, υλοποιεί την σχέση composite μεταξύ πατέρα και immutable συνόλου παιδιών με δυνατότητα αναζητήσεως παιδιού βάσει κλειδιού. Το παιδί C πρέπει να υλοποιεί την διεπαφή IKeyedChild<P, K>, η οποία είναι συνδυασμός τών διεπαφών IKeyedElement<K> και IChild<P>, όπως φαίνεται στο Σχήμα 3-3.

- **IKeyedChildren<P, K, C>:** Κληρονόμος τής παραπάνω διεπαφής IKeyedReadOnlyChildren<P, K, C>, όμως επιτρέπει την μεταβολή τού συνόλου τών παιδιών.
- **IOrderedKeyedReadOnlyChildren<P, K, C>:** Και αυτή η διεπαφή είναι κληρονόμος τής διεπαφής IKeyedReadOnlyChildren<P, K, C>>, με επιπλέον ιδιότητα ότι το σύνολο τών παιδιών είναι διατεταγμένο. Προσφέρεται random-access πρόσβαση.
- **IReadOnlyBiGramSet<F, S>:** Υλοποιεί ένα σύνολο διγραμμάτων τών οποίων το πρώτο μέρος είναι τύπου F και το δεύτερο τύπου S. Κληρονόμος τής IReadOnlyBag<Tuple<F, S>>, προσθέτει μεθόδους ταχείας αναζητήσεως βάσει τού πρώτου ή τού δευτέρου μέρους τών διγραμμάτων.
- **IReadOnlyMultiDictionary<K, E, C>:** Immutable συλλογή τύπου λεξικού με κλειδιά τύπου K, όπου μπορούν να υπάρχουν πολλά μέλη τύπου E με το ίδιο κλειδί. Τα μέλη υπό το ίδιο κλειδί φυλάσσονται σε συλλογή τύπου C, η οποία υποχρεούται να υλοποιεί την διεπαφή IReadOnlyBag<E>.
- **IMultiDictionary<K, E, C>:** Κληρονόμος τής IReadOnlyMultiDictionary<K, E, C>, όμως επιτρέπει την μεταβολή τού συνόλου τών μελών.

3.5 Μονάδα Gramma.Serialization

Όλη η αποθήκευση στο έργο γίνεται μέσω .NET serialization, στο οποίο ήδη είναι προσαρμοσμένες οι κλάσεις τού περιβάλλοντος .NET, και η ενεργοποίησή τού serialization για δικές μας κλάσεις είναι πολύ εύκολη. Το περιβάλλον .NET δίνει πολλές δυνατότητες. Προσθέτουμε απλώς ένα attribute “Serializable” στην κλάση. Εαν θέλουμε περισσότερο έλεγχο μπορούμε να προσθέσουμε “NonSerialized”, “OptionalField” attributes σε fields, να προσθέσουμε “OnSerializing”, “OnSerialized”, “OnDeserializing”, “OnDeserialized” attributes σε μεθόδους ώστε να καλούνται κατά το serialization και deserialization, να υλοποιήσουμε τις διεπαφές του περιβάλλοντος .NET “ISerializable”, “IDeserializationCallback” για λεπτομερή έλεγχο. Μπορούμε να προσθέσουμε ή να αλλάξουμε το serialization σε ήδη υπάρχουσες κλάσεις χωρίς να τις πειράξουμε με την χρήση τής διεπαφής “ISerializationSurrogate”.

Το περιβάλλον .NET δίνει ένα μηχανισμό τύπου “inversion of control” ώστε να μετατρέπονται οι γράφοι αντικειμένων σε ροές bytes. Παρέχει δηλαδή κλάσεις, οι οποίες υλοποιούν μία διεπαφή “IFormatter” και αναλαμβάνουν να μετατρέψουν γράφους αντικειμένων σε ροές. Το έργο αρχικά χρησιμοποιούσε την κλάση συστήματος “BinaryFormatter”, όμως επειδή οι γράφοι τού έργου είναι πολύ μεγάλοι, προσέκοψε σε άνω φράγμα τού “BinaryFormatter”.

Για να προσπερασθεί το πρόβλημα, έγινε έρευνα και αποτίμηση τών εναλλακτικών βιβλιοθηκών που υπάρχουν για serialization. Όλες απαιτούν σαφώς μεγαλύτερη προσπάθεια για τον ορισμό τού serialization γιατί η κάθε μία έχει τις δικές της συμβάσεις, μερικές εκ τών οποίων απαιτούν κάποια ειδική μέριμνα.

Αντί τής χρήσεως λοιπόν μιας έτοιμης βιβλιοθήκης που απαιτεί τελείως διαφορετικές συμβάσεις, εκπονήθηκε η μονάδα Gramma.Serialization η οποία πατάει στον υπάρχοντα μηχανισμό τού standard .NET serialization. Έτσι, δεν χρειάστηκε να αλλάξει ο υπάρχων κώδικας

τών κλάσεων που γίνονται serialize. Η μονάδα Gramma.Serialization περιέχει την κλάση "FastBinaryFormatter", η οποία υλοποιεί την διεπαφή "IFormatter" όπως και οι formatters του περιβάλλοντος .NET, και υποστηρίζει όλα τα αναφερθέντα χαρακτηριστικά τού serialization τού περιβάλλοντος .NET, αλλά είναι μάλιστα ταχύτερη, ειδικά κατά το deserialization. Επειδή είναι πολύ εύκολη, γενική και προφανής η χρήση της και ευεργετική για πολλούς, ο γράφων τήν εξέδωσε ως open source στο αποθετήριο CodePlex:

https://grammaserialization.codeplex.com/

Για αυτόματο εγκατάσταση τής μονάδος σε ένα project, υπάρχει και στο αποθετήριο τού ειδικού εργαλείου διαχειρίσεως βιβλιοθηκών NuGet (αναλόγου τού Ivy για Java):

https://www.nuget.org/packages/Gramma.Serialization/

Κατά την συγγραφή τού παρόντος, έχουν γίνει 125 μεταφορτώσεις τής μονάδος.

Συνοδεύεται από την μονάδα Gramma.Serialization.Testing, η οποία επιτελεί unit testing. Στο solution βρίσκεται στον φάκελο "Tests".

3.6 Μονάδα Gramma.Parallel

Αυτή η μονάδα μιμείται υποσύνολο τού Parallel LINQ ποὺ ήδη υπάρχει στο .NET περιβάλλον. Προορίζεται όμως για παραλληλισμό βρόχων των οποίων η αναδρομή διαρκεί πολύ, και αναθέτει τα τμήματα εργασίας σε αφιερωμένα threads, όχι από thread pool.

Η μονάδα παρέχει επί συλλογών το extension method "AsLongParallel", αντί τού καθιερωμένου "AsParallel", και έπειτα ακολουθεί LINQ έκφραση κατά τα συνήθη.

Συνοδεύεται από την μονάδα Gramma.Parallel.Testing που πραγματοποιεί unit testing. Στο solution βρίσκεται στον φάκελο "Tests".

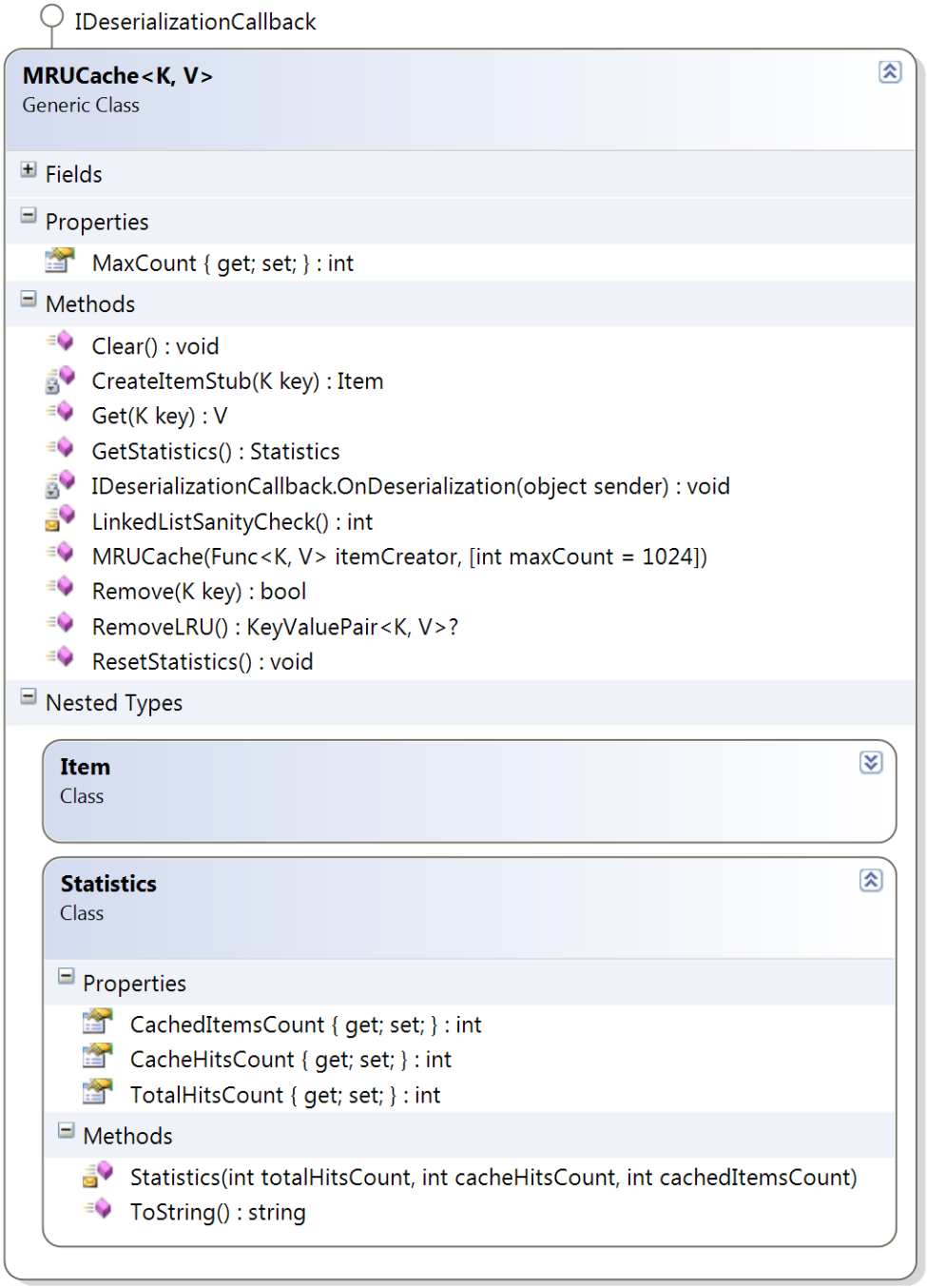
3.7 Μονάδα Gramma.Linq

Εδώ περιέχονται χρήσιμα extension methods. Για συλλογές, παρέχονται οι γνωστές μαθηματικές συναρτήσεις argmax και argmin. Επίσης, δίνει δυνατότητα από μονοδιαστάτους πίνακες να παραχθούν ατέρμονες ακολουθίες, είτε με τυχαία δειγματοληψία είτε με κυκλική σάρωση.

3.8 Μονάδα Gramma.Windows

Αυτή η μονάδα περιέχει χρήσιμα εργαλεία για σύνδεση δεδομένων με φόρμες τεχνολογίας Windows Presentation Foundation (WPF).

3.9 Μονάδα Gramma.Caching



Σχήμα 3-4: Η υλοποίηση cache στην μονάδα Gramma.Caching

Η μονάδα δίνει μία υλοποίηση cache η οποία συγκρατεί έως ενός ορίου τα τελευταία περισσότερο χρησιμοποιημένα στοιχεία (most recently used, MRU). Όπως φαίνεται στο Σχήμα 3-4, αποθηκεύει στοιχεία γενικού τύπου V υπό κλειδιά τύπου K. Τα στοιχεία που ζητούνται ενώ λείπουν από την cache (misses) κατασκευάζονται από συνάρτηση που παρέχεται.

Είναι ασφαλής για χρήση από ταυτόχρονα νήματα, και αν πολλά μαζί ζητήσουν το ίδιο cache miss, ένα μόνο νήμα διεκπεραιώνει την δημιουργία τού στοιχείου για λογαριασμό όλων.

3.10 Μονάδα Gramma.BetaImport

Η ανάγκη για αναπαράσταση ελληνικών γραμμάτων και δη πολυτονικών προϋπήρχε κατά πολύ τών πρωτοβουλιών προτυποποιήσεως. Έτσι, στις αρχές τής δεκαετίας 1970, ο David Packard, συνιδρυτής τής γνωστής εταιρείας Hewlett-Packard, για τις διατριβές του στα αρχαία ελληνικά και την συνεργασία του με την φιλόλογο Marianne McDonald υλοποίησε συστήματα τα οποία αναπαριστούν τα ελληνικά γράμματα επί τού συνόλου χαρακτήρων ASCII [5] μὲ κωδικοποίηση που επεκράτησε να λέγεται Beta code, όπως φαίνεται στο παράδειγμα:

Beta code	Σημαινόμενο
`*mh=nin a)/eide qea\ *phlhi+a/dew *)axilh=os ou)lome/nhn, h(\ muri/’ *)axaioi=s a)/lge’ e)/qhke, polla\s d’ i)fqi/mous yuxa\s *)/ai+di proi/+ayen h(rw/wn, au)tou\s de\ e(lw/ria teu=xe ku/nessin oi)wnoi=si/ te pa=si, *dio\s d’ e)telei/eto boulh/`	Μῆνιν ἄειδε, θεά, Πηληϊάδεω Ἀχιλῆος οὐλομένην, ἣ μυρί᾽ Ἀχαιοῖς ἄλγε᾽ ἔθηκε, πολλὰς δ᾽ ἰφθίμους ψυχὰς Ἄϊδι προΐαψεν ἡρώων, αὐτοὺς δὲ ἑλώρια τεῦχε κύνεσσιν οἰωνοῖσί τε πᾶσι, Διὸς δ᾽ ἐτελείετο βουλή

Εν απουσία προτυποποιήσεως, η κωδικοποίηση αυτή έγινε de facto πρότυπο. Δεν παραμένει μόνο στο κληροδότημα τής McDonald και τού Packard, τον Θησαυρό Ελληνικής Γλώσσης (Thesaurus Linguae Graecae, TLG), ο οποίος περιέχει το σύνολο σχεδόν τών ελληνικών κειμένων μέχρι το 1453 μ.Χ. [6], αλλά την κωδικοποίηση υιοθέτησαν ευρέως και άλλα έργα, όπως ο Περσεύς τού πανεπιστημίου Tufts [7] και πλήθος εργαλείων βιβλικής μελέτης.

Αυτός είναι ο λόγος που η παρούσα εργασία καταναλίσκει πολλές πηγές εκπαιδεύσεως με χαρακτήρες κωδικοποιημένους όχι κατά Unicode αλλά Beta code. Επειδή η παράσταση χαρακτήρων στο σύστημα βασίζεται σε Unicode, η μονάδα Gramma.BetaImport φροντίζει για την μετατροπή τών συμβολοσειρών.

3.11 Μονάδα Gramma.Optimization

Εδώ παρέχονται μέθοδοι βελτιστοποιήσεως, δηλαδή εύρεση ελαχίστου συναρτήσεως, προαιρετικά υπό περιορισμούς. Συνοδεύεται από την μονάδα OptimizationTests η οποία παρέχει unit testing.

Δύο βασικές μέθοδοι βελτιστοποιήσεως παρέχονται, η μέθοδος Συζυγούς Κλίσεως και η Στοχαστική Κατάβαση Κλίσεως.

3.11.1 Μέθοδος Συζυγούς Κλίσεως

Η μέθοδος Συζυγούς Κλίσεως ή Conjugate Gradient [8] για βελτιστοποίηση κυρτής συναρτήσεως είναι δευτέρας τάξεως, δηλαδή προσεγγίζει την υπό βελτιστοποίηση διπλώς διαφορίσιμη κυρτή συνάρτηση f με όρους Taylor έως δευτέρου βαθμού σε τυχόν σημείο:

$$f(a+x) \cong f(a) + g^T x + \frac{1}{2} x^T H \mathrm{x} \tag{3.11.1}$$

Όπου g είναι το διάνυσμα κλίσεως στο σημείο x και H είναι ο Χεσιανός πίνακας:

$$g = \frac{\partial f(a)}{\partial x} \quad H = \frac{\partial f^2(a)}{\partial x^2} \tag{3.11.2}$$

Βάσει τής σχέσεως (3.11.1), η παράγωγος τής f είναι:

$$\frac{\partial f(a+x)}{\partial x} \cong g + Hx \tag{3.11.3}$$

Επειδή η συνάρτηση f είναι κυρτή, ο Χεσιανός πίνακας H είναι θετικώς ωρισμένος. Αυτό σημαίνει ότι υπάρχει ένα ολικό ελάχιστο και βρίσκεται στο σημείο τού μηδενισμού τής παραγώγου (3.11.3). Άν η συνάρτηση f είναι τετραγωνική, τότε οι σχέσεις (3.11.1) και (3.11.3) είναι ακριβείς. Αλλιώς, το σημείο που βρίσκουμε από τον μηδενισμό τής σχέσεως (3.11.3) είναι προσεγγιστικό, και επαναλαμβάνουμε την διαδικασία μέχρι να ικανοποιηθεί ένα κριτήριο συγκλίσεως, το οποίο συνήθως απαιτεί να πλησιάζει κατά κάποια νόρμα η κλίση g τό μηδενικό διάνυσμα. Επειδή η προσέγγιση (3.11.1) είναι δευτέρας τάξεως, περιλαμβάνει δηλαδή Χεσιανή η οποία εκφράζει την στρέβλωση τής συναρτήσεως, για κάθε βήμα η εκτίμηση τής κατευθύνσεως προς την οποία ελαχιστοποιείται η συνάρτηση είναι πολύ καλύτερη από αύτήν που υποδεικνύει μόνη η κλίση (που είναι πρώτης τάξεως προσέγγιση) ειδικά όταν υπάρχει μεγάλη στρέβλωση.

Άρα το πρόβλημα ανάγεται στην η λύση τού γραμμικού συστήματος που προκύπτει από την σχέση (3.11.3), δηλαδή:

$$g + Hx = \underline{0} \tag{3.11.4}$$

Ο πυρήνας τής ιδέας τής μεθόδου Συζυγούς Κλίσεως έγκειται στον τρόπο λύσεως τού γραμμικού συστήματος (3.11.4). Αποφεύγεται η αντιστροφή τού πίνακος H, και το σύστημα λύνεται με ένα επαναληπτικό αλγόριθμο ο οποίος εκμεταλλεύεται τον θετικό ορισμό τού πίνακος H και παράγει ακολουθία προσεγγίσεων x_i, ονομαζομένη Κρυλώφ. Οι προσεγγίσεις x_i είναι ολοένα συγκλίνουσες προς την λύση x. Ο αλγόριθμος δηλαδή, εν αντιθέσει με πολλούς άλλους για λύση γραμμικών συστημάτων, είναι διακόψιμος ώστε τα ενδιάμεσα αποτελέσματά του να είναι χρήσιμα. Αν οι διαστάσεις τού πίνακος H είναι $n \times n$, τότε κατά την θεωρία η αλγόριθμος φθάνει στην ακριβή λύση το πολύ σε n βήματα. Πρακτικώς όμως αναφύονται ζητήματα αριθμητικής ακριβείας αν κάνουμε όλες τις n επαναλήψεις. Αλλά με πολύ μικρό κλάσμα επαναλήψεων ο αλγόριθμος δίνει πολύ καλές προσεγγίσεις για την φύση τού προβλήματος. Αρκεί να θυμηθούμε, ότι λόγω τής προσεγγίσεως κατά Taylor δευτέρου βαθμού, ούτως ή άλλως ο μηδενισμός τής παραγώγου (3.11.3) είναι εν γένει προσεγγιστικός και πρέπει να τον επαναλάβουμε πολλές φορές μέχρι το κριτήριο τερματισμού. Η παρούσα υλοποίηση το εκμεταλλεύεται αυτό, ώστε κατά τούς πρώτους μηδενισμούς οι προσεγγίσεις τού x στο σύστημα (3.11.4) να είναι πολύ χονδρικές με υπολογισμό πολύ λίγων όρων τής ακολουθίας Κρυλώφ x_i, αφού η ακρίβεια δεν έχει νόημα, και προοδευτικά το πλήθος όρων να φθάνει ένα αριθμό, τυπικά $\sqrt[4]{n}$.

Επίσης, ο αλγόριθμος δεν χρειάζεται να έχει εκπεφρασμένη γνώση τών στοιχείων τού πίνακος H. Λαμβάνει τον ορισμό τού πίνακος H από την έκφραση τού τανυστού του $\tau_H(x) = Hx$. Όπως είδαμε στην μονάδα Gramma.Vectors, η έμμεσος έκφραση τού H μέσω τού τανυστού του δίνει πολύ μεγάλη ευελιξία, γιατί, αφού ο αλγόριθμος βλέπει μόνο ως ένα μαύρο κλειστό κουτί την συνάρτηση $\tau_H(x)$, αντιμετωπίζει με ενιαίο τρόπο αποδοτικά κάθε είδους δομή πίνακος, αραιότητος, παραλληλισμού, και επιτρέπει ώστε πολύ μεγάλος όγκος στοιχείων που δεν χωρεί στην μνήμη να βρίσκεται σε βάσεις δεδομένων, σε εξωτερικές συστοιχίες ή στο «σύννεφο».

Η μονάδα παρέχει βάσει τής μεθόδου Συζυγούς Κλίσεως αρκετές παραλλαγές. Προσφέρει βελτιστοποίηση κυρτής συναρτήσεως είτε χωρίς περιορισμούς είτε μέ περιορισμούς οι οποίοι εκφράζονται ως πλήθως ανισοτήτων $f_i(x) \leq 0$ ποὺ πρέπει να πληρούνται, όπου f_i είναι και αυτές κυρτές συναρτήσεις διπλώς διαφορίσμες. Για κάθε όμως τέτοια παραλλαγή δίνονται δύο υποπαραλλαγές. Η πρώτη υποπαραλλαγή χρησιμοποιεί τον τανυστή $\tau_H(x) = Hx$ τής Χεσιανής μήτρας όπως περιεγράφη παραπάνω. Η δεύτερη χρησιμοποιεί τεχνικές αναζητήσεως σε εύθεία (line search) ώστε να απαλλαγεί από την Χεσιανή τελείως.

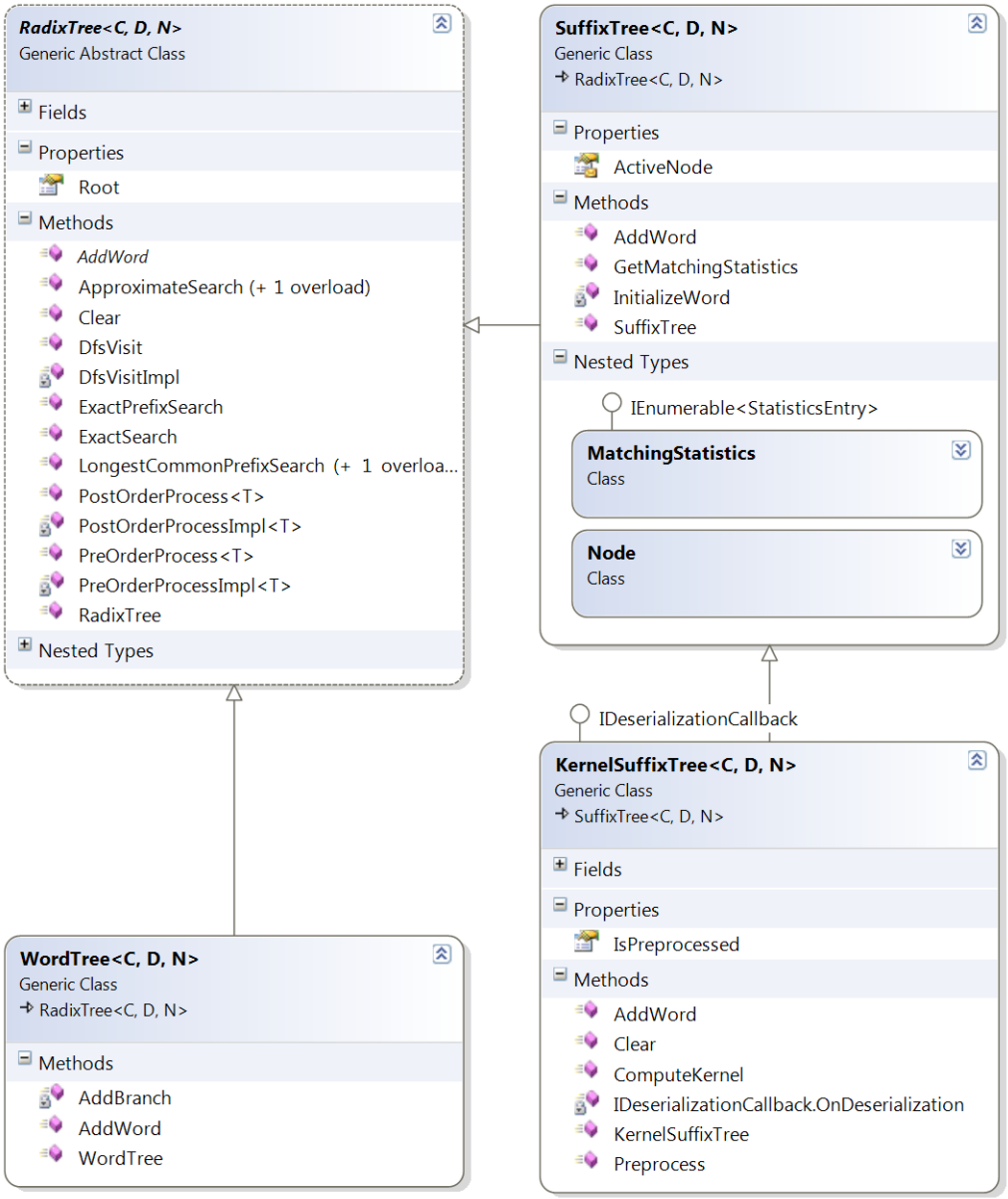
3.11.2 Στοχαστική Κατάβαση Κλίσεως

Από την μονάδα προσφέρεται και ένα άλλο είδος βελτιστοποιήσεως, η Στοχαστική Κατάβαση Κλίσεως ή Stochastic Gradient Descent [9], με προσθήκη παραλληλισμού. Βεβαίως η μέθοδος ανήκει στην γενική οικογένεια Καταβάσεως Κλίσεως, ως εκ τούτου κάνει πρώτης τάξεως προσέγγιση Taylor τής συναρτήσεως υπό βελτιστοποίηση εν αντιθέσει με την μέθοδο Συζυγούς Κλίσεως. Συνεπώς πάσχει από τα προβλήματα που απορρέουν από την πρώτης τάξεως προσέγγιση, ήτοι οι κατευθύνσεις καταβάσεως δεν είναι οι βέλτιστες όταν η συνάρτηση παρουσιάζει έντονο στρέβλωση, με αποτέλεσμα να χρειάζονται περισσότερες επαναλήψεις για σύγκλιση. Όμως έχει το πλεονέκτημα ότι εφαρμόζεται σε δεδομένα που καταφθάνουν σε συνεχή ροή, δηλαδή on-line, χωρίς προτέρα αποθήκευση στην μνήμη. Γι' αυτό, προϋποθέτει ότι η υπό βελτιστοποίηση συνάρτηση f γράφεται ως άθροισμα όρων ώστε ο καθε όρος να είναι συνάρτηση r που εξαρτάται και από ένα εισερχόμενο διάνυσμα δεδομένων a_k:

$$f(x) = \sum_k r(x; a_k) \tag{3.11.5}$$

Αυτού τού είδους συναρτήσεις προς βελτιστοποίηση όπως στην σχέση (3.11.5) απαντώνται συχνά στις εφαρμογές μηχανικής μαθήσεως, καθώς σε αυτές τα διανύσματα a_k απεικονίζουν τα δείγματα εκπαιδεύσεως. Και στο παρόν έργο υπάρχει τέτοια εφαρμογή.

3.12 Μονάδα Gramma.Indexing



Σχήμα 3-5: Τα δένδρα τής μονάδος Gramma.Indexing

Η μονάδα περιέχει ευρετήρια που ανήκουν στην κατηγορία “Radix Tree”. Επίσης περιέχει λειτουργικότητα “Generalized Levenshtein Edit Distance” για ακολουθίες [10], η οποία μπορεί να χρησιμοποιηθεί είτε σε συνδυασμό με τα ευρετήρια είτε μόνη της.

Στα επόμενα εσκεμμένα τονίζεται ο όρος «ακολουθία» έναντι τού όρου «συμβολοσειρά», γιατί η μονάδα βασίζεται σε γενικευμένους τύπους, είτε primitives είτε classes, ώστε μία συμβολοσειρά να θεωρείται απλώς ένα είδος ακολουθίας με στοιχεία τύπου char.

3.12.1 Λειτουργικότητα Edit Distance

Με τον όρο “Levenshtein Edit Distance” μεταξύ δύο ακολουθιών εννοούμε το ελάχιστο απαιτούμενο πλήθος αντικαταστάσεων, εισαγωγών ή διαγραφών στοιχείων της μιας ακολουθίας ώστε να λάβουμε την άλλη. Όταν προσθέτουμε τον όρο “Generalized” εννοούμε ότι στις ενέργειες «προσθήκη, αντικατάσταση, διαγραφή στοιχείου» μπορούμε να προσθέσουμε και άλλες ενέργειες εφ’ ενός στοιχείου αλλά και να αναθέσουμε ένα παράγοντα κόστους ώστε να προτιμώνται κάποιες ενέργειες αντί άλλων. Η υλοποίηση παρέχει όχι μόνο το ελάχιστο συνολικό κόστος μετατροπής τής μίας ακολουθίας στην άλλη ως «απόσταση», αλλά και αναφέρει την αλληλουχία ενεργειών οι οποίες επιτελούν αυτήν την βέλτιστη μετατροπή υπό μορφήν «εντολών».

3.12.2 Δένδρα

Στο άλλο σκέλος τής μονάδος, τά δένδρα είναι δομές για γρήγορη αναζήτηση και για πολλές άλλες χρήσιμες λειτουργίες επί ακολουθιών. Στα μεν μονοπάτια τών κλάδων τους καταγράφονται αποθηκευμένες ακολουθίες, στε δε φύλλα προαιρετικώς φυλάσσονται πληροφορίες που αντιστοιχούν στις καταλήγουσες ακολουθίες. Στο Σχήμα 3-5 βλέπουμε ότι τα δένδρα διέπονται από τις παραμέτρους τύπων C, D, N:

- **Παράμετρος C:** Είναι ο τύπος τού στοιχείου τής ακολουθίας. Αν δοθεί ως char, η ακολουθία είναι συμβολοσειρά. Μπορεί να ιδωθεί ως μεταχαρακτήρας μιας μετασυμβολοσειράς. Για καλή επίδοση, αναμένεται ότι ο τύπος C υλοποιεί τις μεθόδους “Equals” και “GetHashCode” σε χρόνο $O(1)$. Θα δούμε στην συνέχεια τού έργου ότι ως C δίνεται ο τύπος συλλαβής λέξεως.
- **Παράμετρος D:** Είναι ο τύπος τών δεδομένων που φυλάσσονται ανά ακολουθία. Δηλαδή είναι ο τύπος τών αντικειμένων τα οποία ανατίθενται στά φύλλα.
- **Παράμετρος N:** Είναι ο τύπος τών δεδομένων που φυλάσσονται στους κόμβους τού δένδρου.

Τα δένδρα είναι οργανωμένα στις εξής κλάσεις:

- **RadixTree:** Είναι η αφηρημένη βάση των δένδρων. Σε αυτήν ορίζονται οι κοινές δομές αλλά και οι εξής μέθοδοι που κληρονομούνται στις υλοποιήσεις:
    - **ExactSearch:** Αναζητεί στο δένδρο αν υπάρχει μονοπάτι από ρίζα μέχρι φύλλο που ταιριάζει ακριβώς σε μία δεδομένη ακολουθία ολόκληρη.
    - **ExactPrefixSearch:** Αναζητεί στο δένδρο όλα τα μονοπάτια που αρχίζουν από μία ολόκληρη δεδομένη ακολουθία.
    - **LongestCommonPrefixSearch:** Αναζητεί το μέγιστο δυνατό μονοπάτι το οποίο είναι πρόθεμα μιας δεδομένης ακολουθίας.
    - **ApproximateSearch:** Αναζητεί όλα τα πλήρη μονοπάτια από ρίζα έως φύλλο που έχουν απόσταση Generalized Edit Distance μικρότερη από ένα αριθμό σε σχέση με δεδομένη ακολουθία.
- **WordTree:** Κάθε μονοπάτι από την ρίζα σ΄ ένα φύλλο αναπαριστά μία πλήρη αποθηκευμένη «λέξη», μία πλήρη ακολουθία. Δηλαδή το δένδρο φυλάσσει πλήρεις

ακολουθίες. Ως εκ τούτου, οι παρακάτω μέθοδοι που κληρονομούνται από την παραπάνω κλάση RadixTree λαμβάνουν την εξής ερμηνεία:

- **ExactSearch:** Αναζητεί στο δένδρο αν υπάρχει αποθηκευμένη μία δεδομένη ακολουθία.
- **ExactPrefixSearch:** Αναζητεί στο δένδρο όλες τις ακολουθίες που αρχίζουν από μία δεδομένη ακολουθία.
- **LongestCommonPrefixSearch:** Αναζητεί το μέγιστο δυνατό κοινό πρόθεμα από όλες τις αποθηκευμένες ακολουθίες στο δένδρο με μία δεδομένη ακολουθία.
- **ApproximateSearch:** Αναζητεί στο δένδρο όλες τις αποθηκευμένες ακολουθίες που έχουν απόσταση Generalized Edit Distance κάτω από ένα ποσό σε σχέση με μία δεδομένη ακολουθία.

- **SuffixTree:** Κανονικά, ένα suffix tree περιέχει μονοπάτια από ρίζα σε φύλλο που αναπαριστούν μίαν ακολουθία και όλα τα επιθέματά της (suffixes). Η παρούσα κλάση όμως γενικεύει το suffix tree ώστε να περιέχει μονοπάτια για πολλές ακολουθίες και όλα τα επιθέματά τους. Δηλαδή, για κάθε ακολουθία μήκους n που ανατίθεται στο δένδρο, προστίθενται μονοπάτια για όλα τα n επιθέματά της. Για το κλασικό suffix tree τής μιας ακολουθίας, ο Ukkonen έδειξε αλγόριθμο [11] ώστε ένα δένδρο που αναπαριστά μίαν ακολουθία μήκους n να κατασκευάζεται σε χρόνο $O(n)$. Για το γενικευμένο suffix tree τής παρούσης εργασίας γίνεται η κατάλληλη επέκταση τού αλγορίθμου τού Ukkonen η οποία διατηρεί την γραμμική συμπεριφορά κατά χρόνο. Κάθε προσθήκη νέας ακολουθίας στο δένδρο μαζί με τα επιθέματά της χρειάζεται χρόνο ανάλογο τού μήκους της, με προϋπόθεση ότι ο τύπος χαρακτήρος C έχει μεθόδους “GetHashCode” και “Equals” με επίδοση χρόνου $O(1)$. Άρα οι μέθοδοι που κληρονομούνται από την κλάση RadixTree λαμβάνουν την παρακάτω ερμηνεία:
    - **ExactSearch:** Αναζητεί στο δένδρο αν υπάρχει αποθηκευμένη ακολουθία με επίθεμα την δεδομένη ακολουθία.
    - **ExactPrefixSearch:** Αναζητεί στο δένδρο όλες τις ακολουθίες οι οποίες περιέχουν μία δεδομένη ακολουθία σε συνεχείς θέσεις.
    - **LongestCommonPrefixSearch:** Αναζητεί την μεγίστη δυνατή κοινή συνεχή υπακολουθία από όλες τις ακολουθίες τού δένδρου με μία δεδομένη ακολουθία.
    - **ApproximateSearch:** Αναζητεί όλα τα επιθέματα τών αποθηκευμένων ακολουθιών στο δένδρο που έχουν απόσταση Generalized Edit Distance κάτω από ένα ποσό σε σχέση με μία δεδομένη ακολουθία.
- **KernelSuffixTree:** Ως απόγονος τής κλάσεως SuffixTree, κληρονομεί όλη την λειτουργικότητά της. Όμως προσθέτει την δυνατότητα πράξεως «πυρήνος» (kernel) μεταξύ ακολουθιών, την οποία εξετάζουμε ευθύς στο επόμενο.

3.12.3 Ο πυρήνας ακολουθιών

Περισσότερα για πυρήνες θὰ δούμε σε επομένην ενότητα και σχετική μονάδα. Από τώρα όμως ορίζουμε ως ένα πυρήνα K επί δύο στοιχείων x_1, x_2 το εσωτερικό γινόμενο δύο διανυσμάτων

τα οποία προκύπτουν με κάποια αυθαίρετο απεικόνιση ϕ τών στοιχείων x_1, x_2 στον χώρο τών διανυσμάτων:

$$K(x_1, x_2) = \phi(x_1) \cdot \phi(x_2) \tag{3.12.6}$$

Στην περίπτωσή μας τα στοιχεία x_1, x_2 είναι ακολουθίες. Υπάρχουν πολλές πρακτικές επιλογές για την συνάρτηση ϕ όταν έχουμε ακολουθίες. Η μονάδα αυτή υλοποιεί ένα ιδιαίτερα δυνατό και χρήσιμο πυρήνα, λεγόμενο «πυρήνα όλων τών υπακολουθιών» (all-substrings kernel). Κατ'αυτόν τον πυρήνα, θεωρούμε συνάρτηση απεικονίσεως ϕ ως εξής.

Ας θεωρήσουμε το σύνολο S τὸ όπου ανήκει κάθε στοιχείο μιας ακολουθίας. Μπορεί να θεωρηθεί ως το σύνολο «γραμμάτων». Τότε, το σύνολο S^*, γνωστὸ ως το κλείσιμο τού S κατά Kleene, είναι το σύνολο όλων τών δυνατών ακολουθιών από στοιχεία που ανήκουν στο S, ή «συμβολοσειρών», περιλαμβανομένης τής κενής. Το σύνολο S^* είναι αριθμήσιμο, δηλαδή μπορούμε να αντιστοιχήσουμε σε κάθε ακολουθία $s_j \in S^*$ ένα φυσικόν αριθμό j. Ορίζουμε την συνάρτηση απεικονίσεως ϕ μιας ακολουθίας x σε διάνυσμα απείρου διαστάσεως, ώστε κάθε συνιστώσα του j να είναι:

$$[\phi(x)]_j = h(x, s_j)\sqrt{\lambda^{|s_j|}} \tag{3.12.7}$$

…όπου $h(x, s_j)$ είναι το πλήθος τών φορών που περιέχεται η ακολουθία s_j μέσα στην ακολουθία x, και $\lambda^{|s_j|}$ είναι ένα βάρος που εξαρτάται από το μήκος $|s_j|$ τής j-οστής ακολουθίας τού συνόλου S^*.

Άρα βάσει τών σχέσεων (3.12.6) και (3.12.7), ο πυρήνας τύπου «πασών τών υπακολουθιών» γράφεται:

$$K(x_1, x_2) = \sum_j h(x_1, s_j) h(x_2, s_j)\, \lambda^{|s_j|} \tag{3.12.8}$$

Όπως θα δούμε και αργότερα, η ιδέα τών πυρήνων είναι ότι συχνά μπορούμε να υπολογίσουμε τον πυρήνα απ΄ ευθείας και μάλιστα με αποδοτικό τρόπο, χωρίς να χρειασθεί να υπολογίσουμε τις απεικονίσεις $\phi(x_1)$ και $\phi(x_2)$ τών οποίων ο υπολογισμός είναι ενίοτε βαρύς ή ανέφικτος όπως εδώ. Πράγματι, οι Vishwanathan και Smola [12] έδειξαν πώς να υπολογισθεί ο πυρήνας τής σχέσεως (3.12.8) αποδοτικά σε χρόνο $O(|x_1| + |x_2|)$. Ο χρόνος $O(|x_1|)$ αντιστοιχεί στο κτίσιμο τού suffix tree ώστε να περιέχει την ακολουθία x_1, ο χρόνος $O(|x_2|)$ αναλίσκεται στον αλγόριθμο που περιγράφεται στο [12].

Η σημαντική παρατήρηση είναι ότι όταν φτιαχθεί τὸ suffix tree να περιέχει την ακολουθία x_1, ο ο χρόνος που απαιτείται για επομένους υπολογισμούς $K(x_1, x)$ με άλλες ακολουθίες x είναι μόνο τάξεως $O(|x|)$.

Όπως ανεφέρθη, στην παρούσα υλοποίηση έχουμε γενικευμένα suffix trees, δηλαδή μπορούμε να αποθηκεύσουμε πολλές ακολουθίες x_i σ΄ ένα δένδρο, με δυνατότητα να συνοδεύεται κάθε

μία ακολουθία από πληροφορία, όπως ένα αριθμό «βάρος» c_i. Τότε, όπως επισημαίνεται στο [12], η προηγουμένη παρατήρηση γενικεύεται και μπορούμε να υπολογίσουμε αποδοτικά όχι μόνο μία πράξη πυρήνος αλλά ένα ζυγισμένο άθροισμα πράξεων πυρήνος:

$$f(x) = \sum_i c_i K(x_i, x) \tag{3.12.9}$$

Το κόστος χρόνου για τον υπολογισμό τής (3.12.9) είναι η κατασκευή τού γενικευμένου δένδρου ἀπαξ σε χρόνο $O(\sum_i |x_i|)$, έπειτα για κάθε ακολουθία x ο υπολογισμός τής σχέσεως (3.12.9) γίνεται μόνο σε χρόνο $O(|x|)$. Η δυνατότητα αποδοτικού υπολογισμού τής σχέσεως (3.12.9) είναι πολύ σημαντική όπως θα δούμε, γιατί όλες οι μέθοδοι μηχανικής μαθήσεως που ανήκουν στην οικογένεια “kernel methods” καταλήγουν σε αυτήν την μορφή.

3.13 Η μονάδα Gramma.Kernels

Πριν εκθέσουμε τα περιεχόμενα τής μονάδος, είναι χρήσιμο να ανατρέξουμε στις μεθόδους πυρήνος και σε ποια ιδέα βασίζονται.

3.13.1 Εισαγωγικά για τις μεθόδους πυρήνος

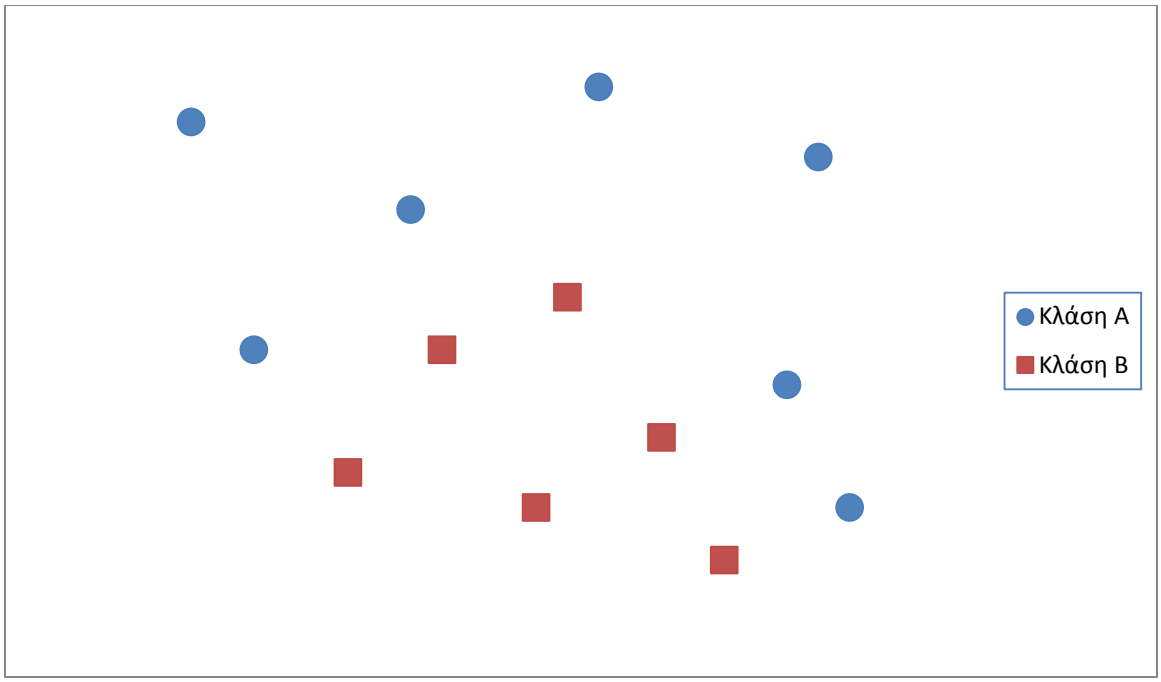
Στον κλάδο τής μηχανικής μαθήσεως υπάρχει η οικογένεια τών μεθόδων που ονομάζεται «Μέθοδοι Πυρήνος» (Kernel Methods). Αυτές οι μέθοδοι εκκινούν από γραμμικές μεθόδους όπου τα δείγματα εκπαιδεύσεως x_i και τα άγνωστα δείγματα x είναι συνήθη διανύσματα τού $\mathbb{R}^n$, τα οποία δεν εμφανίζονται ποτέ μόνα τους στους υπολογισμούς και στην τελική έκφραση τής μεθόδου αλλά πάντοτε μόνο μέσα σε μεταξύ τους εσωτερικά γινόμενα. Η τελική έκφραση τών μεθόδων αυτών έχει μία γενική μορφή:

$$f(x) = \sum_\iota c_i x_i^T x \tag{3.13.1}$$

…όπου c_i είναι συντελεστές που προκύπτουν από την «εκπαίδευση», δηλαδή την βελτιστοποίηση τού στόχου τής μεθόδου.

Για τις γραμμικές αυτές μεθόδους, η συνάρτηση f στην παραπάνω έκφραση (3.13.1) μπορεί να έχει διαφόρους ρόλους, για παράδειγμα μπορεί να έχει τον ρόλο μιας γραμμικής προβλέψεως, αν η μέθοδος επιτελεί παρεμβολή, ή να έχει ρόλο διευκρινιστού αν η μέθοδος επιτελεί δυαδική ταξινόμηση, δηλαδή αν $f(x) > 0$ τότε το δείγμα x ταξινομείται στην κλάση A αλλιώς στην κλάση B.

Όμως συχνά ένα γραμμικό μοντέλο δεν μπορεί να περιγράψει τον χώρο ενός προβλήματος. Στο παράδειγμα τού δυαδικού ταξινομητού, μπορεί να μη υπάρχει υπερεπίπεδο που να διαχωρίζει τα σημεία τής κλάσεως A από αυτά τής B, δηλαδή τά δεδομένα να μή είναι «γραμμικώς διαχωρίσιμα». Η έννοια τών μη γραμμικώς διαχωρισίμων κλάσεων δείχνεται στο Σχήμα 3-6 σε δύο διαστάσεις, όπου δεν μπορεί να βρεθεί ευθεία που να χωρίζει τα σημεία δηλωμένα με κύκλους από τα σημεία δηλωμένα με τετράγωνα.



Σχήμα 3-6: Παράδειγμα μη γραμμικώς διαχωρισίμων κλάσεων σε 2 διαστάσεις

Όταν λοιπόν δεν επαρκεί ένα γραμμικό μοντέλο για να περιγράψει τον χώρο όπου ανήκουν τα διανύσματα τών δειγμάτων x, αναζητούμε ένα μετασχηματισμό ϕ με σκοπό ώστε η εικόνα $\phi(x)$ να επιδέχεται γραμμικό μοντέλο. Αυτό επιτυγχάνεται συνήθως με μετασχηματισμούς που οδηγούν σε χώρους με περισσότερες διαστάσεις από τον αρχικό. Στο παράδειγμα τού δυαδικού ταξινομητού, ζητούμε να απεικονίσουμε τα δείγματα σε χώρο ώστε να είναι γραμμικώς διαχωρίσιμα, δηλαδή να μπορούμε να βρούμε υπερεπίπεδο που να διαχωρίζει τις δύο κλάσεις.

Ο μετασχηματισμός ϕ ανοίγει όμως και άλλες δυνατότητες. Αφού η συνάρτηση ϕ είναι αυθαίρετη και αφού αντικαθιστά όλες τις εμφανίσεις τών δειγμάτων x και x_i με $\phi(x)$ και $\phi(x_i)$ στους υπολογισμούς, μπορούμε να επεκτείνουμε το πεδίο ορισμού τής ϕ να περιλάβει και άλλα σύνολα πέραν τού $\mathbb{R}^n$. Μπορούμε να απεικονίσουμε δένδρα, γράφους εν γένει, ή οποιοδήποτε αντικείμενο σ΄ ένα χώρο διανυσμάτων, πιθανώς απειροδιάστατο. Στην προηγουμένη ενότητα γιά την μονάδα Gramma.Indexing, στην σχέση (3.12.7) είδαμε πώς ορίζουμε απεικόνιση ϕ για συμβολοσειρές ή ακολουθίες εν γένει.

Με βάση την απεικόνιση ϕ λοιπόν, η γραμμική μορφή (3.13.1) μετασχηματίζεται ως εξής:

$$f(x) = \sum_i c_i \phi(x_i)^T \phi(x) \tag{3.13.2}$$

Η ιδέα τών μεθόδων πυρήνος βασίζεται στις ακόλουθες παρατηρήσεις:

- Στους υπολογισμούς και στην τελική μορφή τής μεθόδου, μία απεικόνιση ϕ δεν εμφανίζεται ποτέ μόνη της αλλά πάντοτε σε εσωτερικό γινόμενο με μία άλλη.

- Για κάποιες πολύ σημαντικές και χρήσιμες απεικονίσεις ϕ, ο υπολογισμός τού εσωτερικού γινομένου $\phi(x_1)^T\phi(x_2)$ συχνά μπορεί να γίνει απ΄ ευθείας πολύ αποδοτικώτερα από τους χωριστούς υπολογισμούς $\phi(x_1)$, $\phi(x_2)$ και εσωτερικού γινομένου, οι οποίοι ενίοτε είναι και αδύνατοι (πχ όταν οι διαστάσεις τής απεικονίσεως είναι άπειρες).

Αφού λοιπόν στις μεθόδους αυτές δεν θα δούμε ποτέ μίαν απεικόνιση ϕ χωριστά, ορίζουμε ως «πυρήνα» που απορρέει από την απεικόνιση ϕ την έκφραση τού εσωτερικού γινομένου:

$$K(x_1, x_2) = \phi(x_1)^T\phi(x_2) \tag{3.13.3}$$

Με αυτόν τον τρόπο φεύγει από τους υπολογισμούς και από την τελική έκφραση κάθε άμεσος αναφορά στην συνάρτηση απεικονίσεως ϕ. Η σχέση (3.13.2) τότε έρχεται στην ακόλουθο μορφή, γνωστή και ως «μορφή αναπαραστάσεως» ή «representer form» (από το σχετικό θεώρημα τών χώρων Hilbert):

$$f(x) = \sum_i c_i K(x_i, x) \tag{3.13.4}$$

Το ότι φεύγει η άμεσος αναφορά στην συνάρτηση ϕ δίνει ακόμη ένα κέρδος. Μπορούμε να σκεφθούμε μία συνάρτηση $K(x_1, x_2)$ και να αποδείξουμε ότι είναι πυρήνας κάποιας απεικονίσεως ϕ, δηλαδή ότι υπάρχει ϕ ώστε $K(x_1, x_2) = \phi(x_1)^T\phi(x_2)$, χωρίς να βρούμε την ϕ. Σύμφωνα με πόρισμα τού θεωρήματος τού Mercer, ικανή και αναγκαία συνθήκη για να είναι μία συνάρτηση $K(x_1, x_2)$ πυρήνας κάποιας απεικονίσεως είναι να είναι η συνάρτηση K θετικώς ημιωρισμένη, δηλαδή, για οποιαδήποτε πληθος n δειγμάτων x_i και οποιοδήποτε $\alpha \in \mathbb{R}^n$ και να ισχύει:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j K(x_i, x_j) \geq 0 \tag{3.13.5}$$

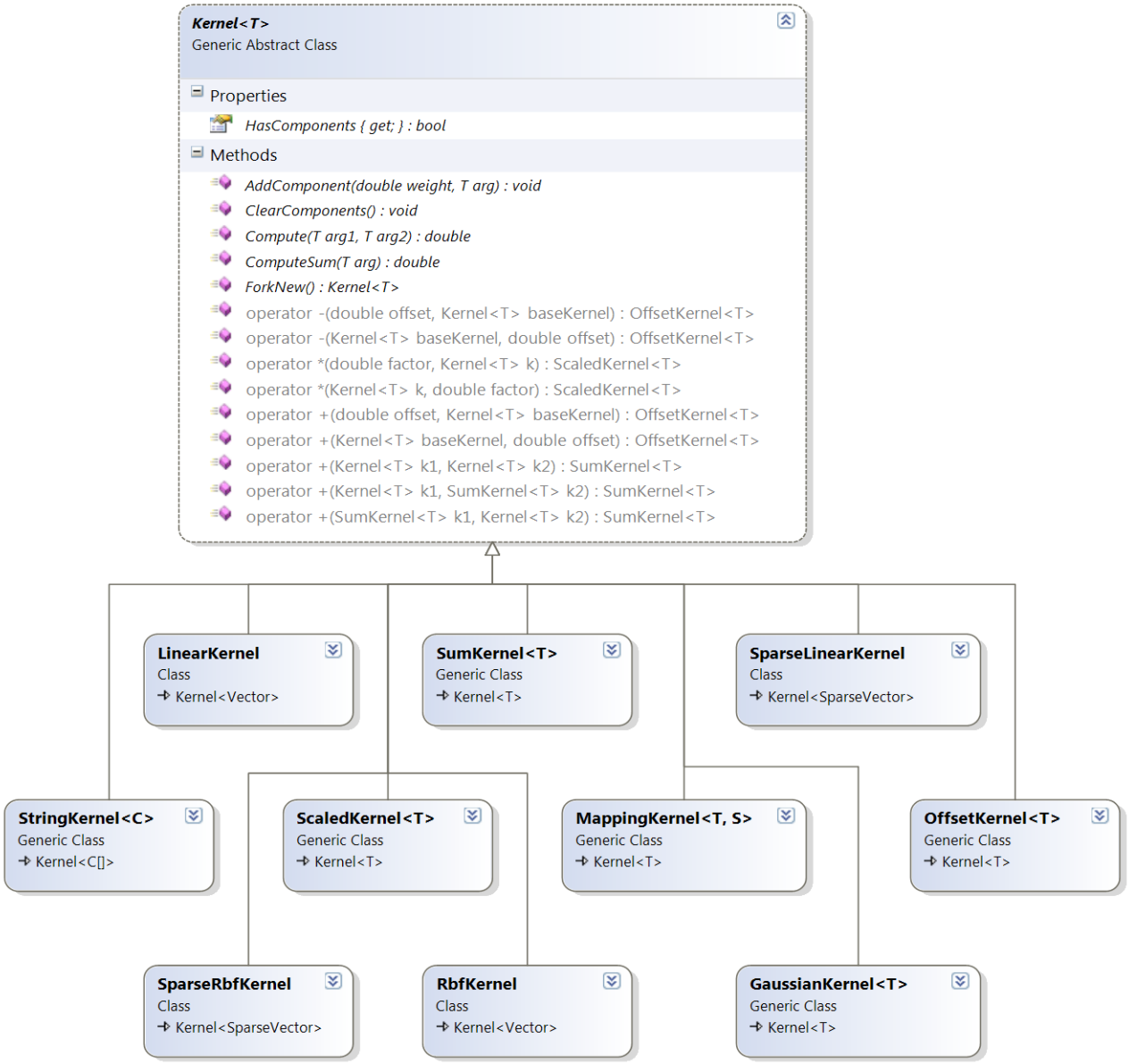
Αυτό είναι ισοδύναμο με το ότι ο ακόλουθος πίνακας (Gram matrix) είναι θετικώς ημιωρισμένος:

$$\begin{bmatrix} K(x_1,x_1) & K(x_1,x_2) & \cdots & K(x_1,x_n) \\ K(x_2,x_1) & K(x_2,x_2) & \cdots & K(x_2,x_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_n,x_1) & K(x_n,x_2) & \cdots & K(x_n,x_n) \end{bmatrix} \succcurlyeq 0 \tag{3.13.6}$$

Με το κριτήριο αυτό μπορούμε να δείξουμε ότι επιτρέπονται συνθέσεις πυρήνων οι οποίες δημιουργούν νέους πυρήνες. Αν K_1 και K_2 είναι έγκυροι πυρήνες που πληρούν το παραπάνω κριτήριο, τότε οι ακόλουθοι πυρήνες είναι επίσης έγκυροι:

$$
\begin{aligned}
K_m(x_1,x_2) &= K_1(x_1,x_2)K_2(x_1,x_2) \\
K_s(x_1,x_2) &= \gamma_1 K_1(x_1,x_2) + \gamma_2 K_2(x_1,x_2) \qquad \gamma_1,\gamma_2 \geq 0 \\
K_g(x_1,x_2) &= \exp\left(\frac{-K_1(x_1,x_1) - K_1(x_2,x_2) + 2K_1(x_1,x_2)}{2\sigma^2}\right)
\end{aligned}
\tag{3.13.7}
$$

3.13.2 Η αρχιτεκτονική τής μονάδος Gramma.Kernels



Σχήμα 3-7: Τα είδη τών πυρήνων στην μονάδα Gramma.Kernels

Η μονάδα ορίζει την σύμβαση που πρέπει να πληροί ένας πυρήνας, και υπό την σύμβαση αυτή υλοποιεί μερικούς βασικούς πυρήνες και επιτρέπει συνδυασμούς πυρήνων για την δημιουργία νέων.

Η σύμβαση που πρέπει να πληρούν όλοι οι πυρήνες εκφράζεται με την αφηρημένη κλάση Kernel<T>, η οποία εικονίζεται στο επάνω μέρος στο Σχήμα 3-7. Η παράμετρος τύπου T είναι ο τύπος τών αντικειμένων για τους οποίους ορίζεται το εσωτερικό γινόμενο. Με άλλα λόγια, είναι ο τύπος τών δύο ορισμάτων τού πυρήνος.

Μπορεί κανείς να πεί ότι θεωρητικώς χρειάζεται μόνο μία μέθοδος "Compute" στην σύμβαση, η οποία να υλοποιεί την παράσταση $K(x_1, x_2)$ δεχομένη δύο ορίσματα τύπου T ως x_1 και x_2 και επιστρέφουσα πραγματικόν αριθμό, όπως φαίνεται στο Σχήμα 3-7. Αλλά είδαμε στην σχέση (3.13.4) ότι η τελική μορφή τών μεθόδων πυρήνος είναι ένα σταθμισμένο άθροισμα υπολογισμών πυρήνος. Μπορεί βεβαίως κάποιος ν' αποτιμήσει την έκφραση με το να υπολογίσει ένα-ένα τους όρους μέσα στο άθροισμα με την μέθοδο "Compute", έπειτα να τους σταθμίσει και να τους αθροίσει. Μερικοί όμως πυρήνες έχουν μίαν επιμεριστική ιδιότητα η οποία επιτρέπει τον υπολογισμό τής εκφράσεως πολύ πιο αποδοτικά. Ένα παράδειγμα επιμεριστικής ιδιότητος βλέπουμε στον γραμμικό πυρήνα $K_l(x_1, x_2) = {x_1}^T x_2$, δηλαδή στο σύνηθες εσωτερικό γινόμενο δύο διανυσμάτων $x_1, x_2 \in \mathbb{R}^n$:

$$f(x) = \sum_i c_i K_l(x_i, x) = \sum_i c_i {x_i}^T x = \left(\sum_i c_i {x_i}^T\right) \cdot x \qquad (3.13.8)$$

Το εντός τής παρενθέσεως άθροισμα περιέχει δεδομένα που είναι σταθερά μετά την εκπαίδευση τής μεθόδου, άρα μπορεί να υπολογισθεί άπαξ, οπότε ο υπολογισμός τής $f(x)$ καταλήγει να είναι μόνο ένα εσωτερικό γινόμενο, όσο και να είναι το πλήθος τών όρων. Είδαμε στην προηγουμένη ενότητα ότι έχει επιμεριστική ιδιότητα και ο πολύ σημαντικός πυρήνας ακολουθιών. Ακόμη όμως και αν ο πυρήνας δεν έχει επιμεριστική ιδιότητα, η προτέρα γνώση τών c_i και x_i μπορεί να επιτρέψει δραματική επιτάχυνση τού υπολογισμού τής τελικής μορφής (3.13.4).

Το κάθε ζεύγος c_i και x_i θα το ονομάσουμε στο εξής «συστατικό» ή "component" που ανατίθεται στον πυρήνα. Με την μέθοδο "AddComponent" προσθέτουμε ένα συστατικό, με την μέθοδο "ClearComponents" αφαιρούμε όλα τα συστατικά, και με την ιδιότητα "HasComponents" ελέγχουμε αν υπάρχει τουλάχιστον ένα συστατικό. Με την μέθοδο "ForkNew" λαμβάνουμε πανομοιότυπο πυρήνα χωρίς όμως τα τρέχοντα συστατικά. Όταν λοιπόν ορίσουμε όλα τα συστατικά με κλήσεις τής μεθόδου "AddComponent" άπαξ, τότε με την μέθοδο "ComputeSum" μπορούμε να υπολογίσουμε την έκφραση $f(x)$ τής γενικής μορφής (3.13.4) αποδοτικώτερα καθ΄ όσον ο πυρήνας δύναται να εκμεταλλευθεί την γνώση τών συστατικών. Με τον τρόπον αυτόν οι κλάσεις Kernel<T> κρατούν όχι μόνο τον πυρήνα αλλά και την τελική λύση μιας μεθόδου πυρήνος. Γι΄ αυτόν τον λόγο οι κλάσεις αυτές είναι "serializable" ώστε να συμμετέχουν εύκολα στον μηχανισμό αποθηκεύσεως μιας εφαρμογής.

Ας δούμε τώρα τις υλοποιήσεις συγκεκριμένων πυρήνων που προσφέρονται από την μονάδα:

- **LinearKernel και SparseLinearKernel:** Είναι υλοποιήσεις τού γραμμικού πυρήνος $K_l(x_1, x_2) = {x_1}^T x_2$ για διανύσματα πυκνά (Vector) και αραιά (SparseVector) τής μονάδος Gramma.Vectors.
- **RbfKernel και SparseRbfKernel:** Είναι υλοποιήσεις τού γκαουσιανού πυρήνος $K_r(x_1, x_2) = \exp\left(-\|x_1 - x_2\|^2/(2\sigma^2)\right)$ για διανύσματα πυκνά (Vector) και αραιά (SparseVector) τής μονάδος Gramma.Vectors.
- **StringKernel<C>:** Είναι πυρήνας ακολουθιών βασισμένος στην προεκτεθείσα μονάδα Gramma.Indexing και την κλάση της KernelSuffixTree. Η παράμετρος C είναι ο τύπος τού στοιχείου ή «χαρακτήρος» τής ακολουθίας.

Πέραν αυτών τών συγκεκριμένων, η μονάδα επιτρέπει σύνθεση πυρήνων όπως στην σχέση (3.13.7) δια τών ακολούθων κλάσεων:

- **ScaledKernel<T>:** Ανακλιμακώνει ένα πυρήνα για στοιχεία τύπου T με μία θετική σταθερά c:

$$K(x_1, x_2) = cK'(x_1, x_2) \tag{3.13.9}$$

- **OffsetKernel<T>:** Προσθέτει σ' ένα πυρήνα για στοιχεία τύπου T μία θετική σταθερά γ:

$$K(x_1, x_2) = K'(x_1, x_2) + \gamma \tag{3.13.10}$$

- **SumKernel<T>:** Υλοποιεί πυρήνα από άθροισμα πυρήνων για στοιχεία τύπου T:

$$K(x_1, x_2) = \sum_j K_j(x_1, x_2) \tag{3.13.11}$$

- **GaussianKernel<T>:** Δημιουργεί πυρήνα K με την σχέση (3.13.12) για πυρήνα στοιχείων τύπου T, μετασχηματίζοντας στον γκαουσιανό χώρο τις συναρτήσεις απεικονίσεως που υπονοούνται από ένα άλλο πυρήνα K' στοιχείων τύπου T. Είναι η γενίκευση τών γκαουσιανών πυρήνων RbfKernel και SparseRbfKernel οι οποίοι είναι ισοδύναμοι αντιστοίχως με GaussianKernel<LinearKernel> και GaussianKernel<SparseLinearKernel>.

$$K(x_1, x_2) = \exp\left(\frac{-K'(x_1, x_1) - K'(x_2, x_2) + 2K'(x_1, x_2)}{2\sigma^2}\right) \tag{3.13.12}$$

- **MappingKernel<T, S>:** Δημιουργεί πυρήνα K με στοιχεία τύπου T τη βοηθεία ενός πυρήνος K' με στοιχεία τύπου S και μιας συναρτήσεως ψ μετασχηματισμού από T σε S:

$$K(x_1, x_2) = K'\left(\psi(x_1), \psi(x_2)\right) \tag{3.13.13}$$

Για να διευκολυνθεί η σύνθεση πυρήνων, έχουν ορισθεί τελεστές "+" και "*" στην κλάση Kernel<T> όπως φαίνεται στο Σχήμα 3-7, ώστε τα ScaledKernel<T>, OffsetKernel<T>, SumKernel<T> να παράγονται και με φυσική γραφή όπως στο παράδειγμα:

```
var Ks = α * K1 + β * K2; // Τα K1 & K2 είναι Kernel<T>, το Ks είναι SumKernel<T>.
```

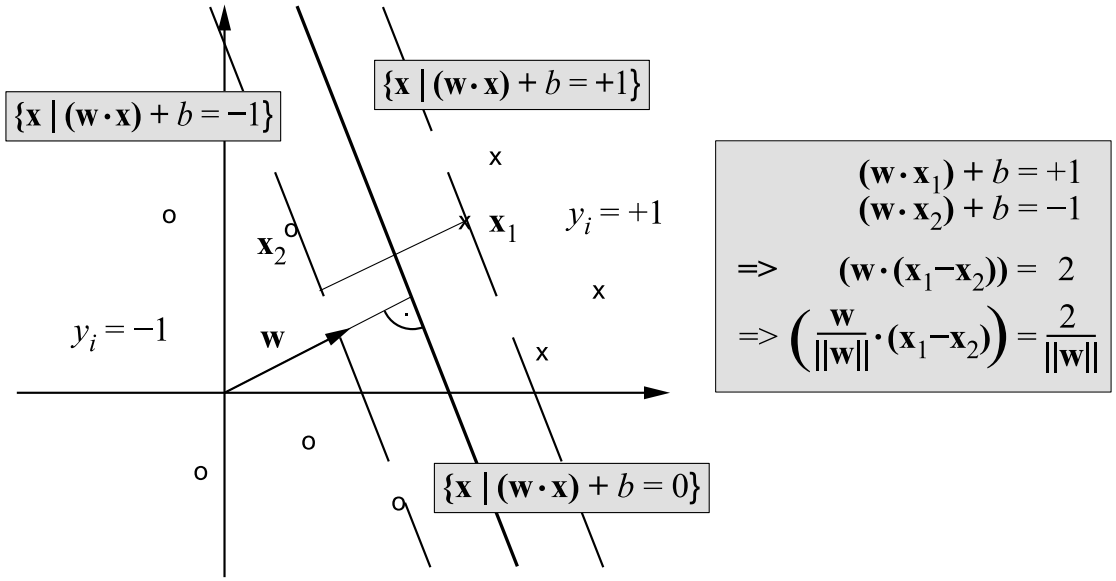
Βεβαίως, μπορεί κανείς να ορίσει νέους πυρήνες με το να υλοποιήσει την αφηρημένη κλάση Kernel<T>.

3.14 Η μονάδα Gramma.SVM

Η μονάδα αυτή υλοποιεί δυαδικούς ταξινομητές. Η μέθοδος που ακολουθεί λέγεται «Ταξινομητής Διανυσμάτων Υποστηρίξεως» ή "Support Vector Machine" [13], και ανήκει στην οικογένεια τών μεθόδων πυρήνος. Πριν παρουσιάσουμε στην δομή τής μονάδος θ' ανατρέξουμε σε λίγα εισαγωγικά για τους ταξινομητές διανυσμάτων υποστηρίξεως.

3.14.1 Ορισμός τού Ταξινομητού Διανυσμάτων Υποστηρίξεως

Αρχίζουμε την ανάπτυξη θεωρώντας ότι τα δεδομένα που θέλουμε να ταξινομήσουμε στην «θετική» κλάση C_+ ή «αρνητική» κλάση C_- είναι διανύσματα σημεία τού $\mathbb{R}^n$. Έστω κατ' αρχήν ότι τα πλήθους l εκπαιδευτικά δείγματα x_i είναι γραμμικώς διαχωρίσιμα, δηλαδή υπάρχει υπερεπίπεδο που να χωρίζει τελείως τα θετικά δείγματα σημείων $x_i \in C_+$ από τα αρνητικά $x_i \in C_-$. Τότε εν γένει υπάρχουν άπειρα τέτοια υπερεπίπεδα που να χωρίζουν τα θετικά σημεία από τα αρνητικά. Ο Ταξινομητής Διανυσμάτων Υποστηρίξεως επιλέγει το ένα τέτοιο υπερεπίπεδο που χωρίζει τις δύο κλάσεις σημείων με το μέγιστο περιθώριο.



Σχήμα 3-8: Παράδειγμα γραμμικού Ταξινομητού Διανυσμάτων Υποστηρίξεως για γραμμικώς διαχωρίσιμα σημεία δύο διαστάσεων [14]

Στο παράδειγμα δύο διαστάσεων στο Σχήμα 3-8, το διαχωριστικό υπερεπίπεδο είναι η ευθεία που μεγιστοποιεί την απόστασή της από τα σημεία κλάσεως θετικής κλάσεως "x" και αρνητικής κλάσεως "o".

Η εξίσωση διαχωριστικής ευθείας για το παράδειγμα ή εν γένει τού διαχωριστικού υπερεπιπέδου είναι:

$$w^T x + b = 0 \tag{3.14.1}$$

Το διάνυσμα w ορίζει την κάθετο στην κατεύθυνση τής ευθείας ενώ η σταθερά b ορίζει την αρνητική απόσταση τής ευθείας από την αρχή τών αξόνων κατά την κατεύθυνση και το μέτρο τού w. Δηλαδή η αλγεβρική απόσταση τής ευθείας από την αρχή τών αξόνων είναι -$b/\|w\|$ με βάση την φορά τού w.

Το υπερεπίπεδο διαιρεί τον χώρο στις δύο κλάσεις ώστε ένα τυχόν σημείο x να ταξινομείται ως εξής:

$$w^T x + b \begin{cases} > 0 & x \in C_+ \\ < 0 & x \in C_- \end{cases} \tag{3.14.2}$$

Η απόσταση γ_+ τού υπερεπιπέδου (3.14.1) από τα κοντινότερα σημεία $x_i \in C_+$ και η αντίστοιχη απόσταση γ_- από τά κοντινότερα σημεία $x_i \in C_-$ είναι:

$$\begin{aligned} \gamma_+ &= \min_{x_i \in C_+} \frac{|w^T x_i + b|}{\|w\|} = \min_{x_i \in C_+} \frac{(w^T x_i + b)}{\|w\|} \\ \gamma_- &= \min_{x_i \in C_-} \frac{|w^T x_i + b|}{\|w\|} = \min_{x_i \in C_-} \frac{-(w^T x_i + b)}{\|w\|} \end{aligned} \tag{3.14.3}$$

Τα σημεία δειγμάτων x_i εκείνα τα οποία δίνουν τις ανωτέρω ελάχιστες αποστάσεις γ_+ και γ_- ονομάζονται «διανύσματα υποστηρίξεως». Όλα τα υπόλοιπα διανύσματα έχουν μεγαλύτερες αποστάσεις από το διαχωριστικό υπερεπίπεδο, συνεπώς ισχύει:

$$\frac{w^T x_i + b}{\|w\|} \begin{cases} \geq \gamma_+ & x_i \in C_+ \\ \leq -\gamma_- & x_i \in C_- \end{cases} \tag{3.14.4}$$

Θέλουμε το διαχωριστικό υπερεπίπεδο να είναι ακριβώς στην μέση μεταξύ τών δειγμάτων τών δύο κλάσεων, άρα θα ισχύει $\gamma_+ = \gamma_-$. Επίσης παρατηρούμε ότι η εξίσωση (3.14.1) τού υπερεπιπέδου μπορεί να πολλαπλασιασθεί κατά μίαν αυθαίρετο σταθερά λ ώστε οι παράμετροι $w' = \lambda w$ και $b' = \lambda b$ να ορίζουν το ίδιο υπερεπίπεδο με τις παραμέτρους w και b. Από τις άπειρες δυνατές κλιμακώσεις στο εξής θα υπονοούμε μία κλιμάκωση ώστε να ισχύει:

$$\|w\|\gamma_+ = \|w\|\gamma_- = 1 \tag{3.14.5}$$

Με την παραπάνω συνθήκη κλιμακώσεως η σχέση (3.14.4) γίνεται:

$$w^T x_i + b \begin{cases} \geq 1 & x_i \in C_+ \\ \leq -1 & x_i \in C_- \end{cases} \tag{3.14.6}$$

Αν ορίσουμε $y_i = 1$ αν $x_i \in C_+$ και $y_i = -1$ αν $x_i \in C_-$ τότε η προηγουμένη σχέση συμπυκνώνεται:

$$(w^T x_i + b) y_i \geq 1 \tag{3.14.7}$$

Συνεπώς, ο σκοπός μας είναι να μεγιστοποιήσουμε υπό τους περιορισμούς (3.14.7) το περιθώριο $\gamma = \gamma_+ + \gamma_-$, όπως ορίζεται από σχέση (3.14.5):

$$\begin{aligned} \min_{w,b} \quad & \gamma = \frac{2}{\|w\|} \\ \text{υπό} \quad & (w^{\,T} x_i + b) y_i \geq 1 \end{aligned} \tag{3.14.8}$$

Ή ισοδύναμα, έχουμε το πρόβλημα βελτιστοποιήσεως τής συναρτήσεως J υπό περιορισμούς:

$$\begin{aligned} \min_{w,b} \quad & J(w,b) = \frac{1}{2}\|w\|^2 \\ \text{υπό} \quad & (w^{\,T} x_i + b) y_i \geq 1 \end{aligned} \tag{3.14.9}$$

Μπορεί στην πράξη όμως κάποια εκπαιδευτικά σημεία να βρίσκονται αρκετά εκτός τής κλάσεώς τους κοντά στην περιοχή τής άλλης κλάσεως, τα λεγόμενα “outliers”. Ίσως μάλιστα να εισέρχονται στην περιοχή τής άλλης κλάσεως ώστε τα σημεία να μη είναι πλέον γραμμικώς διαχωρίσιμα και να μη μπορεί πλέον να ισχύει ο περιορισμός (3.14.7) για αυτά τα “outliers”. Για να μπορέσουμε να συγχωρήσουμε την υπέρβαση αυτών τών σημείων σε περιοχή τής αντιθέτου κλάσεως, εισάγουμε τις λεγόμενες «μεταβλητές χαλαρότητος» $\xi_i \geq 0$ και μεταβάλλουμε τους περιορισμούς (3.14.7):

$$\begin{aligned} & (w^{\,T} x_i + b) y_i \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{3.14.10}$$

Προφανώς από την σχέση (3.14.10) βλέπουμε ότι αν $\xi_i > 1$, το εκπαιδευτικό δείγμα x_i θα ταξινομείται σε λάθος κλάση. Άρα η έκφραση $\sum_i \xi_i$ είναι ένα άνω φράγμα τών δειγμάτων που θα ταξινομούνται λάθος μετά την εκπαίδευση. Συνεπώς μπορούμε να μεταβάλουμε την συνάρτηση στόχου J στο αρχικό πρόβλημα βελτιστοποιήσεως προσθέτοντας αυτήν την έκφραση με ένα παράγοντα ποινής C για την παράβαση τού περιθωρίου και ενημερώνοντας τούς περιορισμούς:

$$\begin{aligned} \min_{w,b,\xi} \quad & J(w,b,\xi) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l} \xi_i \\ \text{υπό} \quad & (w^{\,T} x_i + b) y_i \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{3.14.11}$$

Το παραπάνω πρόβλημα είναι κυρτό, δηλαδή έχει κυρτή συνάρτηση στόχου J και έχει περιορισμούς που μπορούν να εκφραστούν ως συνθήκες με κυρτές συναρτήσεις μικρότερες ή ίσες τού μηδενός. Παρατηρούμε επίσης ότι πάντοτε μπορούμε να σκεφθούμε τιμές τών w, b, ξ ώστε οι ανισότητες τών περιορισμών να ισχύουν αυστηρώς, δηλαδή χωρίς το ίσον. Τα παραπάνω αποτελούν την συνθήκη τού Slater ώστε το δυαδικό κατά Lagrange πρόβλημα να έχει ακριβώς την ίδια βέλτιστη λύση με το αρχικό πρόββλημα. Το δυαδικό πρόβλημα καταλήγει να είναι το εξής:

$$\begin{aligned} \min_{\alpha} \quad & W(\alpha) = \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \\ \text{υπό} \quad & 0 \le \alpha_i \le C \\ & y^T\alpha = 0 \end{aligned} \tag{3.14.12}$$

…όπου α είναι διάνυσμα l διαστάσεων που περιέχει τις δυαδικές μεταβλητές α_i ως προς τις οποίες βελτιστοποιούμε την δυαδική συνάρτηση στόχου W, e είναι διάνυσμα l διαστάσεων με όλα τα στοιχεία ίσα με 1, και Q είναι πίνακας διαστάσεων $l \times l$ με τα εξής στοιχεία:

$$Q_{ij} = y_i y_j {x_i}^T x_j \tag{3.14.13}$$

Η διευκρινιστική συνάρτηση $f(x)$ τού διαχωριστικού υπερεπιπέδου τής σχέσεως (3.14.1) ξαναγράφεται ως προς τις δυαδικές μεταβλητές α που προκύπτουν από την λύση τού πραβλήματος (3.14.12) ως εξής:

$$f(x) = w^T x + b = \sum_{i=1}^{l} \alpha_i y_i {x_i}^T x + b \tag{3.14.14}$$

Το δε b δίνεται από το ακόλουθο:

$$b = \frac{1}{|S|}\sum_{i \in S}\left(y_i - \sum_{j=1}^{l} \alpha_j y_j {x_i}^T x_j\right) \tag{3.14.15}$$
$$S = \{i : \alpha_i \neq 0\}$$

Το σύνολο S είναι ακριβώς οι δείκτες τών δειγμάτων που αποτελούν τα διανύσματα υποστηρίξεως, καθώς οι υπόλοιποι δεν υπεισέρχονται στον ορισμό τού παραπάνω υπερεπιπέδου.

Το δυαδικό πρόβλημα βελτιστοποιήσεως (3.14.12) είναι και αυτό κυρτό γιατί εύκολα μπορεί να δείξει κανείς ότι ο πίνακας Q είναι θετικώς ημιωρισμένος. Συνεπώς μπορεί να λυθεί με καθιερωμένες γενικές μεθόδους κυρτής βελτιστοποιήσεως. Επειδή όμως για μεγάλα προβλήματα ο πίνακας Q δεν χωρεί εύκολα στην μνήμη, και επειδή μπορεί να γίνουν εξειδικευμένες επιταχύνσεις για την περίπτωση τών Ταξινομητών Διανυσμάτων Υποστηρίξεως, έχουν προταθεί μέθοδοι κατατμήσεως τού προβλήματος ώστε η λύση να βρίσκεται από ακολουθία υποπροβλημάτων στα οποία βελτιστοποιείται μέρος μόνον τών δυαδικών μεταβλητών α_i ενώ οι υπόλοιπες θεωρούνται σταθερές [15, 16]. Στο άκρο αυτής τής λογικής, επιλέγουμε ώστε κάθε ακολουθία υποπροβλημάτων να βελτιστοποιεί μόνο δύο δυαδικές μεταβλητές, τον ελάχιστο δυνατό αριθμό αυτών [17]. Δεν θα μπορούσε να βελτιστοποιεί μόνο μία γιατί η συνθήκη ισότητος στην σχέση (3.14.12) θα παραβιαζόταν. Το υποπρόβλημα τών δύο μεταβλητών λύνεται αναλυτικά, δεν χρειάζεται συνήθη αναδρομικό αλγόριθμο βελτιστοποιήσεως. Η μέθοδος που προκύπτει είναι πολύ αποδοτική και λέγεται Sequential Minimal Optimization (SMO).

Από τις σχέσεις (3.14.13), (3.14.14), (3.14.15) που περιγράφουν το δυαδικό πρόβλημα και την τελική μορφή, μπορούμε να συναγάγουμε γιατί ο Ταξινομητής Διανυσμάτων Υποστηρίξεως είναι μέθοδος πυρήνος: Κάθε μία έκφραση περιέχει παραστάσεις τών δειγμάτων μόνο υπό μορφήν εσωτερικών γινομένων. Μπορούμε λοιπόν να μεταβάλουμε τώρα τον ορισμό μας, λέγοντας ότι τα δείγματα x δεν ανήκουν πλέον αναγκαστικά στον χώρο $\mathbb{R}^n$ αλλά σε οποιοδήποτε σύνολο με μία συνάρτηση $\phi(x)$ που να απεικονίζει τα δείγματα σε διανύσματα εφοδιασμένα με εσωτερικό γινόμενο. Άρα στις παραπάνω παραστάσεις αντικαθιστούμε παντού τα εσωτερικά γινόμενα ${x_i}^T x_j$ με $\phi(x_i) \cdot \phi(x_j) = K(x_i, x_j)$.

Ο πίνακας Q τού δυαδικού προβλήματος γίνεται:

$$Q_{ij} = y_i y_j K(x_i, x_j) \tag{3.14.16}$$

Η διευκρινιστική συνάρτηση λοιπόν γίνεται:

$$f(x) = w^T \phi(x) + b = \sum_{i=1}^{l} \alpha_i y_i K(x_i, x) + b \tag{3.14.17}$$

Το δε b γίνεται:

$$b = \frac{1}{|S|} \sum_{i \in S} \left(y_i - \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) \right) \tag{3.14.18}$$
$$S = \{i : \alpha_i \neq 0\}$$

Παρά ταύτα, η παρούσα εργασία δεν προχώρησε σε λύση κατά τα ανωτέρω. Ο γράφων παρουσίασε μία μέθοδο στο [18] ώστε να απλοποιηθεί η διατύπωση τού προβλήματος και να απλοποιηθούν συνάμα και να επιταχυνθούν οι αλγόριθμοι. Ο στόχος τής απλοποιήσεως είναι να απαλειφθεί ο περιορισμός ισότητος $y^T \alpha = 0$ από το δυαδικό πρόβλημα (3.14.12). Χωρίς τον περιορισμό ισότητος, μπορούμε να εφαρμόσουμε απλούστερη γενική μέθοδο κυρτής βελτιστοποιήσεως (πχ “Interior-Point”), με περιορισμούς μόνο ανισότητες. Αλλά επωφελούνται και οι ειδικές μέθοδοι επιλύσεως. Έτσι, η κατάτμηση σε υποπροβλήματα μπορεί να φθάσει να αποτελείται από υποπροβλήματα τής μιας μεταβλητής. Αυτό έχει πολλά πλεονεκτήματα. Πρώτον, κάνει τον κώδικα πολύ απλό και, επειδή αυτός ο κώδικας εκτελείται σε κρισίμους βρόχους, αυτό έχει ευεργετική επίπτωση στην επίδοση. Δεύτερον, επιτρέπει καλύτερες στρατηγικές για ταχύτερη σύγκλιση. Τρίτον, επιτρέπει σχήματα παραλληλισμού ενώ πρίν είχαμε μέθοδο σειριακή, “Sequential Minimal Optimization”.

Για την επίτευξη τών ανωτέρω, μεταβάλλουμε την συνάρτηση στόχου J τού αρχικού προβλήματος βελτιστοποιήσεως σε $\hat{J}$:

$$\hat{J}(w, b, \xi) = J(w, b, \xi) + \frac{1}{2} b^2 \tag{3.14.19}$$

Εν πρώτοις, μπορεί να πεί κάποιος ότι αυτό είναι ένα άλλο πρόβλημα. Όμως οι Mangasarian και Musicant [19] έδειξαν ότι η λύση δεν αλλάζει σχεδόν σε όλες τις περιπτώσεις. Για τις περιπτώσεις που διαφέρουν, η λύση απλώς αντιστοιχεί σε ελαφρώς διαφορετική τιμή τού παράγοντος ποινής C για την παράβαση περιθωρίου. Ο παράγων όμως C είναι αυθαίρετος και επιλέγεται με αποτίμηση (cross-validation) ώστε να λάβουμε την καλύτερη δυνατή απόδοση τού ταξινομητού, άρα δεν έχει σημασία αυτή η διαφορά.

Για να κάνουμε πιο συμπαγείς τις επόμενες παραστάσεις, επεκτείνουμε το διάνυσμα w ώστε να περιλάβει τον όρο b:

$$\widehat{w} = \langle b \quad w \rangle \tag{3.14.20}$$

Επεκτείνουμε επίσης την συνάρτηση απεικονίσεως ϕ:

$$\hat{\phi}(x) = \langle 1 \quad \phi(x) \rangle \tag{3.14.21}$$

Με τις παραπάνω αντικαταστάσεις, η διευκρινιστική συνάρτηση $f(x)$ τού υπερεπιπέδου γίνεται:

$$f(x) = \widehat{w}^T \hat{\phi}(x) \tag{3.14.22}$$

Το πρόβλημα βελτιστοποιήσεως γίνεται:

$$\begin{aligned} \min_{\widehat{w},\xi} \quad & J(\widehat{w},\xi) = \frac{1}{2}\|\widehat{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{υπό} \quad & y_i \widehat{w}^T \hat{\phi}(x) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{3.14.23}$$

Το τροποποιημένο δυαδικό πρόβλημα που προκύπτει κατά Lagrange καταλήγει να είναι πράγματι σχεδόν όμοιο με το προηγούμενο δυαδικό πρόβλημα (3.14.12), χωρίς όμως τον περιορισμό ισότητος $y^T\alpha = 0$:

$$\begin{aligned} \min_{\alpha} \quad & \widehat{W}(\alpha) = \frac{1}{2}\alpha^T \hat{Q}\alpha - e^T\alpha \\ \text{υπό} \quad & 0 \leq \alpha_i \leq C \end{aligned} \tag{3.14.24}$$

...όπου ο πίνακας $\hat{Q}$ τού τροποποιημένου δυαδικού προβλήματος είναι:

$$\hat{Q}_{ij} = y_i y_j \widehat{K}(x_i, x_j) \tag{3.14.25}$$

και όπου $\widehat{K}(x_i, x_j)$ είναι ο πυρήνας που προκύπτει από την επέκταση $\hat{\phi}$ τής συναρτήσεως απεικονίσεως:

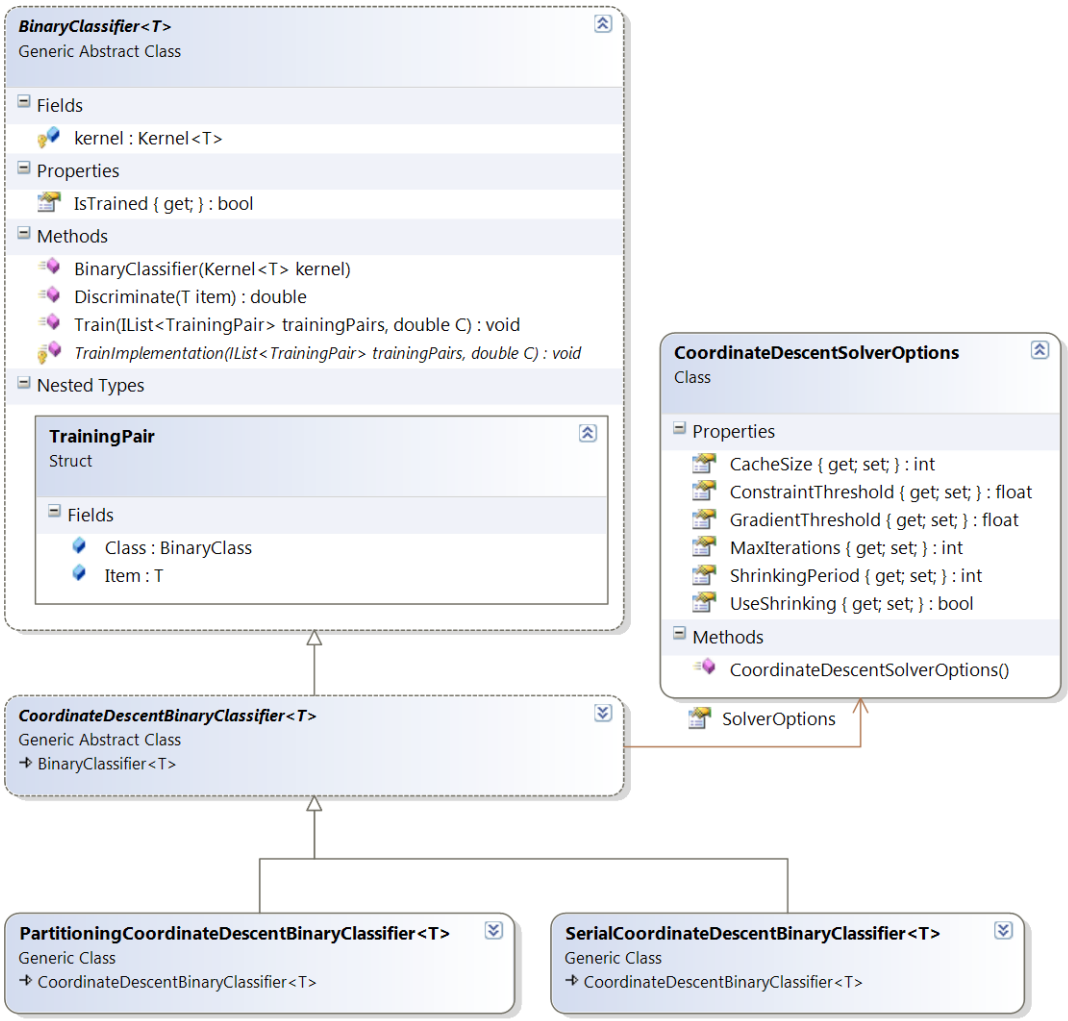
$$\widehat{K}(x_i, x_j) = \hat{\phi}(x_i)^T \hat{\phi}(x_j) = K(x_i, x_j) + 1 \tag{3.14.26}$$

Η διευκρινιστική συνάρτηση $f(x)$ τού διαχωριστικού υπερεπιπέδου γράφεται σε σχέση με τον νέο πυρήνα $\widehat{K}$ και την λύση τού τροποποιημένου δυαδικού προβλήματος (3.14.23) ως εξής:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i \widehat{K}(x_i, x) \tag{3.14.27}$$

Βλέπουμε ότι απαλλασσόμαστε και από την ανάγκη να υπολογίσουμε την σταθερά b. Παρατηρούμε επίσης ότι η παραπάνω έκφραση για την διευκρινιστική συνάρτηση f έχει απ΄ ευθείας την «μορφή αναπαραστάσεως» (3.13.4) στην οποία καταλήγουν όλες οι μέθοδοι πυρήνος.

3.14.2 Αρχιτεκτονική μονάδος Gramma.SVM



Σχήμα 3-9: Η βασική σύμβαση ενός δυαδικού ταξινομητού και δύο ενδεικτικές υλοποιήσεις της στην μονάδα Gramma.SVM

Στο Σχήμα 3-9 εικονίζεται η βασική δομή τής μονάδος Gramma.SVM και κάποιες ενδεικτικές κλάσεις της. Οι δυαδικοί ταξινομητές πρέπει να υλοποιούν την σύμβαση BinaryClassifier<T>. Η παράμετρος T είναι ο τύπος τών αντικειμένων που ταξινομούνται.

Για να δημιουργηθεί ένας ταξινομητής για αντικείμενα τύπου T χρειάζεται να παρασχεθεί ένας πυρήνας γι΄ αυτόν τον τύπο αντικειμένων, δηλαδή μία υλοποίηση τής κλάσεως Kernel<T> τής μονάδος Gramma.Kernels. Η εκπαίδευση γίνεται με την μέθοδο “Train”, με ορίσματα μία συλλογή εκπαιδευτικών ζευγών, τα οποία αποτελούνται από ένα δείγμα και την σωστή του ταξινόμηση, και την σταθερά ποινής C για την παράβαση περιθωρίου, την οποία είδαμε προηγουμένως. Έπειτα, κάθε εκπαιδευμένος ταξινομητής μπορεί ν΄ αποθηκευθεί μέσω .NET serialization.

Η ταξινόμηση ενός δείγματος γίνεται με την μέθοδο “Discriminate”, η οποία δέχεται ένα όρισμα τύπου T. Η μέθοδος δεν επιστρέφει δυαδικό αποτέλεσμα αλλά την ακατέργαστο τιμή τύπου double όπως δίνεται από την διευκρινιστική συνάρτηση f στην σχέση (3.14.27). Ο λόγος είναι ότι η τιμή αυτή μπορεί να χρησιμοποιηθεί ώστε να δοθεί και ένα μέτρο πιθανότητος για κάθε ταξινόμηση [20]. Η παρούσα εργασία θα επεικτείνει την ιδέα αυτή, όπως θα δούμε.

Στο Σχήμα 3-9 φαίνονται και δύο εκ τών υλοποιήσεων ταξινομητού που δόθηκαν από τον γράφοντα στο [18]. Και οι δύο ανήκουν στην μεθοδολογία κατατμήσεως τού προβλήματος SVM σε υποπροβλήματα τής μιας μεταβλητής, καθώς αυτή ανεφέρθη προηγουμένως. Η μεθοδολογία ονομάζεται Κατάβαση Συντεταγμένης ή Coordinate Descent, και οι υλοποιήσεις της στην μονάδα Gramma.SVM κατάγονται από την αφηρημένη κλάση CoordinateDescentBinaryClassifier<T>, η οποία κρατεί στην ιδιότητα “SolverOptions” κοινές ρυθμίσεις για τους αλγορίθμους που ανήκουν στην μεθοδολογία. Η υλοποίηση SerialCoordinateDescentBinaryClassifier<T> εκτελεί σειριακό αλγόριθμο, ενώ η PartitioningCoordinateDescentBinaryClassifier<T> διαμοιράζει το εύρος τών εκπαιδευτικών δειγμάτων σε παράλληλο επεξεργασία.

3.15 Μονάδα Gramma.CRF

Σε αυτήν την μονάδα υλοποιείται ένα είδος Πιθανοτικών Γραφικών Μαρκοβιανών Μοντέλων ή Δικτύων που λέγεται Conditional Random Fields [21], και ειδικώτερα μία περίπτωση αυτών γνωστή ως Linear Chain Conditional Random Fields. Στην συνέχεια θα δούμε μερικές βασικές έννοιες, έπειτα θα γίνει παρουσίαση τής υλοποιήσεως.

3.15.1 Γενικά Πιθανοτικά Δίκτυα

Γενικά, τα Πιθανοτικά Γραφικά Μοντέλα ή Δίκτυα (Probabilistic Graphical Models) αναπαριστούν μία από κοινού έκφραση πιθανότητος $p(y; w)$ για ένα πλήθος μεταβλητών $y = (y_1, y_2, \dots, y_n)$, όπου $w = (w_1, w_2, \dots, w_d)$ είναι d παράμετροι τού μοντέλου. Οι μεταβλητές y_i αντιστοιχίζονται στις κορυφές ενός γράφου τού οποίου οι ακμές δηλούν κάποια άμεσο εξάρτηση μεταξύ τών μεταβλητών.

Όταν ένα Πιθανοτικό Δίκτυο έχει σχηματισθεί ή εκπαιδευτεί, τότε μπορεί με διάφορες μεθόδους να παράσχει όχι μόνο την από κοινού πιθανότητα $p(y; w)$ όλων τών μεταβλητών αλλά και άλλες παράγωγες πιθανότητες τών μεταβλητών του, όπως περιθώριες ή υπό συνθήκην. Μπορούμε δηλαδή να ζητήσουμε την πιθανότητα να έχουν κάποιες εκ τών μεταβλητών μία τιμή, προαιρετικώς υπό συνθήκην ότι κάποιες άλλες μεταβλητές έχουν μία

δεδομένη τιμή. Δεν υπάρχει περιορισμός στην εκλογή τών μεταβλητών για τους παραπάνω ρόλους. Έτσι, ένα Πιθανοτικό Δίκτυο μπορεί να τελέσει χρέη ταξινομητού εννοιολογικώς, αν θεωρήσουμε κάποιες εκ τών μεταβλητών y ως «μεταβλητές εξόδου» και ζητήσουμε την πιθανότητα νά έχουν κάποια τιμή υπό συνθήκην ότι κάποιες «μεταβλητές εισόδου» έχουν μία δοσμένη τιμή. Τότε, ο πιθανότερος συνδυασμός τιμών για τις «μεταβλητές εξόδου» είναι η «κλάση» που αντιστοιχεί στις «μεταβλητές εισόδου». Αυτή η εκτίμηση τών τιμών τών μεταβλητών είναι γνωστή ως “Maximum a Posteriori (MAP)”.

Ανακύπτει όμως ένα πρακτικό ζήτημα όταν κινούμαστε μέ την έννοια τού ταξινομητού. Το Πιθανοτικό Δίκτυο, όπως είπαμε, απεικονίζει την πλήρη από κοινού πιθανότητα $p(y; w)$ τών μεταβλητών και μπορεί να δώσει οποιανδήποτε παράγωγο πιθανότητα αυτών. Δηλαδή στην περίπτωση όπου μιλάμε με όρους ταξινομητού, μπορεί ανά πάσα στιγμή να αποδώσει τον ρόλο «μεταβλητή εισόδου» και «μεταβλητή εξόδου» σε οποιεσδήποτε μεταβλητές. Τα παραπάνω συνεπάγονται ότι, για ένα ακριβές και αποδοτικό μοντέλο, πρέπει να υπάρχει πλήρης μοντελοποίση όλων τών αλληλοεξαρτήσεων όλων τών μεταβλητών. Αυτό σημαίνει ότι, όταν δουλεύουμε το δίκτυο με την έννοια ενός ταξινομητού, πρέπει να μοντελοποιηθούν όχι μόνο οι αλληλοεξαρτήσεις τών «μεταβλητών εξόδου» αλλά και τών «μεταβλητών εισόδου» όπως και αυτές μεταξύ τών δύο ομάδων μεταβλητών. Πρέπει να καταβληθεί μεγάλη προσπάθεια σχεδιασμού καί υπολογιστικής ισχύος, τόσο στην εκπαίδευση όσο και στην λειτουργία τού δικτύου, ώστε να μοντελοποιηθούν οι αλληλοεξαρτήσεις τών «μεταβλητών εισόδου»,ενίοτε πολύπλοκες, που για το έργο τής ταξινομήσεως κανονικά δεν θα έπρεπε να μας ενδιαφέρουν, γιατί οι τιμές τών «μεταβλητών εισόδου» είναι πάντοτε δεδομένες και ποτέ δεν ζητείται κάποια πιθανότητα γι’ αυτές. Παραβαίνεται δηλαδή ο κανόνας τής Υπολογιστικής Μαθήσεως «μή μάθεις κάτι που δέν χρειάζεται νά μάθεις».

Αν απλοποιήσουμε το μοντέλο θεωρώντας ότι οι «μεταβλητές εισόδου» είναι ανεξάρτητες ενώ δεν είναι, αυτό φθείρει την ακρίβεια τού μοντέλου. Παράδειγμα τέτοιας απλοποιήσεως και μειώσεως ακριβείας είναι το γνωστό δίκτυο “Naïve Bayes”, το οποίο οφείλει τον χαρακτηρισμό του ως «αφελούς» στην υπεραπλουστευμένη αυτή παραδοχή ότι οι «μεταβλητές εισόδου» είναι ανεξάρτητες.

3.15.2 Conditional Random Fields

Για να αρθεί το παραπάνω πρόβλημα τής ανάγκης ορισμού, εκπαιδεύσεως και υπολογισμού τών εξαρτήσεων τών «μεταβλητών εισόδου» όταν θέλουμε ένα ταξινομητή, με τα ειδικά Πιθανοτικά Δίκτυα που ονομάζονται Conditional Random Fields χωρίζουμε εξ’ αρχής τίς μεταβλητές σέ «εισόδου» x και «εξόδου» y, και μοντελοποιούμε όχι την από κοινού πιθανότητά τους $p(x, y; w)$, αλλά την πιθανότητα τών «εξόδων» δεδομένων τών εισόδων, δηλαδή $p(y|x; w)$. Από αυτό το μοντέλο δεν μπορούμε να εξαγάγουμε καμμία πιθανότητα για τις τιμές τών «μεταβλητών εισόδου» x, αλλά όπως ανεφέρθη, δεν την χρειαζόμαστε στην έννοια ενός ταξινομητού, αφού οι τιμές τών x είναι πάντοτε δοσμένες. Κερδίζουμε όμως σε σχέση με τα γενικά Πιθανοτικά Δίκτυα ότι δέν οφείλουμε να ορίζουμε, να εκπαιδεύουμε και νά υπολογίζουμε τις αλληλοεξαρτήσεις τών «μεταβλητών εισόδου» x.

Θα επικεντρωθούμε στα Conditional Random Fields τών οποίων το μοντέλο πιθανότητος είναι Λογαριθμικώς Γραμμικό ή Log-Linear:

$$p(y|x;w) = \frac{\exp w^T F(x,y)}{Z(x,w)} \tag{3.15.1}$$

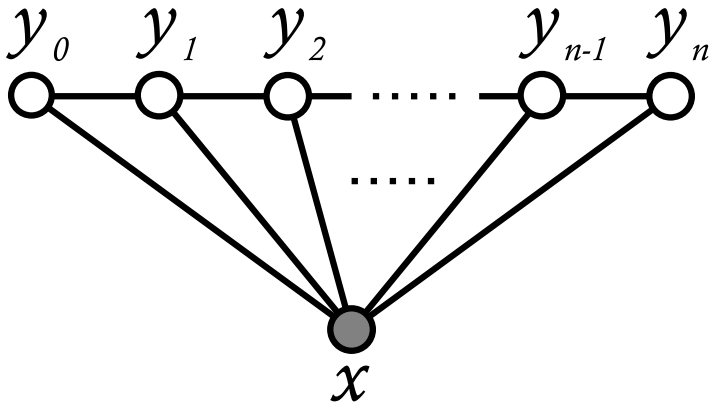
Στην παραπάνω σχέση, η συνάρτηση F επιστρέφει διάνυσμα διαστάσεως d και μπορεί να ειδωθεί ως συσκευασία βαθμωτών «συναρτήσεων χαραρακτηριστικών» ή “feature functions”. Οι «μεταβλητές εξόδου» y λαμβάνουν τιμές από ένα πεδίο ορισμού Y^p, και ο παράγων Z κανονικοποιεί την έκφραση (3.15.1) ώστε το άθροισμα τών τιμών της για όλα τά $y \in Y^p$ να ισούται μέ ένα:

$$Z(x,w) = \sum_{y \in Y^p} \exp w^T F(x,y) \tag{3.15.2}$$

3.15.3 Linear Chain Conditional Random Fields

Η σχέση (3.15.1) είναι γενική, και υπό την μορφήν αυτή μπορούν να εκφρασθούν όλες οι πιθανοτικές εξαρτήσεις μεταξύ τών «μεταβλητών εξόδου» y, δηλαδή όλες οι δυνατές συνδέσεις τών μεταβλητών y στο δίκτυο. Οι υπολογισμοί όμως και τής εκπαιδεύσεως και τών αποτιμήσεων τού δικτύου είναι ανάλογοι τού γινομένου τών πιθανών τιμών τών μεταβλητών που συμμετέχουν σε μία μεγίστη κλίκα τού δικτύου. Τουτέστιν, όσο πιο πυκνές συνδέσεις έχει το δίκτυο, το υπολογιστικό κόστος αυξάνεται εκθετικά.

Για την περίπτωση που οι μεταβλητές εξόδου παριστάνουν ακολουθία, όπως σήμανση λέξεων σε πρόταση, η σύνδεση τών μεταβλητών μπορεί να είναι σε ευθεία, ώστε κάθε μία να συνδέεται με την προηγουμένη και την επομένη της, όπως φαίνεται στο Σχήμα 3-10.



Σχήμα 3-10: Γράφημα ενός δικτύου Linear Chain Conditional Random Field

Αυτή είναι η διάταξη τού Linear Chain Conditional Random Field. Όπως φαίνεται και στο σχήμα, οι κλίκες τών μεταβλητών y έχουν μέγεθος 2. Αν n είναι το μήκος τής ακολουθίας και κάθε μεταβλητή y_i παίρνει τιμές από ένα σύνολο Y χωρίς περιορισμό, τότε το κόστος μιάς αποτιμήσεως τού δικτύου είναι τάξεως $O(n|Y|^2)$. Ο τετραγωνικός εκθέτης απεικονίζει το

μέγεθος τών κλικών. Όμως και κατά την εκπαίδευση λαμβάνει χώρα αποτίμηση γιά κάθε δείγμα, άρα σε κάθε περίπτωση το κόστος είναι ανάλογο τού τετραγώνου τού πλήθους τών δυνατών τιμών μιας μεταβλητής y_i.

Συνεπώς, όπως μας υποδεικνεύει και το Σχήμα 3-10, οι παράγοντες που διαμορφώνουν τον αριθμητή τής σχέσεως (3.15.1) πρέπει κατά τις μεταβλητές εξόδου y να εξαρτώνται μόνο από γειτονικές y_{i-1} και y_i. Αυτό σημαίνει ότι η συνάρτηση διανύσματος χαρακτηριστικών $F(x, y)$ αναλύεται σε άθροισμα συναρτήσεων $f(x, y_{i-1}, y_i, i)$, καθώς μέσα στο εκθετικό το γινόμενο γίνεται άθροισμα:

$$F(x, y) = \sum_{i=1}^{n} f(x, y_{i-1}, y_i, i) \tag{3.15.3}$$

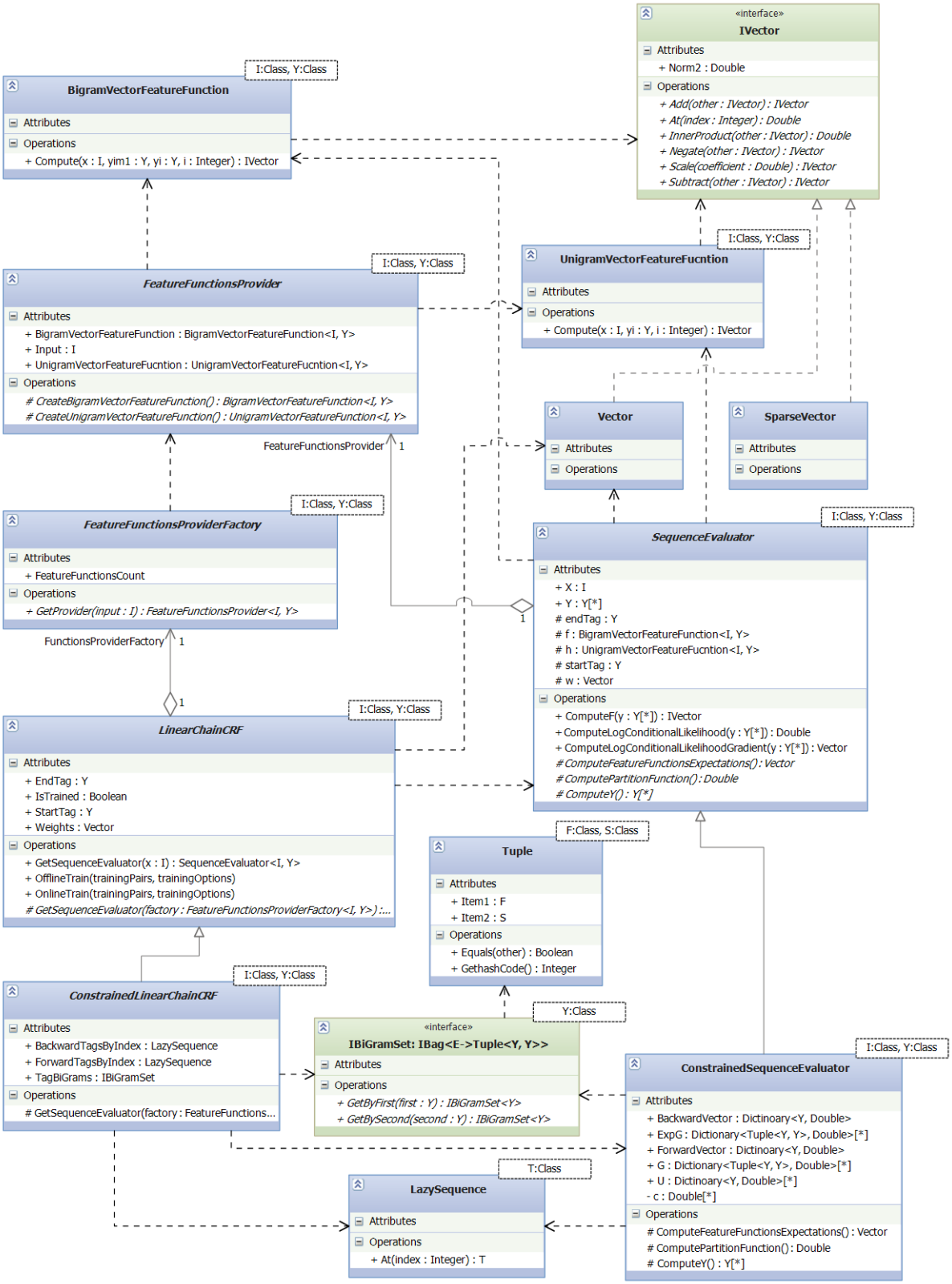
Έτσι, βάσει τής σχέσεως (3.15.1), το μοντέλο πιθανότητος για Linear Chain Conditional Random Field είναι:

$$p(y|x; w) = \frac{\exp w^T \sum_{i=1}^{n} f(x, y_{i-1}, y_i, i)}{Z(x, w)} \tag{3.15.4}$$

Υποθέτουμε ότι η μεταβλητή εξόδου y_0 έχει πάντοτε την ειδική τιμή S και η y_n την ειδική τιμή E.

Τα Linear Chain Conditional Random Fields έχουν χρησιμοποιηθεί με επιτυχία για σήμανση ακολουθιών, για παράδειγμα την σήμανση μερών τού λόγου Αγγλικών κειμένων. Σε γλώσσες όμως με πλουσία μορφολογία όπως τα Αρχαία Ελληνικά, αναφύεται ένα πρόβλημα. Ενώ για τα Αγγλικά ένα τυπικό πλήθος $|Y|$ τών δυνατών σημάνσεων τών λέξεων είναι 46, για το παρόν έργο και τα Αρχαία Ελληνικά είναι τής τάξεως τών 1400, λόγω τής πλουσίας μορφολογίας τής γλώσσας. Ενθυμούμενοι ότι το υπολογιστικό κόστος είναι ανάλογο τού τετραγώνου αυτού τού πλήθους, καταλαβαίνουμε ότι διογκούται κατά τάξεις μεγέθους και γίνεται απαγορευτικό. Γι' αυτόν τον λόγο, όπως θα δούμε στην επομένην υποενότητα, εκμεταλλευόμαστε τους ακολουθιακούς περιορισμούς στις τιμές τών y_i, τους οποίους μαθαίνουμε από εκπαίδευση, ώστε να μειωθεί δραματικά το κόστος υπολογισμών και να φθάσει σε ανεκτά επίπεδα.

3.15.4 Αρχιτεκτονική τής μονάδος Gramma.CRF



Σχήμα 3-11: Βασικά μέρη τής μονάδος Gramma.CRF

Η μονάδα Gramma.CRF αναλαμβάνει να πραγματοποιήσει Linear Chain Conditional Random Fields για αντικείμενα εισόδου με γενικό τύπο I και ακολουθία εξόδου τής οποίας κάθε στοιχείο είναι γενικού τύπου Y. Στο Σχήμα 3-11 φαίνεται ένα υποσύνολο τών κλάσεων τής μονάδος καθώς και μερικές υποστηρικτικές κλάσεις από μονάδες που έχουν προαναφερθεί. Το σημείο εκκινήσεως τής μονάδος είναι η αφηρημένη κλάση LinearChainCRF<I, Y>. Παρέχονται δύο επίσης αφηρημένες απόγονοί της, η FullLinearChainCRF<I, Y> (δεν εικονίζεται στο σχήμα) η οποία δίνει το κλασικό Linear Chain Conditional Random Field τής βιβλιογραφίας, και η ConstrainedLinearChainCRF<I, Y> η οποία εκμεταλλεύεται πληροφορία περιορισμών στις δυνατές ακολουθίες εξόδους για να επιτύχει δραστική αύξηση επιδόσεων, όπως ανεφέρθη στην προηγουμένην υποενότητα 3.15.3.

Κάθε μία από τις απογόνους τής κλάσεως LinearChainCRF<I, Y> χρειάζεται να γνωρίζει ποιες είναι οι συναρτήσεις χαρακτηριστικών που αποτελούν την $f(x, y_{i-1}, y_i, i)$ στην σχέση (3.15.3). Για λόγους επιδόσεων, η μονάδα διακρίνει τις συναρτήσεις χαρακτηριστικών σε αυτές που κάνουν χρήση και τών δύο γειτονικών μεταβλητών y_{i-1} και y_i ώστε έχουν την πλήρη μορφή $h(x, y_{i-1}, y_i, i)$ και σε αυτές που κάνουν χρήση μόνο τής μιας μεταβλητής y_i ώστε έχουν την μορφή $q(x, y_i, i)$. Άρα η $f(x, y_{i-1}, y_i, i)$ αναλύεται ως εξής:

$$f(x, y_{i-1}, y_i, i) = h(x, y_{i-1}, y_i, i) + q(x, y_i, i) \tag{3.15.5}$$

Οι συναρτήσεις h και q αποδίδονται αντιστοίχως από τους τύπους συναρτήσεων (delegates) BigramVectorFeatureFunction<I, Y> και UnigramVectorFeatureFunction<I, Y>. Τα delegates παρέχονται από μία υλοποίηση τής αφηρημένης κλάσεως FeatureFunctionsProvider<I, Y> η οποία κατασκευάζεται κάθε φορά πού έχουμε αποτίμηση για ένα ζεύγος (x, y). Για να γνωρίζει το σύστημα πώς να παρέχει την παραπάνω κλάση όποτε ζητείται για ένα ζεύγος (x, y), οι κατασκευαστές (constructors) τών αφηρημένων κλάσεων υπό την LinearChainCRF<I, Y> ζητούν μεταξύ τών παραμέτρων μία υλοποίηση τής κλάσεως FeatureFunctionsProviderFactory<I, Y>. Αυτό είναι ένα τυπικό παράδειγμα τού σχεδιαστικού μορφήματος “factory method” στην τεχνολογία λογισμικού.

Η εκπαίδευση μιας απογόνου τής LinearChainCRF<I, Y> γίνεται με μία από τις μεθόδους “OfflineTrain” και “OnlineTrain”. Και οι δύο βασίζονται στην μονάδα Gramma.Optimization τής ενότητος 3.11. Η πρώτη χρησιμοποιεί μία μέθοδο τής οικογενείας quasi-Newton όπως την Συζυγούς Κλίσεως, και απαιτεί όλα τα εκπαιδευτικά δεδομένα να είναι στην μνήμη έτοιμα για τυχαία προσπέλαση. Η δεύτερη χρησιμοποιεί Παράλληλη Στοχαστική Κατάβαση Κλίσεως και επιτρέπει να έρχονται τα εκπαιδευτικά δεδομένα σε συνεχή ροή χωρίς ενδιάμεση αποθήκευση, με δυνατότητα διακοπής.

Για την αποτίμηση ενός ζεύγους (x, y), καλούμε με παράμετρο x την μέθοδο “GetSequenceEvaluator” που κληρονομείται στις απογόνους τής LinearChainCRF<I, Y>. Αυτή επιστρέφει μιαν υλοποίηση τής κλάσεως SequenceEvaluator<I, Y>. Από αυτήν μπορούμε να ζητήσουμε την πιθανότερη ακολουθία εξόδου Y, την πιθανότητα μιας δεδομένης εξόδου y με την μέθοδο “ComputeLogConditionalLikelihood” και άλλα.

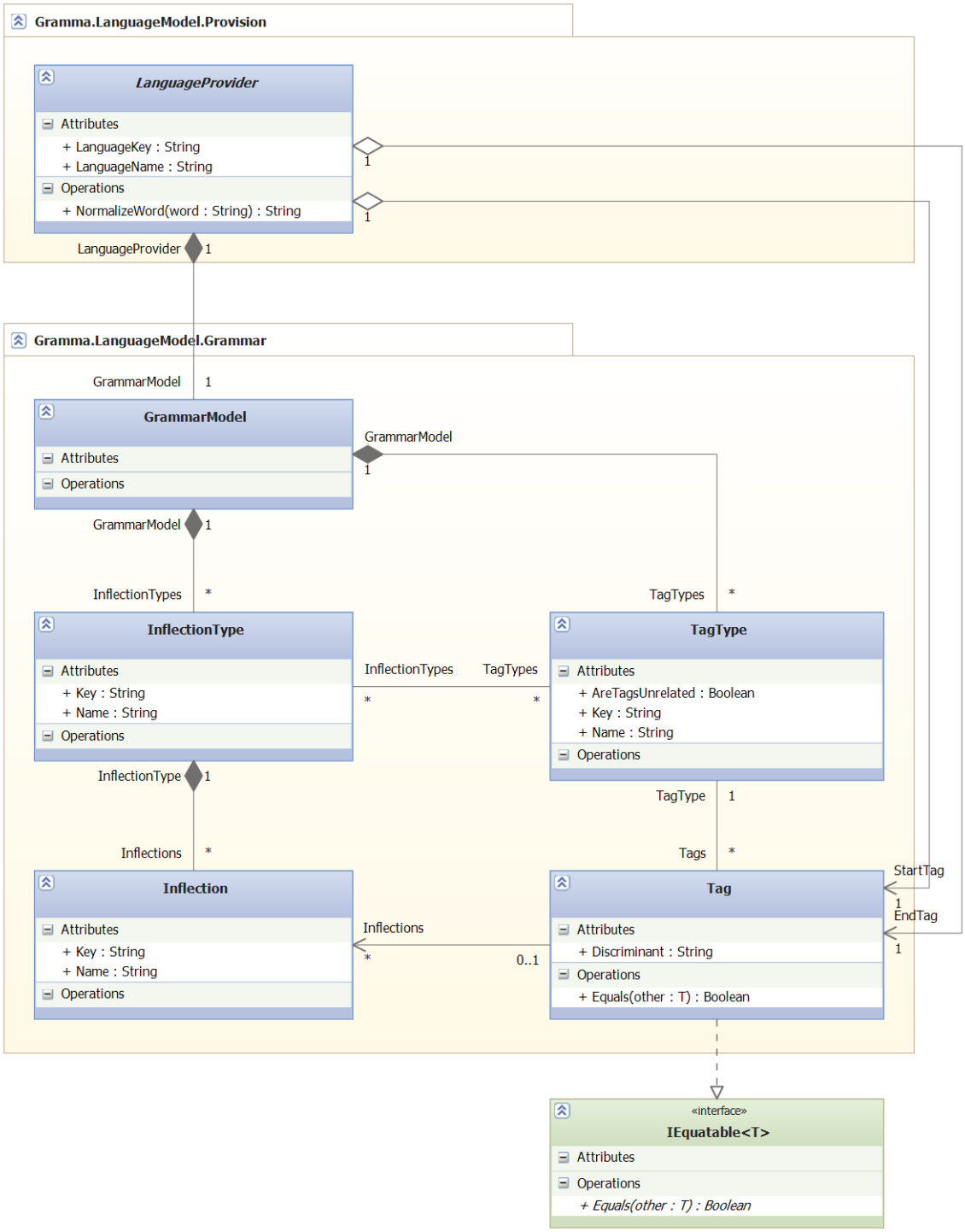
Για βελτίωση στην αριθμητική ευστάθεια κατά την αποτίμηση, έχει ληφθεί ειδική μέριμνα για τον υπολογισμό τών “forward vector” και “backward vector”, που αποτελούν την εξειδίκευση τού “message passing” των Μαρκοβιανών Πιθανοτικών Δικτύων για τοπολογία “Linear Chain”. Δεν ακολουθείται η συνήθης πρακτική εναλλαγής log-exp που βρίσκουμε σε υπάρχουσες βιβλιοθήκες για Conditional Random Fields η οποία είναι ακριβή υπολογιστικά, αλλά έχει προσαρμοσθεί η μεθοδολογία ανακλιμακώσεως την οποία έχει προτείνει ο Rabiner [22], με τις διορθώσεις τού Rahimi [23], για τα παρόμοια “forward vector” και “backward vector” τών Hidden Markov Models.

3.16 Μονάδα Gramma.LanguageModel

Αυτή η μονάδα είναι η βάση για την προσαρμογή τών γλωσσών στο σύστημα. Είναι ο μηχανισμός που πραγματοποιεί την αφαιρετικότητα επί τών ειδικών χαρακτηριστικών μας γλώσσας.

Κάθε υποστηριζομένη γλώσσα οφείλει να υλοποιεί την αφηρημένη κλάση LanguageProvider. Αυτή παρέχει δύο σκέλη, το γραμματικό μοντέλο τής γλώσσας και τούς χειρισμούς συμβολοσειρών. Η δήλωση τής κάθε γλώσσας και τής αντιστοίχου υλοποιήσεως τής κλάσεως LanguageProvider γίνεται σε ένα αρχείο τύπου XAML το οποίο θα δούμε αφού εξετάσουμε και τις μονάδες “Gramma.Inference” και “Gramma.LanguageModel.Greek.TrainingSources”.

3.16.1 Το γραμματικό μοντέλο



Σχήμα 3-12: Το μοντέλο πεδίου που περιγράφει τα μέρη τού λόγου μιας γλώσσας

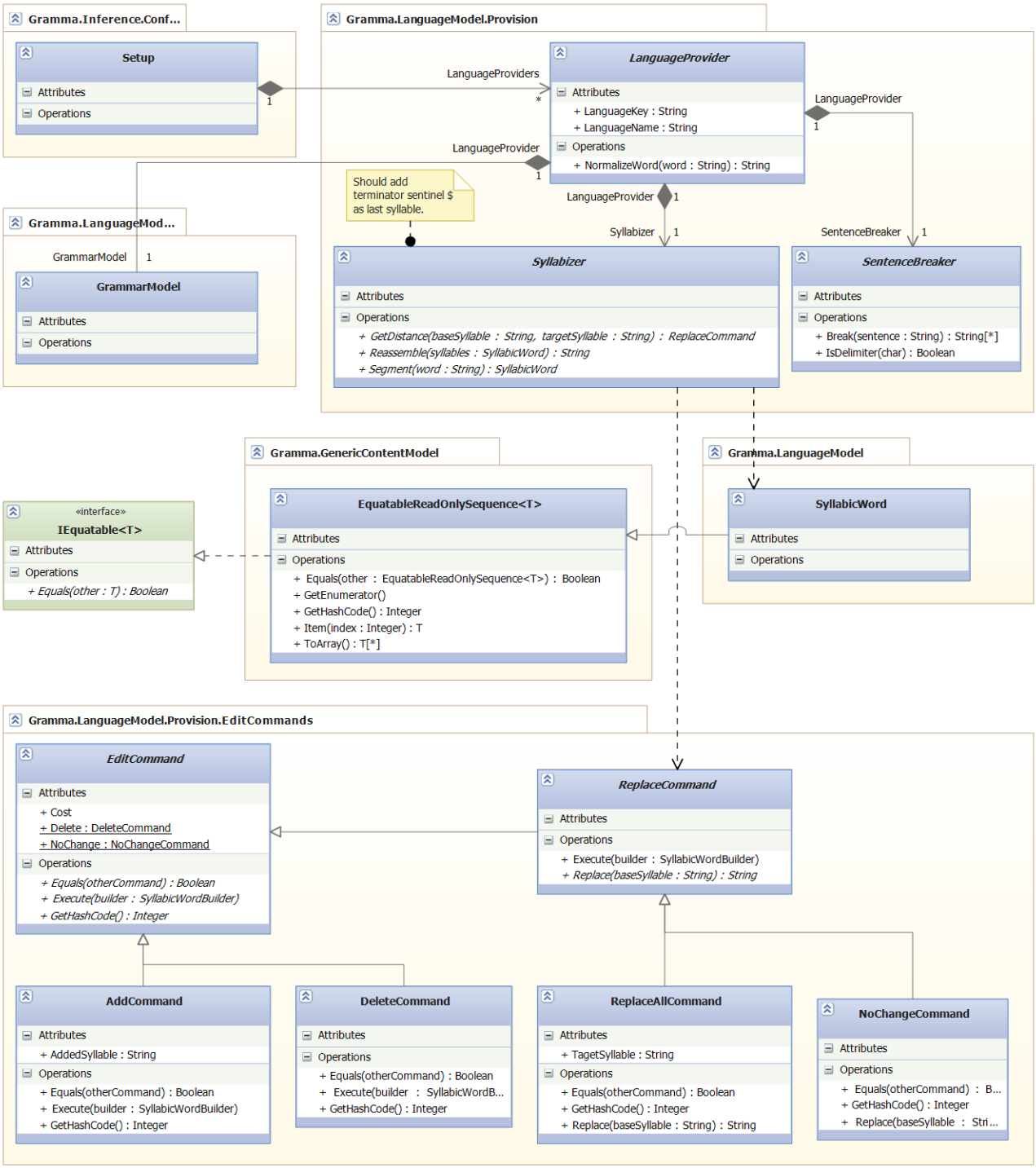
Μία γλώσσα ορίζει το μοντέλο γραμματικής της μέσω αντιστοίχου υλοποιήσεως τής κλάσεως LanguageProvider. Το μοντέλο γραμματικής είναι ένα απλό μοντέλο πεδίου (domain model) τού οποίου οι κλάσεις βρίσκονται στο namespace “Gramma.LanguageModel.Grammar” όπως

φαίνεται στο Σχήμα 3-12. Σε αυτό το μοντέλο περιέχονται τα μέρη τού λόγου τής γλώσσας, οι διάφορες δυνατές κλίσεις τους, ώστε με τα παραπάνω νά μπορεί να σημανθεί μία λέξη.

Ρίζα τού μοντέλου είναι η κλάση GrammarModel. Αυτή περιέχει τα μέρη τού λόγου, όπως «ἄρθρον», «οὐσιαστικόν», «ἐπίθετον», «ἀντωνυμία», «ῥῆμα», «μετοχή», «ἐπίῤῥημα» και λοιπά, παριστάμενα με την κλάση TagType, και τα είδη κλίσεων, όπως «πτῶσις», «γένος», «πρόσωπο», «βαθμός» καί λοιπά, παριστάμενα με την κλάση InflectionType. Τα μέρη τού λόγου και τα είδη τών κλίσεων σχετίζονται μεταξύ τους. Για παράδειγμα, τα μέρη τού λόγου «ἄρθρον», «οὐσιαστικόν», «ἐπίθετον», «ἀντωνυμία», «μετοχή» ἔχουν εἴδη κλίσεως «πτῶσις», «γένος». Τα μέρη τού λόγου «ἐπίθετον», «ἐπίῤῥημα» ἔχουν είδος κλίσεως «βαθμός». Τά μέρη τού λόγου «ῥῆμα», «μετοχή» έχουν είδη κλίσεως «χρόνο», «φωνή». Κάθε είδος κλίσεως έχει τις δυνατές κλίσεις του. Επί παραδείγματι, η «πτῶσις» ἔχει δυνατές κλίσεις «ὀνομαστικὴ ἑνικοῦ», «γενικὴ ἑνικοῦ» και λοιπά, ο «βαθμός» έχει κλίσεις «θετικός», «συγκριτικός», «ὑπερθετικός», ο «χρόνος» έχει κλίσεις «ἐνεστώς», «παρατατικός», «μέλλων» και λοιπά.

Η σήμανση μιας λέξεως παριστάνεται με την κλάση Tag. Αυτή πρέπει να έχει ένα ωρισμένο TagType, δηλαδή ν’ ανήκει σ’ ένα μέρος τού λόγου, και, εάν αυτό είναι κλιτό, να περιέχει τα αντίστοιχα αντικείμενα Inflection που περιστάνουν την κλίση τής λέξεως. Ένα Tag λοιπόν μπορεί για παράδειγμα να έχει TagType τύπου «ἐπίθετον» και Inflections «γενικὴ ἑνικοῦ», «θηλυκόν», «ὑπερθετικός». Η κλάση Tag είναι κεντρικό στοιχείο τού έργου.

3.16.2 Χειρισμοί συμβολοσειρών



Σχήμα 3-13: Ορισμός τού χειρισμού συμβολοσειρών που αντιστοιχούν σε μία γλώσσα

Οι λειτουργίες που περιλαμβάνονται στους χειρισμούς συμβολοσειρών αποτελούνται από την διάσπαση προτάσεων σε λέξεις, την διάσπαση λέξεων σε συλλαβές και ανασυναρμολόγησή

τους, τον υπολογισμό αποστάσεως συλλαβών για αλγορίθμους που δουλεύουν με Generalized Edit Distance.

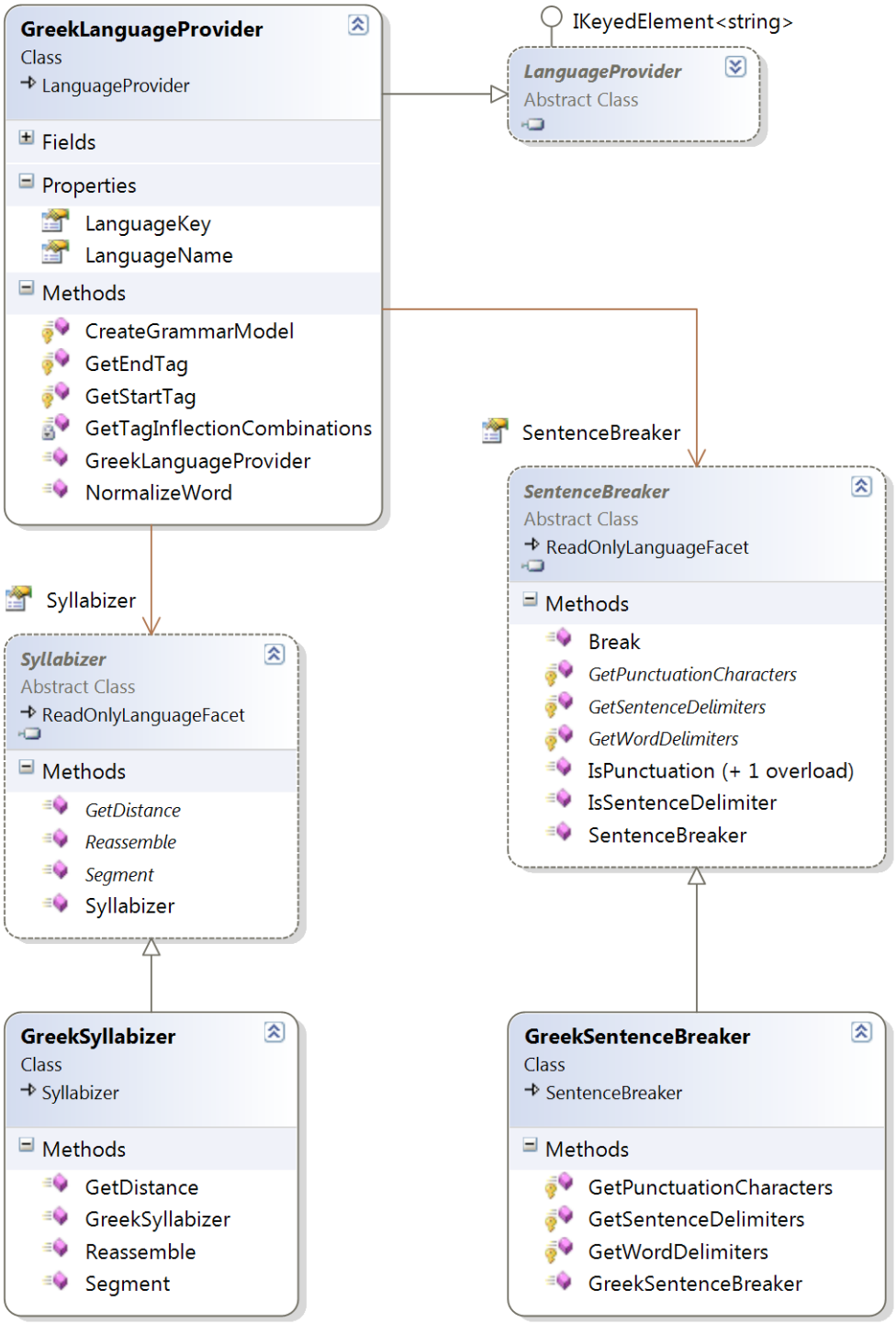
Στο Σχήμα 3-13, βλέπουμε ότι η υλοποίηση LanguageProvider μιας γλώσσας παρέχει υλοποιήσεις τών κλάσεων SentenceBreaker και Syllabizer. Η πρώτη ορίζει πού χωρίζεται ένα κείμενο στις προτάσεις του, συμφώνως με τα σημεία στίξεως τής γλώσσας, και διασπά τις προτάσεις στις λέξεις τους. Η δεύτερη αναλαμβάνει τον συλλαβισμό μιας λέξεως, δηλαδή διασπά μία λέξη στις συλλαβές της, ανασυναρμολογεί συλλαβές σε λέξη και δίνει την απόσταση δύο συλλαβών. Στο εξής θα αναφερόμαστε στην διασπασμένη σε συλλαβές λέξη ως «συλλαβολέξη».

Η απόσταση δύο συλλαβών δηλούται με την ιδιότητα “Cost” μιας απογόνου τής αφηρημένης κλάσεως ReplaceCommand. Η κλάση ReplaceCommand είναι μέρος τού μηχανισμού “Generalized Edit Distance” τού συστήματος, καταγομένη από την αφηρημένη κλάση EditCommand, και δηλώνει όχι μόνο την «απόσταση» τών συλλαβών με την ιδιότητα “Cost” αλλά και την αντίστοιχο ενέργεια που μετατρέπει την μία συλλαβή στην άλλην. Έτσι, με την βοήθεια τής παρεχομένης κλάσεως ReplaceCommand, το σύστημα βρίσκει όχι μόνο την ελαχίστη απόσταση μεταξύ δύο συλλαβολέξεων αλλά δίνει και την ελαχίστη ακολουθία εντολών που μετατρέπει την μία συλλαβολέξη στην άλλη.

Η μονάδα δίνει τις ακόλουθες έτοιμες υλοποιήσεις τής κλάσεως ReplaceCommand για συνήθεις περιπτώσεις. Η κλάση NoChangeCommand δηλοί ότι οι συλλαβές είναι ίσες με «κόστος» μετατροπής μηδέν. Η κλάση ReplaceAllCommand δηλοί ότι οι συλλαβές είναι εντελώς διαφορετικές κατά την ερμηνεία τής γλώσσας, και ως συλλαβική εντολή αντικαθιστά όλην την συλλαβή τής μιας λέξεως με την συλλαβή τής άλλης με το «υψηλό κόστος» 1.0.

Όπως θα δούμε σε επομένην ενότητα, μία γλώσσα μπορεί να ορίσει δικές της απογόνους τής κλάσεως ReplaceCommand ώστε να δηλώσει συλλαβικά πάθη ιδιαίτερα στην γλώσσα με χαμηλότερα «κόστη», όπως τροπή φωνηέντων ή συμφώνων.

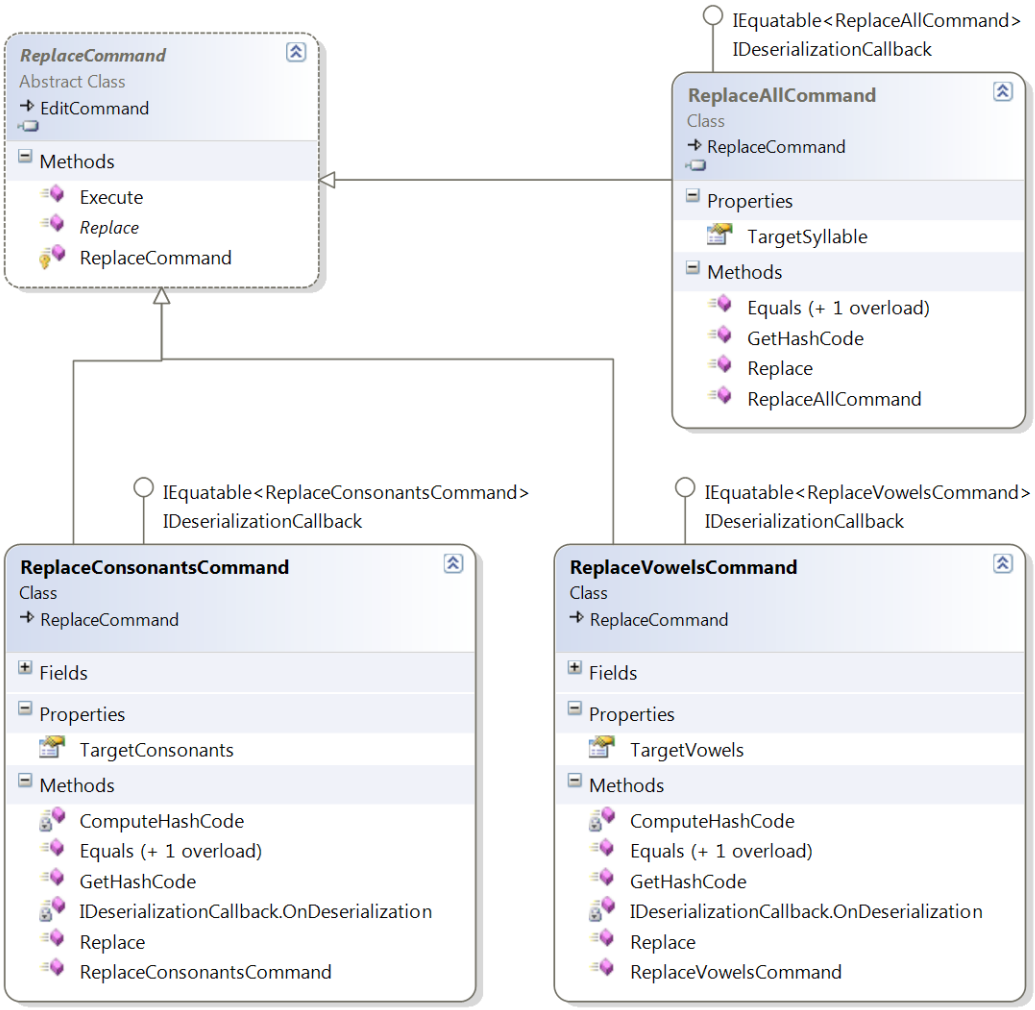
3.17 Η μονάδα Gramma.LanguageModel.Provision.Greek



Σχήμα 3-14: Με έντονα γράμματα εικονίζονται οι κλάσεις που προσαρμόζουν την Αρχαία Ελληνική γλώσσα στο σύστημα

Η μονάδα αυτή προσαρμόζει στο σύστημα την Αρχαία Ελληνική γλώσσα, δίνοντας με την κλάση GreekLanguageProvider μίαν υλοποίηση τής κλάσεως LanguageProviderκαθώς και συνοδευτικές υλοποιήσεις άλλων αφηρημένων κλάσεων τής προηγουμένης ενότητος. Στο Σχήμα 3-14 με έντονα γράμματα εικονίζονται κλάσεις αυτής τής μονάδος ενώ με αχνά εικονίζονται κλάσεις τής προηγουμένης μονάδος “Gramma.LanguageModel” τις οποίες αυτές οι κλάσεις κληρονομούν.

Η κλάση GreekLanguageProvider δίνει στην ιδιότητα “SentenceBreaker” την απόγονο κλάση GreekSentenceBreaker. Ομοίως, στην ιδιότητα “Syllabizer” δίνει την απόγονο κλάση “GreekSyllabizer”. Η τελευταία, στην μέθοδο “GetDistance” δεν επιστρέφει μόνο την υπάρουσα κλάση ReplaceAllCommand που είδαμε στην μονάδα “Gramma.LanguageModel” αλλά, όπως φαίνεται στο Σχήμα 3-15, επιστρέφει κατά περίπτωση και τις ειδικές για τα Αρχαία Ελληνικά κλάσεις ReplaceVowelsCommand και ReplaceConsonantsCommand για να δηλώσει τροπή φωνηέντων και συμφώνων μιας συλλαβής αντιστοίχως.



Σχήμα 3-15: Με έντονα γράμματα εικονίζονται οι συλλαβικές εντολές ειδικές για τα Αρχαία Ελληνικά

3.18 Το βοηθητικό πρόγραμμα LXXCombiner

Το LXXCombiner δεν είναι μονάδα αλλά αυτοτελές πρόγραμμα. Ο σκοπός του είναι να παραγάγει σεσημασμένη Παλαιά Διαθήκη κατά τούς Εβδομήκοντα ώστε να γίνει εκπαιδευτική πηγή.

Η ανάγκη δημιουργίας τού προγράμματος ήταν ότι οι ήδη διαθέσιμες πηγές είχαν διάφορες ελλείψεις. Είτε σε μία πηγή έλειπε ο τονισμός, είτε σε άλλες πηγές έλειπε η στίξη είτε σε άλλη πηγή υπήρχε μόνο η σήμανση τών λέξεων και τα λήμματά τους, όχι όμως το αρχικό κείμενο. Για την παραγωγή σωστής εκπαιδευτικής πηγής, το βοηθητικό αυτό πρόγραμμα συνδυάζει δύο εξ αυτών τών πηγών που να έχουν εκπονηθεί στην ίδια κριτική έκδοση “Rahlfs” τής Παλαιάς Διαθήκης. Η μία πηγή είναι το καθαρό κείμενο, η άλλη πηγή περιέχει μόνο την μορφολογία και το λήμμα κάθε λέξεως. Το πρόγραμμα ευθυγραμμίζει, συνδυάζει τις δύο πηγές και παράγει μίαν έξοδο.

Η φύση τής διαδικασίας αυτής ταίριαζε ώστε το πρόγραμμα να υλοποιηθεί πολύ αποδοτικά και συνοπτικά στην συναρτησιακή γλώσσα F#.

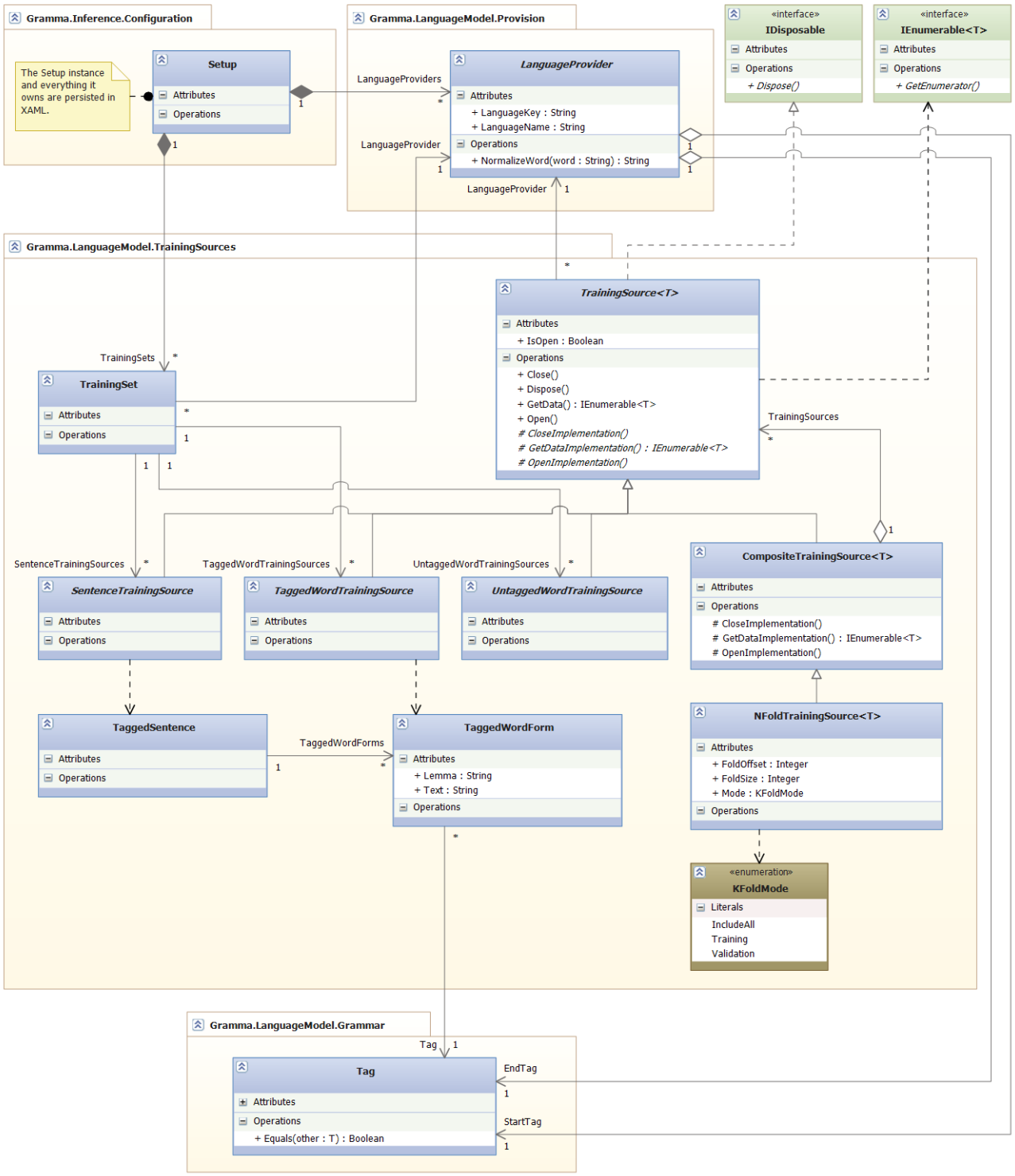
3.19 Η μονάδα Gramma.Inference

Το κέντρο τής εργασίας αυτής είναι η μονάδα “Gramma.Inference”. Πραγματοποιεί τους σκοπούς τού έργου:

- Ανοίγει πηγές εκπαιδευτικών δεδομένων και εκπαιδεύει το σύστημα.
- Εκπαίδευση ταξινομητών χαρακτηριστικών λέξεως.
- Εκπαίδευση Μαρκοβιανού δικτύου αποτιμήσεως προτάσεων.
- Μετά την εκπαίδευση, εκτιμά τά μέρη τού λόγου τών λέξεων μιας προτάσεως και προτείνει τα λήμματα τών λέξεων.

Αυτά θα εκτεθούν στην συνέχεια.

3.19.1 Άντληση εκπαιδευτικών δεδομένων



Σχήμα 3-16: Το σύστημα αντλήσεως εκπαιδευτικών δεδομένων

Όλες οι πηγές δεδομένων ακολουθούν το πρότυπο τής γενικευμένης αφηρημένης κλάσεως TrainingSource<T> η οποία εικονίζεται στο Σχήμα 3-16. Το σχεδιαστικό μόρφημα που

ακολουθείται εδώ είναι το “Inversion of Control” ή γενικότερα το “Strategy”. Για να αντλήσει το σύστημα εκπαιδευτικά δεδομένα, αναμένει ότι η πηγή υλοποιεί την κατάλληλο απόγονο τής κλάσεως αυτής.

Ο τύπος T παριστάνει τον τύπο τών εκπαιδευτικών δεδομένων. Με εξειδίκευση τού τύπου T λαμβάνουμε τις συγκεκριμένες πηγές δεδομένων. Έτσι, στην αφηρημένη απόγονο TaggedWordTrainingSource ο τύπος T έχει τεθεί να είναι η κλάση TaggedWordForm, η οποία κρατεί μία μορφή λέξεως στο πεδίο “Text”, την σήμανσή της στην σχέση “Tag” και το λήμμα της στο πεδίο “Lemma”. Στην αφηρημένη απόγονο SentenceTrainingSource, ο τύπος T έχει τεθεί να είναι συλλογή από αντικείμενα TaggedWordForm η οποία απεικονίζει μία πλήρως σεσημασμένη πρόταση.

Κάθε μία υλοποίηση τών παραπάνω κλάσεων πρέπει να αποδώσει τις μεθόδους “OpenImplementation”, “GetDataImplementation” και “CloseImplementation” για να ανοιχθεί η πηγή, να αντληθούν δεδομένα καν να κλεισθεί η πηγή αντιστοίχως.

Για την σύνθεση πηγών, το σύστημα παρέχει έτοιμες βοηθητικές πηγές που υλοποιούν το σχεδιαστικό μόρφημα “Composite”, δηλαδή είναι περιέκτες άλλων πηγών. Η κλάση CompositeTrainingSource<T>, καταγομένη από την κλάση TrainingSource<T>, μπορεί να περιέχει ένα πλήθος άλλων πηγών τύπου TrainingSource<T>. Η απόγονός της NFoldTrainingSource<T> δίνει επιπλέον την δυνατότητα να εξαιρούνται κλάσματα από τα δεδομένα τών πηγών ώστε να μπορεί να διεξαχθεί χειροκίνητος επαλήθευση τύπου “N-fold validation”. Η κλάση χρησιμοποιείται και εσωτερικά από το σύστημα για να επιτύχει αυτόματο “N-fold validation”.

3.19.2 Δήλωση γλωσσών και αντιστοίχων εκπαιδευτικών πηγών

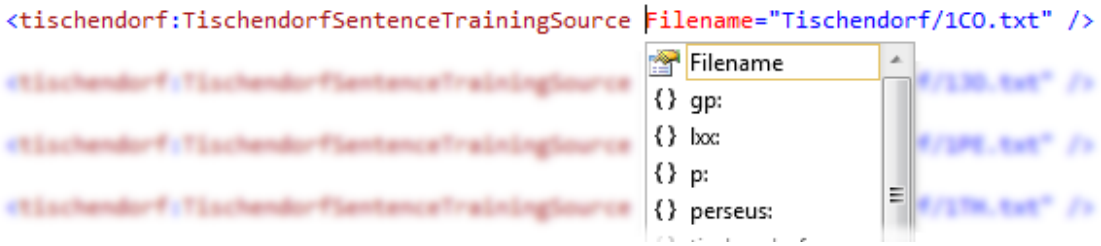
Στο Σχήμα 3-13 και στο Σχήμα 3-16, επάνω αριστερά, βλέπουμε ότι οι διάφορες υλοποιήσεις τής κλάσεως LanguageProvider περιέχονται στην συλλογή “LanguageProviders” τής κλάσεως Setup. Ακόμη, προαιρετικά, όταν η εφαρμογή μας χρησιμοποιεί την μονάδα και για εκπαίδευση, βλέπουμε στο Σχήμα 3-16 ότι η κλάση Setup περιέχει υπό την συλλογή “TrainingSets” και τις εκπαιδευτικές πηγές για την εκάστοτε γλώσσα.

Πώς όμως φορτώνεται και αποθηκεύεται η κλάση Setup; Θα μπορούσε να είναι ένα κομμάτι κώδικος αρχικοποιήσεως που την δημιουργεί, είτε αμέσως είτε με κάποια υποδομή “Dependency Injection” από τις πολλές που διατίθενται. Κάθε φορά όμως που θα γίνεται αλλαγή στα περιεχόμενα και τις ρυθμίσεις θα χρειαζόταν το εκτελέσιμο μεταγλώττιση πάλι. Εναλλακτικά, μερικές υποδομές “Dependency Injection” επιτρέπουν να περιγράφεται η αρχικοποίηση τής κλάσεως σ΄ ένα XML αρχείο με κάποια ειδική σύνταξη. Αυτή η τακτική αποφεύγει την μεταγλώττιση. Όμως, για σύνθετες δομές αντικειμένων όπως τού παρόντος έργου, η σύνταξη τού XML αρχείου τείνει να γίνει θορυβώδης και φλύαρη. Παράδειγμα θα φέρουμε την υποδομή “Spring” που είναι γνωστή και από τον κόσμο τής Java. Τα elements και τα attributes τού XML αρχείου λέγονται “bean”, “property”, “value”, υπάρχουν παντού στο αρχείο γεμίζοντας οπτικό θόρυβο ενώ η ωφέλιμος πληροφορία είναι στις τιμές τών attributes

και στα περιεχόμενα τών “value” elements. Έτσι, σε ένα πολύ απλό παράδειγμα αποσπάσματος ενός τέτοιου XML αρχείου, για να δηλώσουμε και ν΄ αρχικοποιήσουμε μία πηγή εκπαιδευτικών προτάσεων που λέγεται TischendorfSentenceTrainingSource, το αντίστοιχο κομμάτι τού αρχείου XML κατά την υποδομή Spring θα ήταν κάπως έτσι:

```
<bean id="myDataSource"
class="Gramma.LanguageModel.Greek.TrSources.TischendorfSentenceTrainingSource">
  <property name="Filename">
    <value>Tischendorf/1CO.txt</value>
  </property>
</bean>
```

Αντί για το παραπάνω και τα παρόμοια, προκρίθηκε η τεχνολογία XAML [24], η οποία είναι XML αναπαράσταση και αυτή αλλά απεικονίζει απ΄ ευθείας τά ονόματα τών κλάσεων σε elements και τις ιδιότητές τους σε attributes ή elements. Το αντίστοιχο παράδειγμα κομματιού XAML είναι:

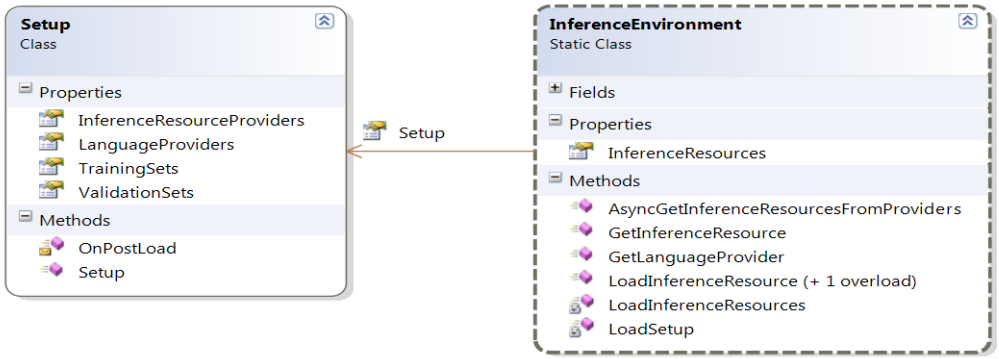


Στην εικόνα βλέπουμε ότι όχι μόνο το κομμάτι XML είναι τελείως καθαρό, περιέχον μόνο ωφέλιμο πληροφορία, αλλά επίσης το περιβάλλον αναπτύξεως (εδώ είναι το Visual Studio 2010) γνωρίζει να προτείνει τα κατάλληλα elements και attributes γιατί καταλαβαίνει τις αντίστοιχες κλάσεις όπου αυτά απεικονίζονται.

Η θέση τού αρχείου XAML δηλούται σε ένα άλλο αρχείο XML με κατάληξη “.config” (Application Configuration File), το οποίο συνοδεύει μία εφαρμογή .NET και έχει ίδιο πρόθεμα με το εκτελέσιμο τής εφαρμογής. Εκεί, πρέπει να δηλωθεί μέσα στην ενότητα “configSections” το στοιχείο “inferenceSection”, έπειτα να δοθεί το στοιχείο “inferenceSection” με την ιδιότητα “setupXamlPath” να δηλώνει το αρχείο XAML, όπως παρακάτω:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>

  <configSections>
    <section name="inferenceSection"
             type="Gramma.Inference.Configuration.InferenceConfigurationSection,
Gramma.Inference, Version=1.0.0.0" />
    </configSections>
    ...
    ...
    <inferenceSection setupXamlPath="Setup.xaml" />

</configuration>
```

Η ρίζα τού γράφου αντικειμένων που αποθηκεύεται στὸ αρχείο XAML είναι ένα αντικείμενο τής κλάσεως Setup το οποίο εικονίζεται στο Σχήμα 3-17.

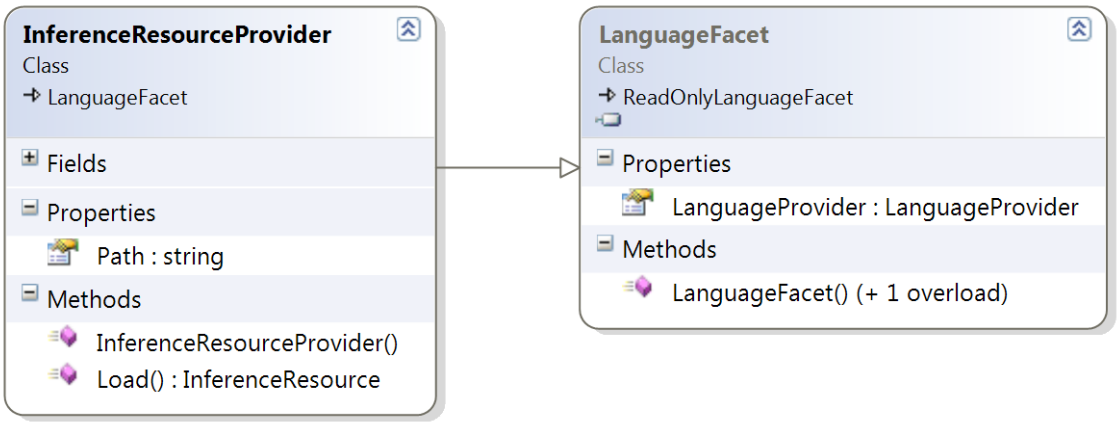


Σχήμα 3-17: Ένα αντικείμενο τύπου Setup είναι η ρίζα τού αρχείου XAML και δίνεται αυτομάτως από την ιδιότητα "Setup" τής στατικής κλάσεως InferenceEnvironment

Οι ιδιότητες τής κλάσεως είναι:

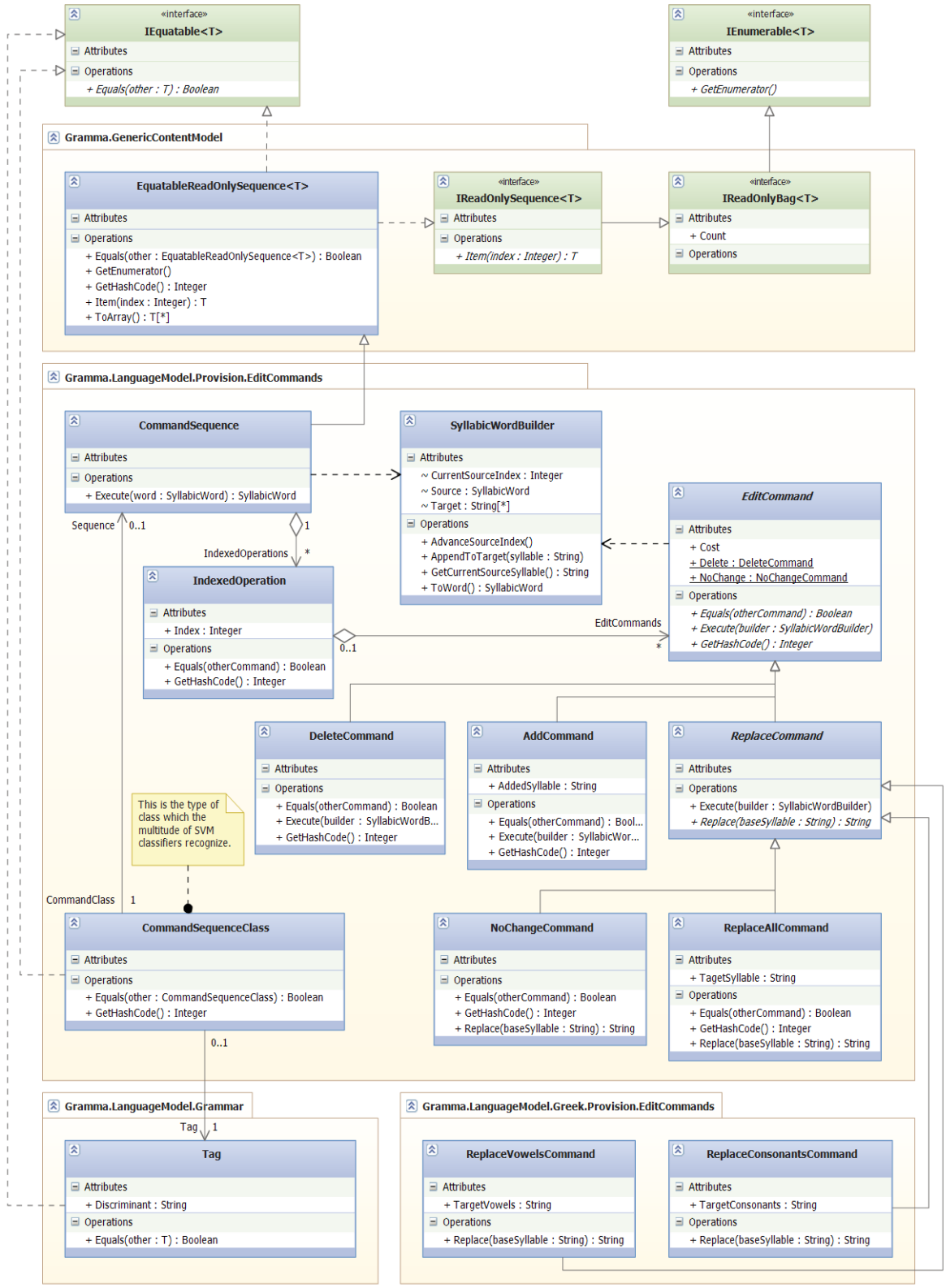
- **LanguageProviders:** Περιέχει την συλλογή τών γλωσσών που υποστηρίζονται από το σύστημα.
- **TrainingSets:** Προαιρετικά, περιέχει σύνολα πηγών δεδομένων εκπαιδεύσεως για υποστηριζόμενες γλώσσες. Για μία εφαρμογή που επιτελεί εκπαίδευση, σε αυτήν την ιδιότητα πρέπει να περιέχονται οι πηγές.
- **ValidationSets:** Προαιρετικά, περιέχει σύνολα πηγών δεδομένων προς αποτίμηση αποδόσεως (validation) για υποστηριζόμενες γλώσσες.
- **InferenceResourceProviders:** Για μία εφαρμογή που χρησιμοποιεί προεκπαιδευμένους αποθηκευμένους πόρους, η ιδιότητα αυτή περιέχει αντικείμενα τύπου InferenceResourceProvider τα οποία ορίζουν από πού θα φορτωθούν οι πόροι για μία υποστηριζομένη γλώσσα. Όπως φαίνεται στο Σχήμα 3-18, καθένα από αυτά στην ιδιότητα "Path" κρατεί το αρχείο από όπου θα φορτωθούν οι πόροι, και στην "LanguageProvider" κρατεί την γλώσσα που αφορούν οι πόροι που θα φορτωθούν. Οι πόροι που προκύπτουν από την παραπάνω φόρτωση είναι τύπου InferenceResource, τον οποίο θα δούμε στην επομένη υποενότητα.

Όταν έχει δηλωθεί καταλλήλως το στοιχείο "inferenceSection" που είδαμε στο αρχείο config, τότε το αντικείμενο τύπου Setup που περιέχεται στο XAML αρχείο τίθεται αυτομάτως στην ιδιότητα "Setup" τής στατικής (singleton) κλάσεως InferenceEnvironment.



Σχήμα 3-18: Η κλάση InferenceResourceProvider δείχνει από που θα φορτωθούν εκπαιδευμένοι πόροι για μία γλώσσα

3.19.3 Το σύστημα τών συλλαβικών εντολών και οι γραμματικές κλάσεις



Σχήμα 3-19: Γραμματικές κλάσεις, σήμανση λέξεων και συλλαβικές εντολές

Όπως έχει αναφερθεί στο κεφάλαιο 2 όπου γίνεται επισκόπηση τού συστήματος, το σύστημα έχει δύο στάδια. Το πρώτο εξάγει χαρακτηριστικά για κάθε λέξη μεμονωμένα και το δεύτερο συνθέτει τα δεδομένα τών λέξεων σε κάθε πρόταση. Στην ενότητα αυτή θα δούμε ποία είναι τα χαρακτηριστικά λέξεως όπως τα εξάγει το πρώτο στάδιο.

Κάθε μορφή λέξεως σε μία πρόταση έχει την γραμματική σήμανσή της. Αυτή απεικονίζεται με την κλάση Tag και τα συστατικά της, όπως τα είδαμε στην προηγουμένην ενότητα 3.16.1 για το γραμματικό μοντέλο και στο Σχήμα 3-12. Την κλάση Tag την βλέπουμε πάλι, χωρίς τα παρελκόμενά της, στο Σχήμα 3-19 κάτω αριστερά. Ακόμη, κάθε μορφή λέξεως σε μία πρόταση έχει και το αντίστοιχο λήμμα της, το οποίο είναι η κανονική της μορφή. Εξετάζοντας μεμονωμένα μία μορφή λέξεως, η γραμματική σήμανσή της και το λήμμα της συχνά δεν είναι βεβαία. Παράδειγμα στα Αρχαία Ελληνικά είναι όλα τα ουδέτερα κλιτά, καθώς η ονομαστική, η αιτιατική και η κλιτική πτώση γράφονται ομοίως. Άλλο παράδειγμα όπου έχουμε και αμφίβολο λήμμα είναι η λέξη «μένει» που μπορεί να είναι είτε το τρίτο πρόσωπο τού ρήματος «μένω» είτε η δοτική ενικού τού ουδετέρου ουσιαστικού «μένος», όπως είδαμε και στην εισαγωγική ενότητα 1.1. Την επιλογή μεταξύ τών πιθανών ενδεχομένων αναλαμβάνει το δεύτερο στάδιο το οποίο αποτιμά την πρόταση, δηλαδή τα συμφραζόμενα.

Υπενθυμίζουμε ότι κάθε λέξη στο σύστημα παριστάνεται συλλαβικά, με «συλλαβολέξεις». Αφού κάθε μορφή συλλαβολέξεως έχει ένα αντίστοιχο πιθανό συλλαβολήμμα, τότε ορίζουμε την ελαχίστη ακολουθία συλλαβικών εντολών που μετατρέπει την μορφή στο λήμμα. Αυτή γίνεται με τον μηχανισμό “Generalized Edit Distance” που προσφέρει η μονάδα “Gramma.Indexing” όπως είδαμε στην υποενότητα 3.12.1. Στον μηχανισμό συλλαβικών αποστάσεων υπεισέρχεται και ο πάροχος γλώσσας όπως είδαμε στην υποενότητα 3.16.2. Βλέπουμε λοιπόν στο κάτω μέρος στο Σχήμα 3-19 την συνεισφορά τής μονάδος “Gramma.LanguageModel.Greek.Provision”. Παράγεται έτσι μία ακολουθία αντικειμένων καταγομένων από την αφηρημένη κλάση EditCommand, όπως βλέπουμε στο μέσον στο Σχήμα 3-19, τα οποία συσκευάζονται στην κλάση IndexedOperation. Η τελευταία προσθέτει πληροφορία για την θέση σε μία συλλαβολέξη όπου η συλλαβική εντολή δρα. Η δε ακολουθία αναπαρίσταται με την κλάση CommandSequence, η οποία περιέχει την σειρά τών αντικειμένων IndexedOperation. Η κλάση, καθώς κατάγεται από την EquatableReadOnlySequence<IndexedOperation>, επίσης υποστηρίζει ταχείς τελεστές ισότητος και hash codes που βασίζονται στα περιεχόμενα τής ακολουθίας. Έτσι, μπορεί να συμμετάσχει πολύ αποδοτικά σε δομές δεδομένων όπου η ισότης ακολουθιών σημαίνει ισότητα τών περιεχομένων ένα προς ένα.

Με δεδομένο ότι ο σκοπός είναι να βρούμε με ποιο μηχανισμό θα συλλάβουμε την έννοια ότι οι λέξεις στην ίδια γραμματική οικογένεια έχουν την ίδια μορφολογία, η ιδέα τής δομής τών χαρακτηριστικών τού συστήματος βασίζεται στην εξής παρατήρηση: Οι λέξεις οι οποίες έχουν ίδια γραμματική σήμανση και ανήκουν στην ίδια γραμματική οικογένεια έχουν κοινή ακολουθία CommandSequence. Για παράδειγμα, τα αρσενικά ουσιαστικά τής δευτέρας κλίσεως έχουν την ίδια μορφολογία, η οποία μεταφράζεται σε κοινή ακολουθία εντολών μετατροπής από μορφή σε λήμμα. Συγκεκριμένα, η συλλαβομορφή «ἀνθρ-ωπ-ους» αιτιατικής

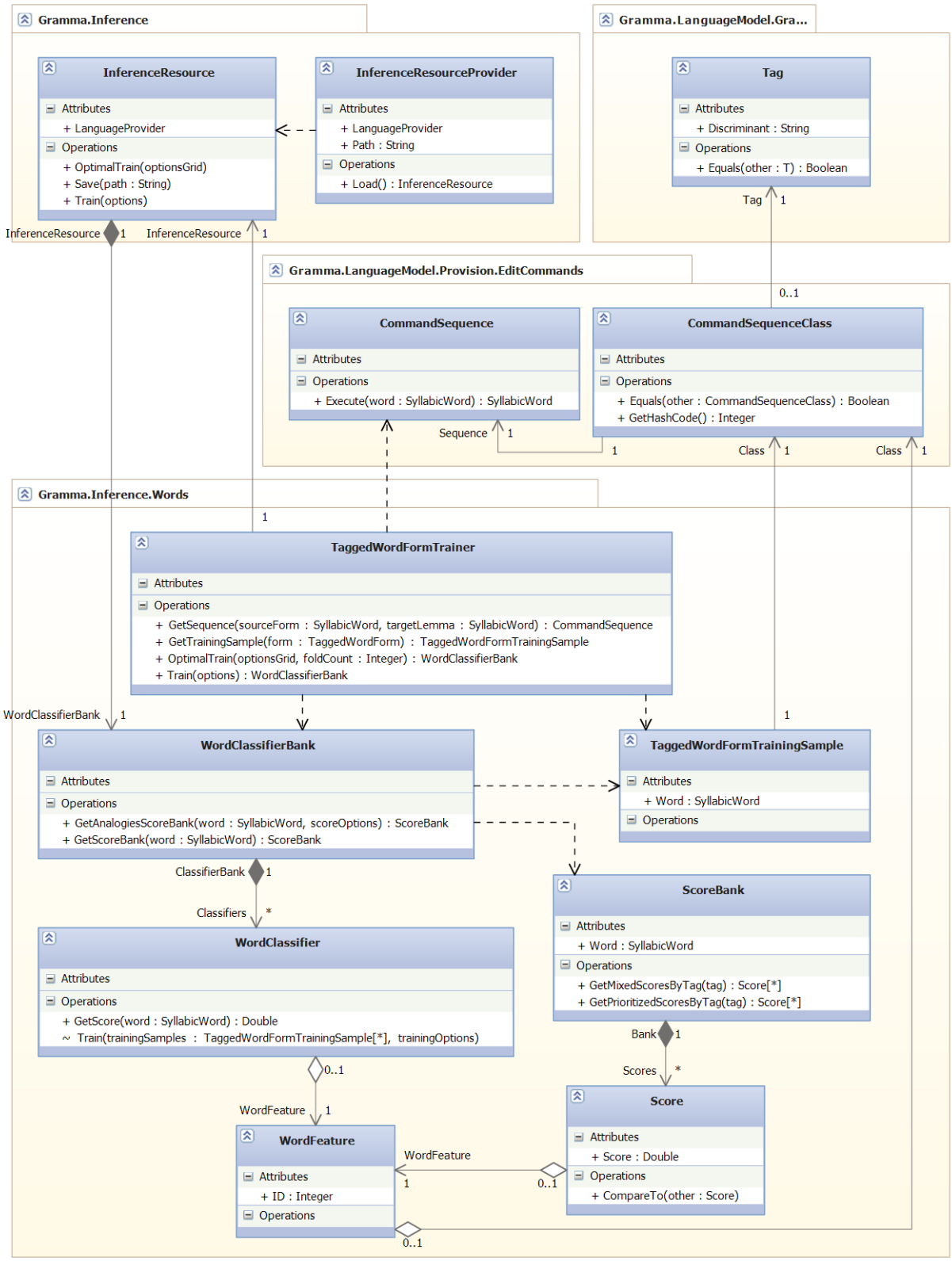
πληθυντικού αρσενικού ουσιαστικού δευτέρας κλίσεως μετατρέπεται στο λήμμα «άνθρ-ωπ-ος» με μία ακολουθία εντολών η οποία περιέχει μία μόνη εντολή «άλλαξε το φωνήεν τής ληγούσης σε ‘ο’». Την ίδια όμως ακολουθία εντολών που οδηγεί στο λήμμα έχει και κάθε άλλη μορφή αρσενικού ουσιαστικού δευτέρας κλίσεως σε πτώση αιτιατική πληθυντικού, όπως «κινδ-υν-ους», «μολ-υβδ-ους», «ὑμν-ους». Συνεπώς το σύστημα θεωρεί ως εκτιμώμενο χαρακτηριστικό τον συνδυασμό τών συλλαβικών εντολών, που μετατρέπουν την μορφή στο λήμμα, και τής γραμματικής σημάνσεως, η οποία περιέχει την κλίση. Ο συνδυασμός αυτός αναπαριστάνεται στο Σχήμα 3-19 με την κλάση CommandSequenceClass, η οποία είναι επίσης εξοπλισμένη με ταχείς τελεστές ισότητος και hash code.

Συνάγουμε λοιπόν ότι η κλάση CommandSequenceClass γίνεται χαρακτηριστικό μορφής λέξεως, ή αλλιώς κλάση[1] ταξινομήσεως για την οποία ένας ταξινομητής επιστρέφει θετικό αριθμό όταν η λέξη ανήκει σ΄ αυτήν. Από τις εκπαιδευτικές πηγές μορφών λέξεων ανακαλύπτονται όλες οι κλάσεις ταξινομήσεως στις οποίες μπορούν να ανήκουν οι λέξεις, και εκπαιδεύονται ισόποσοι ταξινομητές ώστε να επιστρέφουν θετική τιμή αν πιθανώς μια μορφή λέξεως ανήκει στην αντίστοιχο μορφολογική περίπτωση. Αυτές ακριβώς οι δέσμες τιμών που προκύπτουν είναι τα χαρακτηριστικά που αποδίδει το σύστημα σε μία μορφή λέξεως. Για τα Αρχαία Ελληνικά, από το διαθέσιμο εκπαιδευτικό υλικό προκύπτουν δέσμες τών 140 χιλιάδων χαρακτηριστικών ανά λέξη.

Περισσότερα για την λειτουργία τών ταξινομητών αυτών θα δούμε στα επόμενα.

[1] Προσοχή στην διπλή χρήση τής λέξεως «κλάση»· εδώ δεν εννοούμε λογισμικού αλλά ταξινομήσεως.

3.19.4 Πρώτο στάδιο: Χαρακτηριστικά λέξεων



Σχήμα 3-20: Το σύστημα εξαγωγής χαρακτηριστικών λέξεων

Το κέντρο και για τα δύο στάδια είναι η κλάση InferenceResource. Δημιουργείται για κάθε γλώσσα που υπάρχει στο σύστημα και συνδέεται με αυτήν με την ιδιότητα “LanguageProvider” που έχει τύπο την ομώνυμο κλάση που είδαμε στα προηγούμενα. Η κλάση InferenceResource περιέχει τους πόρους και για τα δύο στάδια. Οι πόροι αυτοί σχηματίζονται είτε από εκπαίδευση είτε από φόρτωμα αποθηκευμένων εκπαιδευμένων πόρων. Στο Σχήμα 3-20 η κλάση εικονίζεται επάνω αριστερά. Η εκπαίδευση γίνεται με τις μεθόδους “Train” και “OptimalTrain”. Η πρώτη ομάδα μεθόδων “Train” εκπαιδεύει όλους τους ταξινομητές τού συστήματος με κοινές δεδομένες παραμέτρους ενώ η δεύτερη ομάδα “OptimalTrain” εκτελεί n-fold cross validation για καθένα από τους ταξινομητές επί μιάς μήτρας παραμέτρων ώστε να διαλέξει γι΄ αυτόν τις καλύτερες παραμέτρους. Το φόρτωμα όλης τής κλάσεως μαζί με τους εκπαιδευμένους πόρους της αυτοματοποιείται με αντικείμενα τύπου InferenceResourceProvider, τα οποία, όπως είδαμε, περιέχονται στο αντικείμενο τύπου Setup που έχει ορισθεί στο αρχείο XAML.

Εν τέλει, τα φορτωμένα ή άρτι εκπαιδευθέντα αντικείμενα τύπου InferenceResource κρατούνται στην ιδιότητα “InferenceResources” τής στατικής (singleton) κλάσεως InferenceEnvironment η οποία εικονίζεται στο σχήμα Σχήμα 3-17.

Για το πρώτο αυτό στάδιο τού χαρακτηρισμού λέξεων, η κλάση InferenceResource προσφέρει τήν ζητούμένη λειτουργικότητα μέσω τής ιδιότητος “WordClassifierBank”, η οποία περιέχει αντικείμενο τής ομωνύμου κλάσεως, εικονιζόμενο στο μέσον τού σχήματος. Το αντικείμενο είτε φορτώνεται μέσω .NET serialization και τού μηχανισμού που είδαμε, είτε κατασκευάζεται από τις μεθόδους “Train” και “OptimalTrain” με την βοήθεια τής εσωτερικής κλάσεως TaggedWordFormTrainer. Η τελευταία παίρνει τα δεδομένα τών πηγών σεσημασμένων μορφών λέξεων και λημμάτων που είδαμε στην υποενότητα 3.19.1 και το Σχήμα 3-16 και τα μετατρέπει σε αντικείμενα τύπου TaggedWordFormTrainingSample. Αυτά περιέχουν στην ιδιότητα “Word” την λέξη σε συλλαβική μορφή και στην ιδιότητα “Class” τύπου CommandSequenceClass, που είναι αυτό που θα αναγνωρίζουν οι ταξινομητές και είδαμε στην προηγουμένην υποενότητα 3.19.3, δηλαδή την γραμματική σήμανση μαζί με τις εντολές μετατροπής τής μορφής στο λήμμα της, όλα όπως ορίζει ο σχετικός LanguageProvider. Έπειτα, η κλάση TaggedWordFormTrainer δημιουργεί τούς ταξινομητές τύπου WordClassifier για τα αντίστοιχα CommandSequenceClass, επιλέγοντας παραμέτρους για τον καθένα με n-fold cross validation αν ζητηθεί, και τους συσκευάζει σε αντικείμενο τύπου WordClassifierBank. Ακόμη, αναθέτει σε κάθε χαρακτηριστικό τύπου CommandSequenceClass ένα “feature ID” προς χρήσιν από το δεύτερο στάδιο.

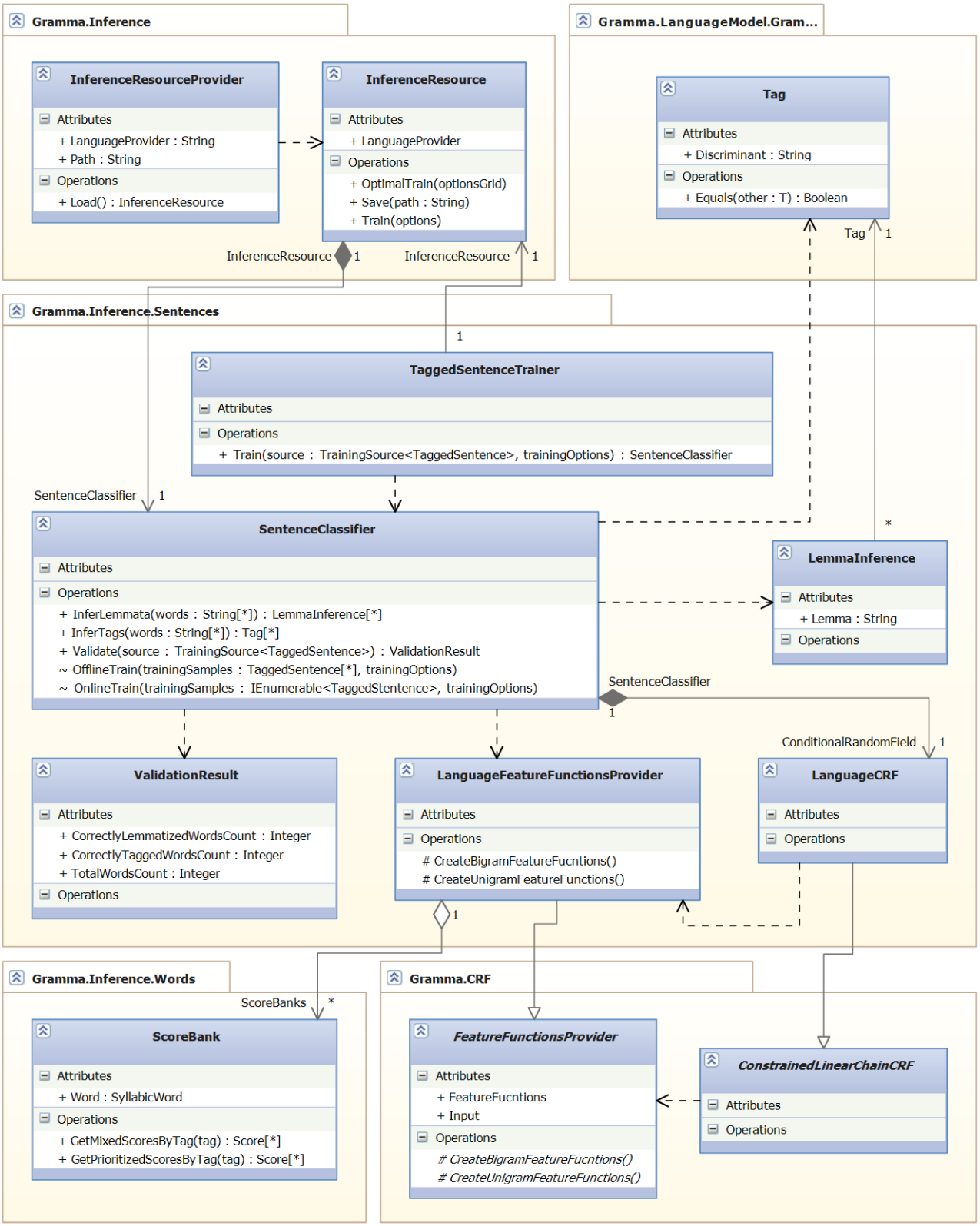
Το αντικείμενο τύπου WordClassifierBank λοιπόν περιέχει ταξινομητές τύπου WordClassifier για κάθε «συχνό» χαρακτηριστικό τύπου CommandSequenceClass μαζί με το “feature ID” του. Τα δύο τελευταία συσκευάζονται στην κλάση WordFeature, στο σχήμα κάτω. Ως «συχνά» χαρακτηριστικα λέγονται αυτά που εμφανίζονται με μία συχνότητα άνω ενός ρυθμιζομένου ποσοστού επί τού πλήθους τών εκπαιδευτικών δεδομένων, ενώ ως «σπανίζοντα» χαρακτηριστικά εννοούμε τα υπόλοιπα. Τα μεν «συχνά» χαρακτηριστικά, εσωτερικά στην κλάση WordClassifier, αναγνωρίζονται με εκπαίδευση ενός SVM με πυρήνα

StringKernel<string> (ίδε υποενότητες 3.13.2 και 3.12.3), δηλαδή με πυρήνα ακολουθιών ή γενικευμένων συμβολοσειρών ή συλλαβολέξεων τών οποίων κάθε στοιχείο είναι συλλαβή παρισταμένη ως string. Τα δε «σπάνια» χαρακτηριστικά αναγνωρίζονται από απλό λεξικό συλλαβολέξεων, επειδή εξ ορισμού δεν διαθέτουν αρκετά θετικά παραδείγματα ώστε να είναι αποδοτική η γενίκευση από ένα SVM ταξινομητή.

Για τα Αρχαία Ελληνικά, ένα τυπικό κατώφλι συχνότητος χαρακτηριστικών είναι το ποσοστό 0,0006% επί συνόλου παραδειγμάτων, το οποίο χωρίζει τα χαρακτηριστικά σε περίπου 10290 «συχνά» και σε περίπου 130000 «σπανίζοντα». Αυτό φανερώνει τις μεγάλες υπολογιστικές απαιτήσεις και την ανάγκη για βελτίστη δυνατή επίδοση τών μονάδων τού έργου: Για κάθε λέξη που εμφανίζεται δουλεύουν 10290 SVM ταξινομητές με πυρήνες γενικευμένων συμβολοσειρών.

Εν τέλει, με την μέθοδο “GetScoreBank” η κλάση WordClassifierBank για μία ζητουμένη συλλαβολέξη συσκευάζει τις τιμές που αποδίδονται στα «συχνά» και τα «σπάνια» χαρακτηριστικά σε ένα αντικείμενο τύπου ScoreBank. Το τελευταίο δηλαδή περιέχει τις δέσμες τών τιμών τών χαρακτηριστικών ως αντικείμενα τύπου Score και οδηγεί το δεύτερο στάδιο, όπως θα δούμε. Μπορεί για μία ζητουμένη σήμανση λέξεως (Tag) να συνθέσει τις σχετικές με αυτήν τιμές χαρακτηριστικών δια τών μεθόδων “GetPrioritizedScoresByTag” (πρώτα σπάνια χαρακτηριστικά αν πληρούνται, έπειτα συχνά) και “GetMixedScoresByTag” (μίξη σπανίων και συχνών χαρακτηριστικών). Οι μέθοδοι αυτές χρησιμοποιούνται από το δεύτερο στάδιο.

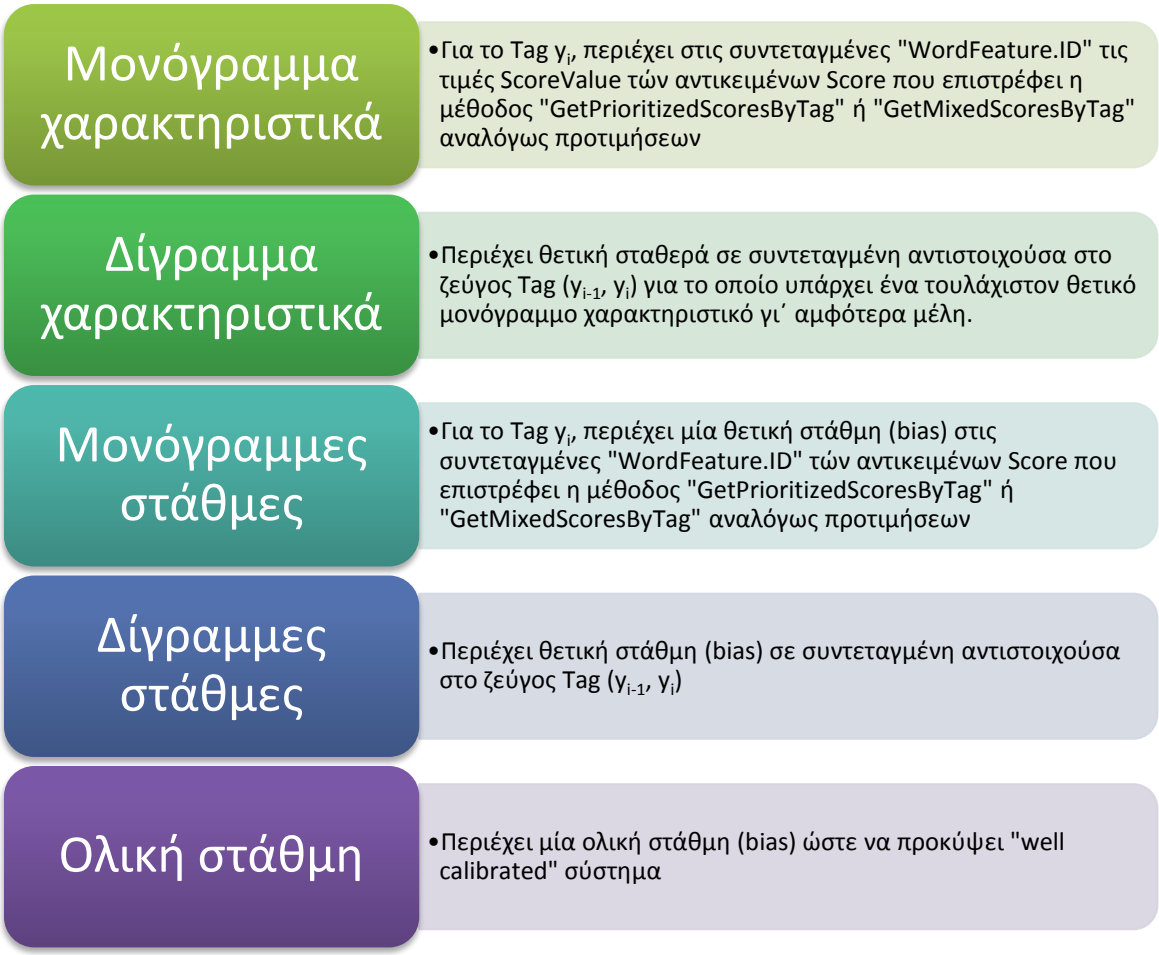
3.19.5 Δεύτερο στάδιο: Το δίκτυο αποτιμήσεως προτάσεων



Σχήμα 3-21: Το σύστημα αποτιμήσεως προτάσεων

Αντιστοίχως με το πρώτο στάδιο, η κλάση InferenceResource κρατεί πόρους για την αποτίμηση προτάσεων στην ιδιότητα “SentenceClassifier” η οποία περιέχει αντικείμενο ομωνύμου κλάσεως, το οποίο είτε φορτώνεται, είτε εκπαιδεύεται με τις μεθόδους “Train” και “OptimalTrain” μέσω τής ιδιωτικής κλάσεως TaggedSentenceTrainer, όπως φαίνεται στο Σχήμα 3-21.

Η κλάση SentenceClassifier βασίζει την λειτουργία της σε μία υλοποίηση τής κλάσεως ContrainedLinearChainCRF<I, Y> που είδαμε στην υποενότητα 3.15.4, όπου ο τύπος εισόδου I είναι string[], δηλαδή μονοδιάστατος πίνακας περιέχων τα συστατικά μιάς προτάσεως, και ο τύπος Y τών στοιχείων τής ακολουθίας εξόδου είναι σήμανση Tag. Η υλοποίηση αυτή λέγεται LanguageCRF και ορίζει τις συνεργαζόμενες συναρτήσεις χαρακτηριστικών να παρέχονται από την κλάση LanguageFeatureFunctionsProvider, η οποία υλοποιεί την αφηρημένη κλάση FeatureFunctionsProvider<string[], Tag> όπως την ζητεί η αφηρημένη μητρική κλάση ConstrainedLinearChainCRF<string[], Tag>.



Σχήμα 3-22: Δομή τού διανύσματος που επιστρέφει η διανυσματική συνάρτηση χαρακτηριστικών $f(x, y_{i-1}, y_i, i)$ όπως ορίζεται από την κλάση LanguageFeatureFunctionsProvider μέσω των αντικειμένων ScoreBank που περιέχει για κάθε συστατικό μιας προτάσεως

Αυτή η κλάση LanguageFeatureFunctionsProvider, η οποία δημιουργείται για κάθε αποτίμηση προτάσεως, κρατεί μέσα της για κάθε στοιχείο τής προτάσεως (λέξεις και στίξεις) αντίστοιχα αντικείμενα τύπου ScoreBank τα οποία είδαμε στην προηγουμένην ενότητα. Βάσει αυτών τών αντικειμένων, η κλάση υπολογίζει την διανυσματική συνάρτηση χαρακτηριστικών $f(x, y_{i-1}, y_i, i)$ που είδαμε στην σχέση (3.15.3) τής ενότητος για τα Linear Chain Conditional Random Fields. Στο Σχήμα 3-22 εικονίζεται συμβολικά η δομή τού διανύσματος που επιστρέφει η συνάρτηση $f(x, y_{i-1}y_i, i)$.

Το πρώτο κομμάτι «μονόγραμμα χαρακτηριστικά» τού διανύσματος, στην αντίστοιχη συντεταγμέννη, περιέχει για τα χαρακτηριστικά WordFeature τών οποίων η ιδιότητα “Class” έχει πεδίο “Tag” ίσο με y_i τις βαθμολογήσεις “ScoreValue” τής λέξεως x_i.Η αντίστοιχη συντεταγμένη αυτή είναι το πεδίο “ID” τού αντικειμένου WordFeature. Οι βαθμολογήσεις Score δίνονται είτε από την μέθοδο“GetPrioritizedScoresByTag” είτε από την “GetMixedScoresByTag” τού αντιστοίχου αντικειμένου ScoreBank. Η επιλογή γίνεται από τις ρυθμίσεις κατά την εκπαίδευση. Η πρώτη δίνει προτεραιότητα στην βαθμολόγηση «σπανίων» χαρακτηριστικών όταν αυτά βρεθούν σε μία λέξη έναντι τών «συχνών», η δεύτερη αναμειγνύει τις βαθμολογήσεις «σπανίων» και «συχνών» χαρακτηριστικών.

Το δεύτερο κομμάτι «δίγραμμα χαρακτηριστικά» περιέχει, σε συντεταγμένη που αντιστοιχεί στο ζεύγος (y_{i-1}, y_i) σημάνσεως Tag, μία θετική τιμή αν υπάρχει για αμφότερα τα (x_{i-1}, x_i) τουλάχιστον μία θετική βαθμολογία «μονογράμμου χαρακτηριστικού» το οποίο υπάγεται στην αντίστοιχο σήμανση.

Το τρίτο και τό τέταρτο κομμάτι είναι επανάληψη τού πρώτου και τού δευτέρου, με την διαφορά ότι την θέση τών βαθμολογιών αντικαθιστούν σταθερές στάθμες ή “biases”. Αυτές, σε συνδυασμό με το τελευταίο κομμάτι πού περιέχει μία συνεχώς παρούσα στάθμη, βοηθούν το πιθανοτικό μοντέλο τού Conditional Random Field που θα προκύψει από την εκπαίδευση να είναι “well calibrated”.

Φθάνουμε λοιπόν εδώ στην κεντρική λειτουργία αυτής τής εργασίας. Απο το Linear Chain Conditional Random Field που προκύπτει, η κλάση SentenceClassifier αμέσως προσφέρει την μέθοδο “InferTags”, η οποία για μία ακολουθία λέξεων και στίξεων που αποτελούν πρόταση επιστρέφει είτε μία ισομήκη ακολουθία πιθανοτέρων σημάνσεων τύπου Tag είτε null ως ένδειξη ότι η πρόταση για την δεδομένη γλώσσα έχει μηδενική πιθανότητα εμφανίσεως. Επάνω στην μέθοδο “InferTags” έχει κτισθεί η μέθοδος “InferLemmata” η οποία μαζί με τις εκτιμώμενες σημάνσεις επιστρέφει και τα εκτιμώμενα λήμματα. Αυτη δουλεύει ως εξής. Καλεί την μέθοδο “InferTags” και για κάθε συλλαβολέξη εισόδου x_i εξετάζει την εκτιμημένη σήμανση y_i. Για να εκτιμήσει και το λήμμα τής λέξεως, εξετάζει ποίο «μονογραμματικό χαρακτηριστικό» της λέξεως x_i συνεισφέρει περισσότερο στην πιθανότητα τής σημάνσεως y_i. Εκ τού μοντέλου πιθανότητος (3.15.4) ενός Linear Chain Conditional Random Field συνάγουμε ότι αυτό είναι:

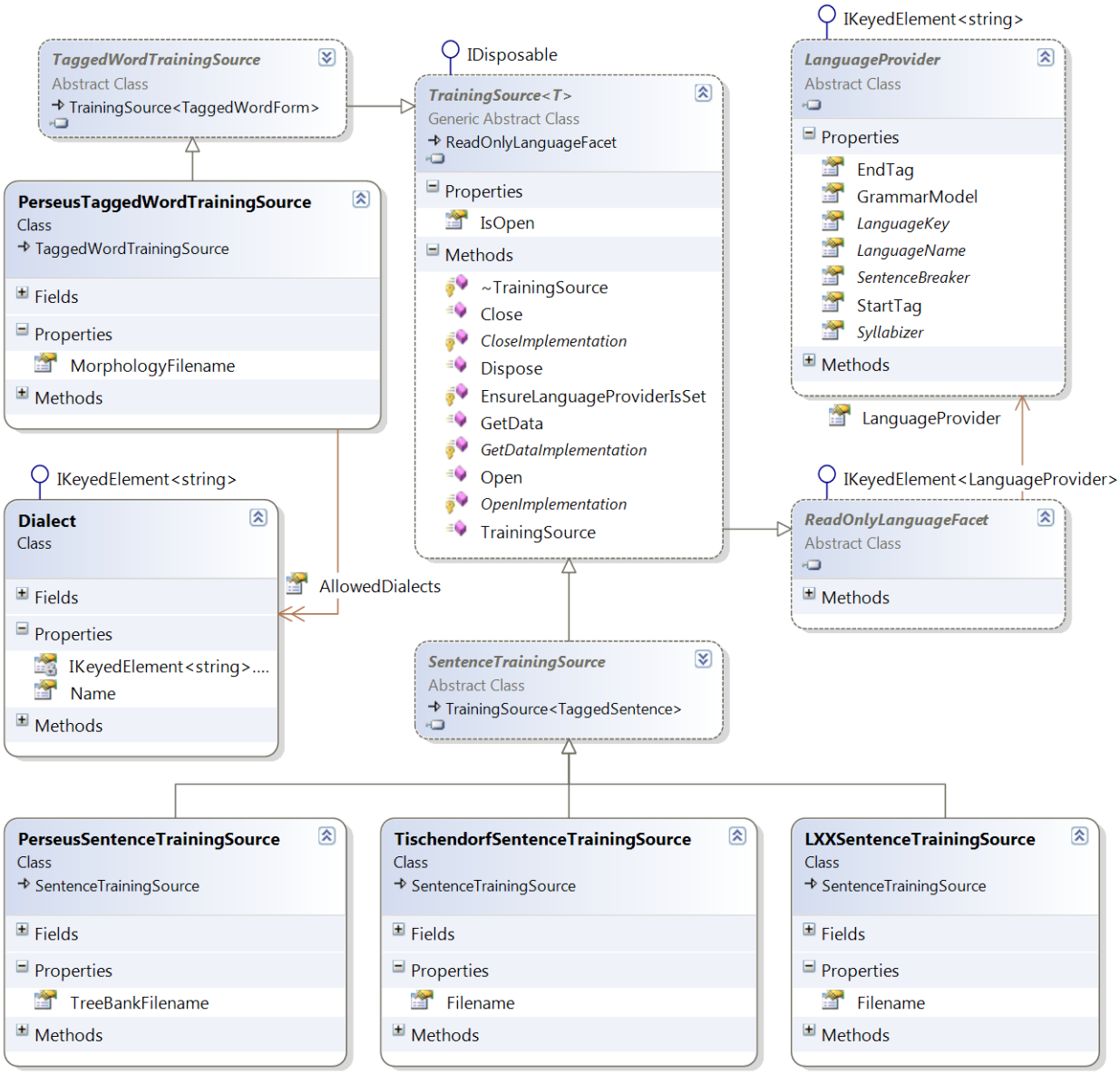
$$j^* = \arg\max_{j \in Q} \left[w_j f_j(x, y_{i-1}, y_i, i) \right] \tag{3.19.1}$$

Στην παραπάνω σχέση, το Q είναι το σύνολο τών αριθμών που αντιστοιχούν στα «μονογραμματικά χαρακτηριστικά», και j^* είναι ο αριθμός τού χαρακτηριστικού. Όπως είδαμε στην σχέση (3.15.5), για την υλοποίηση τού Linear Chain Conditional Random Field η διανυσματική συνάρτηση χαρακτηριστικών αναλύεται στην διγραμματική συνιστώσα h και την μονογραμματική συνισιστώσα q. Επειδή εξ ορισμού το σύνολο Q αφορά αποκλειστικώς μονογραμματικά χαρακτηριστικά, η παραπάνω σχέση (3.19.1) γίνεται:

$$j^* = \arg\max_{j \in Q} \left[w_j q_j(x, y_i, i) \right] \tag{3.19.2}$$

Καθώς οι δείκτες $j \in Q$ είναι ακριβώς τα πεδία “ID” τών αντικειμένων WordFeature στο Σχήμα 3-20, ο δείκτης j^* μέσω τής κλάσεως WordFeature μάς οδηγεί μέσω τής ιδιότητος “Class” στην εκτιμωμένη κλάση ταξινομήσεως CommandSequenceClass, η οποία έπειτα μέσω τής ιδιότητος “Sequence” μάς δίνει τις εντολές που θα μετατρέψουν την συλλαβομορφή x_i στο συλλαβολήμμα. Οι συλλαβές τού λήμματος που προκύπτουν συναρμολογούνται σε κανονικό λήμμα από τον LanguageProvider όπως αυτός παρέχεται από το σχετιζόμενο αντικείμενο InferenceResource.

3.20 Η μονάδα Gramma.LanguageModel.TrainingSources.Greek



Σχήμα 3-23: Με έντονα στοιχεία εικονίζονται οι κλάσεις που εκτίθενται ως πηγές δεδομένων από την μονάδα Gramma.LanguageModel.Greek.TrainingSources

Η μονάδα αυτή προσαρμόζει διάφορες πηγές εκπαιδευτικών δεδομένων για τα Αρχαία Ελληνικά σύμφωνα με τα πρότυπα που έχουν ορισθεί στην υποενότητα 3.19.1. Αυτό το επιτυγχάνει με αντίστοιχες κλάσεις οι οποίες εικονίζονται με έντονα γράμματα στο Σχήμα 3-23. Με αχνά γράμματα εικονίζονται οι κλάσεις που αυτές πρέπει να κληρονομούν ώστε να αποτελούν πηγές δεδομένων, όπως φαίνεται στο Σχήμα 3-16 τής ενότητος 3.19.1 για το σύστημα αντλήσεως εκπαιδευτικών δεδομένων.

Κατ΄ αρχήν η μονάδα προσφέρει πηγή μεμονωμένων σεσημασμένων λέξεων με την κλάση PerseusTaggedWordTrainingSource. Η πηγή αυτή είναι μέρος τού ανοικτού λογισμικού πού προσφέρει το έργο Περσεύς [7] τού πανεπιστημίου Tufts. Είναι ένα μεγάλο XML αρχείο το οποίο περιέχει περίπου 1,5 εκατομμύρια μορφές λέξεων σεσημασμένες και αντιστοιχισμένες στο λήμμα τους. Στην ιδιότητα “MorphologyFilename” τής κλάσεως PerseusTaggedWordTrainingSource δηλώνουμε το XML αρχείο. Με την ιδιότητα “AllowedDialects” μπορούμε να ορίσουμε τις διαλέκτους τής Ελληνικής Γλώσσας για τις οποίες θέλουμε να αντλήσουμε μορφές λέξεων από το αρχείο. Η χρησιμότης τού αρχείου στις εφαρμογές τού Περσέως φαίνεται ότι είναι η δημιουργία συνδέσμων στα εμφανιζόμενα στον αναγνώστη κείμενα από κάθε μορφή λέξεως προς τα διαθέσιμα ηλεκτρονικά λεξικά. Για το παρόν όμως έργο εκτιμήθηκε ότι αυτό το αρχείο είναι μία πρώτης τάξεως εκπαιδευτική πηγή για την μορφολογία και την σήμανση λέξεων τής Αρχαίας Ελληνικής.

Έπειτα, για την άντληση σεσημασμένων εκπαιδευτικών προτάσεων η μονάδα προσφέρει τρία είδη εκπαιδευτικών πηγών με αντίστοιχες κλάσεις. Το πρώτο είδος πηγής είναι τα σεσημασμένα κείμενα (treebanks) τού Περσέως [25] και αναπαρίσταται από την κλάση PerseusSentenceTrainingSource. Το δεύτερο είδος πηγής, αναπαριστάμενο με την κλάση TischendorfSentenceTrainingSource, είναι η Καινή Διαθήκη εκ τού Σιναϊτικού κώδικος που ανεκάλυψε ο Tischendorf, όπως έχει σημανθεί από το έργο MorphGNT [26]. Το τρίτο είδος πηγής είναι η Παλαιά Διαθήκη όπως παράγεται σεσημασμένη από το βοηθητικό πρόγραμμα LXXCombiner το οποίο είδαμε στην υποενότητα 3.18.

4 Εκπαίδευση, αποτίμηση και συμπεράσματα

Στο παρόν κεφάλαιο θά δούμε την εκπαίδευση τού συστήματος, θα αξιολογήσουμε τα αποτελέσματα και θα εξαγάγουμε συμπεράσματα.

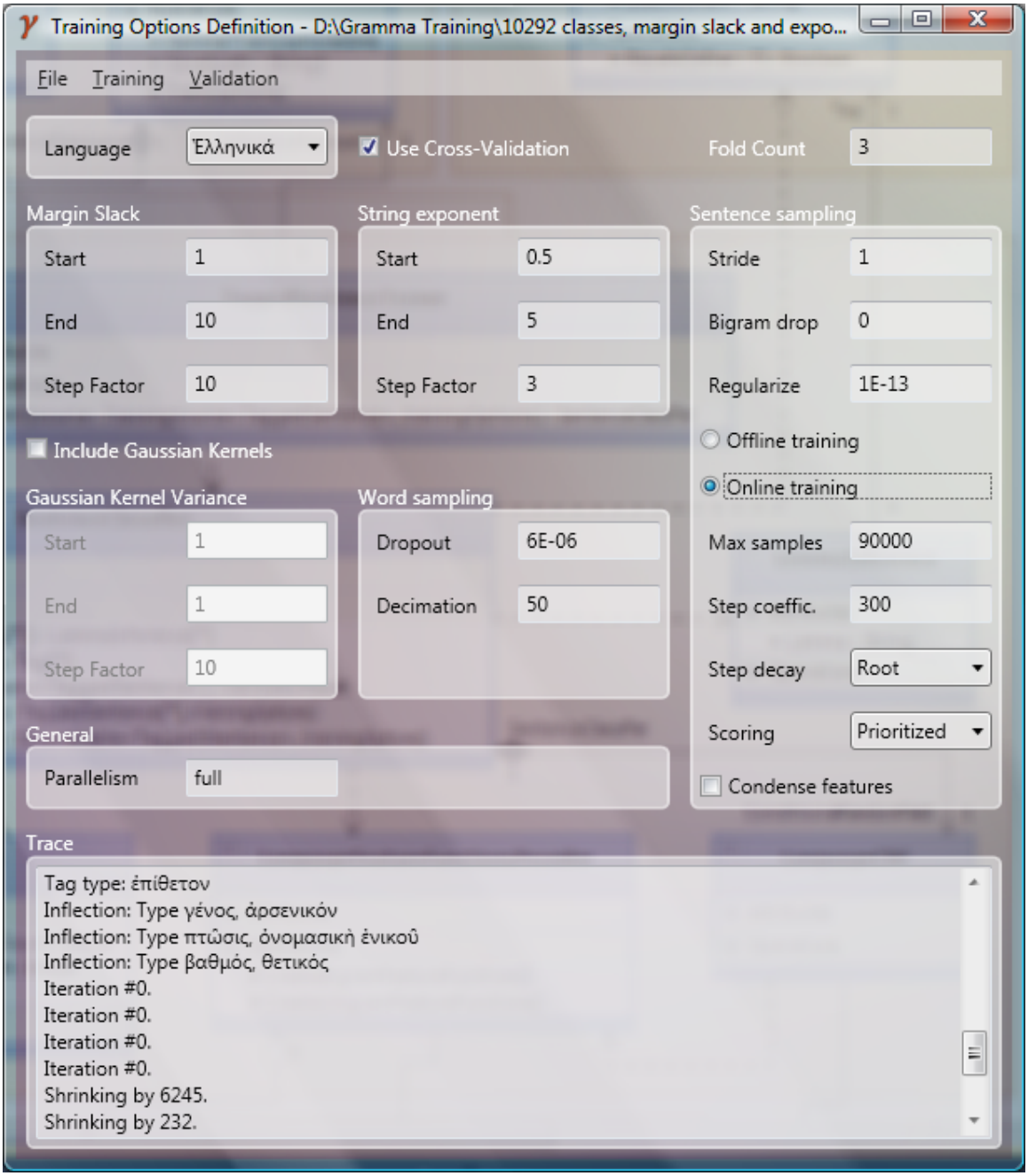
4.1 Τα προγράμματα τού συστήματος

Οι εκτεθείσες στο προηγούμενο κεφάλαιο μονάδες χρησιμοποιούνται από δύο κυρίως προγράμματα προς επαλήθευση και διερεύνηση τού συστήματος. Το πρώτο πρόγραμμα επιτελεί εκπαίδευση και συνολικήν αποτίμηση. Το δεύτερο δείχνει με γραφικό τρόπο την αποτίμηση κειμένων.

Αυτά τα προγράμματα, εκτός από εργαλεία επαληθεύσεως, είναι και παραδείγματα χρήσεως τών μονάδων. Με παρόμοιο τρόπο μπορούν να κτισθούν πραγματικές εφαρμογές. Για παράδειγμα, αν εξασφαλισθούν δικαιώματα κειμένων, μπορεί να εκπονηθεί δικτυακός τόπος Αρχαίων Ελληνικών με ευφυή αναζήτηση πλήρους κειμένου (full text search). Όταν οι λέξεις υφίστανται επεξεργασία, είτε κατά την δεικτοδότηση είτε κατά την αναζήσητη, αντί να αντιστοιχίζονται στην ρίζα τους με κάποιον αλγόριθμο όπως τού Porter για τα Αγγλικά, θα αντιστοιχίζονται στο λήμμα τους από το παρόν σύστημα. Άλλωστε, για μία τόσο πλουσία μορφολογικώς γλώσσα όπως τα Αρχαία Ελληνικα είναι πολύ δύσκολο να βρεθεί αλγοριθμική μέθοδος ευρέσεως λήμματος. Εδώ φαίνεται η χρησιμότης μιας ευφυούς μεθόδου όπως τής παρούσης. Και επειδή το κάθε λήμμα λέξεως εξάγεται με βάση και τα συμφραζόμενα στην πρόταση, το σύστημα θα ευνοεί την αναζήτηση σε φυσική γλώσσα. Βεβαίως, όπως ανεφέρθη στην εισαγωγή, οι μονάδες αυτές μπορούν να γίνουν το πρώτο στρώμα ενός συστήματος οντολογιών, αναζητήσεως εννοιών ή αυτομάτου μεταφράσεως.

Στα επόμενα θα δούμε τα δύο αναφερθέντα προγράμματα εκπαιδεύσεως και αποτιμήσεως.

4.2 Εκπαίδευση με το πρόγραμμα Gramma.TrainingApplication



Σχήμα 4-1: Η διεπαφή του προγράμματος εκπαιδεύσεως όπου φαίνονται οι επιλεγμένες ρυθμίσεις

Με αυτό το πρόγραμμα εκπαιδεύουμε το σύστημα. Στο αρχείο Gramma.TrainingApplication.exe.config έχει δηλωθεί με τον τρόπο που εξετέθη στην υποενότητα 3.19.2 ότι οι δηλώσεις γλωσσών και εκπαιδευτικών πηγών βρίσκονται στο αρχείο Setup.xaml.

4.2.1 Επεξήγηση παραμέτρων και λειτουργίας

Στο Σχήμα 4-1 φαίνονται οι παράμετροι εκπαιδεύσεως. Εν συντομία, έχουν ως εξής.

- **Language:** Επιλογή τής δηλωμένης στο αρχείο Setup.xaml γλώσσας που θα εκπαιδευτεί.
- **Use Cross-Validation και Fold Count:** Εάν το πρώτο είναι επιλεγμένο, ενεργοποιείται το δεύτερο και ορίζει ότι καθείς εκ τών ταξινομητών SVM θα εκπαιδευτεί με αυτόνομο N-fold cross-validation διαμερίζοντας τα εκπαιδευτικά δείγματα σε τεμάχια με αριθμό Fold Count. Τότε, ενεργοποιούνται επίσης και όλα τα πεδία “End” και “Step Factor” στις παραμέτρους που θα δούμε στην συνέχεια, ώστε να ορισθεί ένα «πλέγμα» παραμέτρων που θα δοκιμασθεί για κάθε SVM ταξινομητή ώστε να επιλεγούν οι αποδοτικότερες, αλλιώς οι παράμετροι δεν κυμαίνονται και σχηματίζονται μόνο από τα πεδία “Start”. Τα πεδία “Step Factor” ορίζουν το πολλαπλασιαστικό βήμα με το οποίο μεταβάλλεται μία παράμετρος ώσπου να φθάσει ή να περάσει την τιμή “End”. Για κάθε περίπτωση παραμέτρων στο «πλέγμα» γίνονται “Fold Count” τον αριθμό δοκιμαστικές εκπαιδεύσεις όπου για κάθε μία κρατείται ένα τεμάχιο δεδομένων εκτός εκπαιδεύσεως, ώστε να αποτιμηθεί επ΄ αυτού η απόδοση τού ταξινομητού. Έπειτα, για κάθε συνδυασμό παραμέτρων τού πλέγματος λαμβάνεται ο μέσος όρος τής αποδόσεως όλων τών διαμερίσεων “folds”, οπότε επιλέγεται ο συνδυασμός με την υψηλότερη απόδοση. Με αυτόν τον βέλτιστο δυνδυασμό τελικά γίνεται εκπαίδευση εφ΄ όλων τών εκπαιδευτικών δεδομένων.
- **Παράμετροι πρώτου σταδίου (εκπαίδευση χαρακτηριστικών τών λέξεων)**
    - **Margin Slack:** Είναι η γνωστή παράμετρος C κατά την εκπαίδευση τών SVM, η οποία είναι ένα μέτρο «ποινής» για τα εκπαιδευτικά δείγματα που διαπερνούν το περιθώριο τού ταξινομητού όπως είδαμε στην υποενότητα 3.14.1.
    - **String Exponent:** Είναι η παράμετρος λ τού πυρήνος συλλαβικών ακολουθιών, όπως ορίσθηκε στην σχέση (3.12.8). Τιμές μεγαλύτερες τού 1 τείνουν να δίνουν μεγαλύτερες τιμές πυρήνος όταν οι δύο ακολουθίες έχουν μεγάλα μήκη κοινών υπακολουθιών.
    - **Include Gaussian Kernels και Gaussian Kernel Variance:** Όταν το πρώτο είναι ενεργό, τότε η συνάρτηση απεικονίσεως (3.12.7) η οποία υπονοείται στον πυρήνα ακολουθιών απεικονίζεται περαιτέρω στο γκαουσιανό πεδίο, με τον μετασχηματισμό τού πυρήνος όπως είδαμε στην σχέση (3.13.12), όπου το το σ^2 είναι η παράμετρος “Gaussian Kernel Variance”. Με το “Cross Validation” ανενεργό, όλοι οι πυρήνες λαμβάνουν την απεικόνιση, αλλιώς εκτελείται cross-validation με και χωρίς την επιλογή και εκλέγεται η διαρρύθμιση με το καλύτερο αποτέλεσμα. Αυτό είναι ένα ακριβό υπολογιστικώς χαρακτηριστικό, και τα πειραματικά αποτελέσματα δεν έδειξαν ουσιώδη βελτίωση στην τελική ακρίβεια, συνεπώς καλύτερα είναι να μη ενεργοποιείται.
    - **Word Sampling:** Η ομάδα αυτή ορίζει την δειγματοληψία από τις πηγές σεσημασμένων μεμονωμένων λέξεων προς την εκπαίδευση τών ταξινομητών οι οποίοι αναγνωρίζουν χαρακτηριστικά λέξεων. Ακολουθεί εξήγηση τών συγκεκριμένων πεδίων τής ομάδος.

- **Dropout:** Η τιμή αυτή είναι το κατώφλι συχνότητος εμφανίσεως ενός χαρακτηριστικού στα εκπαιδευτικά δεδομένα κάτω από το οποίο θεωρείται «σπάνιο», αλλιώς θεωρείται «συχνό».Καθώς έχουμε δεί στην υποενότητα 3.19.4, τα «συχνά» χαρακτηριστικά αναγνωρίζονται από ταξινομητές SVM ενώ τα «σπάνια» από ένα ευρετήριο λέξεων σπανίων χαρακτηριστικών.
- **Decimation:** Όπως ελέχθη, οι εκπαιδευτικές πηγές σεσημασμένων λέξεων για τα αρχαία ελληνικά προσφέρουν 1,5 εκατομμύρια δειγμάτων. Η εκπαίδευση τών ταξινομητών SVM, τών οποίων ο αριθμός είναι τής τάξεως τών 10290 αφού αναγνωρίζουν τα αντίστοιχα «συχνά» χαρακτηριστικά, γίνεται ασύμφορος όταν τελείται επί 1,5 εκατομμυρίου δειγμάτων. Γι΄ αυτόν τον λόγο, με την τιμή αυτή αποδεκατίζονται τα «αρνητικά» παραδείγματα με την τιμή αυτού τού πεδίου κατά την εκπαίδευση εκάστου ταξινομητού, ενώ διατηρούνται όλα τα «θετικά». Η πολιτική αυτή είναι εφικτή χωρίς ουσιαστική απώλεια ακριβείας γιατί τα αρνητικά δείγματα είναι πολλές τάξεις περισσότερα από τα θετικά. Με τιμή 50, ο αποδεκατισμός καταλήγει να αφήνει περίπου 30 χιλιάδες δείγματα εκπαιδεύσεως ανά ταξινομητή.

- **Sentence Sampling: Παράμετροι δευτέρου σταδίου (εκπαίδευση Μαρκοβιανού δικτύου προτάσεων)**
  - **Stride**: Οδηγεί το σύστημα να αποδεκατίζει και να περιλαμβάνει παραδείγματα προτάσεων από τις πηγές μόνο ανά τόσα δείγματα, παραλείποντας τα υπόλοιπα (modulo). Η τιμή 1 έχει αποτέλεσμα να περιλαμβάνει όλα τα παραδείγματα τών πηγών. Τιμές άνω τού 1 είναι χρήσιμες μόνο για δοκιμαστικές εκπαιδεύσεις ώστε να μη διαρκούν πολύ, ειδικά όταν η επομένη επιλογή “Offline Training” είναι ενεργός.
  - **Bigram Drop:** Είναι η συχνότης τών δυνατών ζευγών γειτονικών σημάνσεων λέξεων (Tag) στις εκπαιδευτικές πηγές κάτω από την οποία θα θεωρηθούν αυτά ως «σπάνια». Τα σπάνια ζεύγη αφαιρούνται από το σχήμα αραιότητος του δικτύου προτάσεων. Η ρύθμιση δημιουργήθηκε ώστε να ελαφρυνθεί το κόστος εκπαιδεύσεως και αποτιμήσεως το οποίο κατ΄ αρχήν είναι τετραγωνικώς ανάλογο με το πλήθος τών δυνατών σημάνσεων, εις βάρος τής ακριβείας. Αλλά οι τελευταίες βελτιώσεις επιδόσεων στο δίκτυο προτάσεων κατά την εκπόνηση τού έργου κατέστησαν δυνατή την εκπαίδευση να περιλαμβάνει όλα τα ζεύγη όπως προκύπτουν από τις εκπαιδευτικές πηγές. Συνεπώς η προτεινομένη τιμή είναι 0.
  - **Regularize:** Για καλύτερη γενίκευση και αποφυγή υπερπροσαρμογής (overfitting), θεωρούμε ότι οι παράμετροι w τού Linear Chain Conditional Random Field τού δικτύπου προτάσεων έχουν μία προτέρα πιθανότητα $p(w)$ η οποία ακολουθεί μία ομότροπο Γκαουσιανή κατανομή μέ διασπορά αντιστρόφως ανάλογο αυτού τού αριθμού.

- **Offline Training:** Εάν επιλεγεί, ορίζει ότι ο αλγόριθμος εκπαιδεύσεως τού Linear Chain Conditional Random Field είναι η μέθοδος συζυγούς κλίσεως με αναζήτηση σε ευθεία.
- **Online Training:** Εάν επιλεγεί, ορίζει ότι ο αλγόριθμος εκπαιδεύσεως τού Linear Chain Conditional Random Field είναι η στοχαστική κατάβαση κλίσεως, οπότε ενεργοποιούνται και οι επόμενες τρείς παράμετροι σχετικές με τον αλγόριθμο.
    - **Max Samples:** Ο μέγιστος αριθμός δειγμάτων προτάσεων που θα αντληθούν από τις εκπαιδευτικές πηγές. Ο αριθμός αυτός δεν επηρεάζει από ποία πηγή θα ληφούν τα δείγματα διότι κάθε δείγμα λαμβάνεται τυχαία από όλο το εύρος τών πηγών και όχι με την σειρά αντλήσεως.
    - **Step Decay και Step coefficient:** Η στοχαστική κατάβαση κλίσεως είναι ένα είδος στοχαστικής ανελίξεως. Η παράμετροι αυτές είναι η συνάρτηση μειώσεως βάρους $\beta(n)$ κάθε n-οστού δείγματος και η κλιμάκωσή της αντιστοίχως. Όταν η παράμετρος “Step Decay” έχει την τιμή “Linear” τότε η συνάρτηση βάρους είναι $\beta(n) = 1/n$, για την οποία έχει βρεθεί ότι είναι καλό να συνοδεύεται από τιμή “Step Coefficient” τής τάξεως τής παραμέτρου “Max Samples”. Όταν η παράμετρος “Step Decay” έχει την τιμή “Root” τότε η συνάρτηση βάρους είναι $\beta(n) = 1/\sqrt{n}$, οπότε έχει βρεθεί ότι συμφέρει να είναι η παράμετρος “Step Coefficient” τής τάξεως τής τετραγωνικής ρίζας τής παραμέτρου “Max Samples”.
- **Scoring:** Είναι ο τρόπος βαθμολογίας τών χαρακτηριστικών που θα ζητεί το δεύτερο στάδιο για κάθε λέξη. Οι δυνατοί τρόποι έχουν περιγραφεί στην υποενότητα 3.19.4. Η τιμή “Prioritized” προορίζεται όταν τα εκπαιδευτικά δείγματα μεμονωμένων σεσημασμένων λέξεων είναι πολύ εκτενή, αλλιώς προτείνεται η τιμή “Mixed”. Οι υπάρχουσες πηγές Αρχαίων Ελληνικών προσφέρουν 1,5 εκατομμύρια σεσημασμένες λέξεις, οπότε έχει βρεθεί ότι η απόδοση τού συστήματος είναι κάπως καλύτερη όταν το πεδίο έχει τεθεί ως “Prioritized”.
- **Condense features:** Το διάνυσμα χαρακτηριστικών το οποίο επιστρέφει η διανυσματική συνάρτηση χαρακτηριστικών, όπως εικονίζεται στο Σχήμα 3-22, τυπικώς έχει μεγάλη διάσταση. Για τα Αρχαία Ελληνικά έχει περίπου 400 χιλιάδες στοιχεία. Για αύξηση τής ταχύτητος συγκλίσεως, όταν το πεδίο αυτό είναι ενεργοποιημένο, όλα τα «σπάνια» μονόγραμμα χαρακτηριστικά με κοινή σήμανση (Tag) συμπτύσσονται σε ένα αντίστοιχο χαρακτηριστικό, ώστε για τα Αρχαία Ελληνικά οι διαστάσεις να μειώνονται κατά μία τάξη μεγέθους. Τα πειράματα έδειξαν ότι αυτό επιφέρει μία απώλεια ολίγων μονάδων επί τοις εκατό στην ακρίβεια ως αντάλλαγμα μιας επιταχύνσεως στην εκπαίδευση.

- **Parallelism:** Το πεδίο ορίζει τον βαθμό παραλληλισμού για την εκπαίδευση και τών δύο σταδίων. Όταν αφήνεται κενό τότε αυτομάτως παίρνει την τιμή “full”, η οποία σημαίνει ότι ισούται με τούς διαθεσίμους πυρήνες επεξεργαστών στο σύστημα.

Οι ρυθμίσεις αυτές μπορούν να αποθηκευτούν ή να ανακληθούν με τις συνήθεις επιλογές “Open”, “Save”, “Save As…” στο μενού File.

Η εκπαίδευση γίνεται με τις επιλογές τού μενού “Training”. Η εκπαίδευση όλου τού συστήματος, δηλαδή και τών δύο σταδίων του, γίνεται με την επιλογή “Train All”. Τα δύο στάδια τού συστήματος είναι αλληλένδετα και ως όλον φορτώνονται από τις εφαρμογές, οπότε θεωρητικώς αυτή είναι η μόνη αναγκαία επιλογή. Όμως η εκπαίδευση τού πρώτου σταδίου για 10290 ταξινομητές διαρκεί τυπικώς δύο ημέρες, και πολλές φορές χρειάζεται να δοκιμάσουμε πολλές παραμέτρους για το δεύτερο στάδιο επί τού ιδίου πρώτου σταδίου. Γι’αυτόν τον λόγο, υπάρχουν και οι επιλογές “Train Tagged Word Forms” για την εκπαίδευση τού πρώτου σταδίου μόνο και “Train Tagged Sentences” για τού δευτέρου. Η τελευταία επιλογή ενεργοποιείται μόνον όταν υπάρχει εκπαιδευμένο πρώτο στάδιο στην μνήμη, πράγμα που μπορεί να γίνει είτε από την επιλογή “Train All”, είτε την επιλογή “Train Tagged Word Forms”, είτε με φόρτωση τού πρώτου σταδίου διά μιας εκ τών μεθόδων που θα δούμε.

Η αποθήκευση και ανάκληση τού πλήρως εκπαιδευμένου συστήματος γίνεται με τις επιλογές “Load Whole Inference Resource” και “Save Whole Inference Resource”. Αυτές οι επιλογές φορτώνουν και σώζουν αρχεία με κατάληξη “inference”, τα οποία χρησιμοποιούν στην συνέχεια οι εφαρμογές. Η αποθήκευση και η ανάκληση τού πρώτου σταδίου γίνεται με τις επιλογές “Load Tagged Word Forms Training” και “Save Tagged Word Forms Training”. Η αποθήκευση και η ανάκληση τού δευτέρου σταδίου γίνεται με τις επιλογές “Load Sentences Training” και “Save Sentences Training”. Η χρήση όμως τών τελευταίων θέλει προσοχή, γιατί απαιτεί να προϋπάρχει πρώτο στάδιο το οποίο να δίνει τα ίδια χαρακτηριστικά με αυτό που υπήρχε όταν γινόταν η εκπαίδευση τού δευτέρου σταδίου το οποίο φορτώνεται. Ένας τρόπος να γίνει σφάλμα είναι να φορτωθούν στάδια από διαφορετικές γλώσσες ή πηγές.

Όταν το σύστημα είναι εκπαιδευμένο, στο μενού “Validation” ενεργοποιούνται οι λειτουργίες μαζικής αποτιμήσεως τής αποδόσεως. Με την επιλογή “Validate Configured Words Set” υπολογίζονται τα στατιστικά επιτυχίας τού πρώτου σταδίου στην εκτίμηση χαρακτηριστικών ως προς σεσημασμένα δεδομένα τα οποία έχουν δηλωθεί στην κατάλληλη θέση στο αρχείο Setup.xaml. Όμοια επιλογή είναι η “Validate Words Set…”, με την διαφορά ότι τα προς σεσημασμένα δεδομένα έχουν δηλωθεί σε ένα άλλο αρχείο XAML το οποίο επιλέγουμε. Ανάλογες επιλογές υπάρχουν για τα στατιστικά επιτυχίας τού δευτέρου σταδίου, δηλαδή όλου τού συστήματος, επί σεσημασμένων προτάσεων. Οι επιλογές αυτές είναι οι “Validate Configured Sentences Set” και “Validate a Sentences Set…”. Αυτές εμφανίζουν αναλυτικά στατιστικά για κάθε σεσημασμένη πρόταση και στο τέλος δίνουν συνολικά αποτελέσματα.

4.2.2 Οι πηγές εκπαιδεύσεως για τα Αρχαία Ελληνικά

Είδαμε την προσαρμογή ελληνικών πηγών στην ενότητα 3.20. Μέσω αυτών, δηλώθηκαν στο αρχείο Setup.xaml και αντλήθηκαν οι πηγές που περιγράφονται εν συνεχεία.

Πηγή για σεσημασμένες μεμονωμένες λέξεις είναι κυρίως το αρχείο μορφολογίας που προσφέρει το έργο «Περσεύς» όπως είδαμε. Από αυτό όμως δεχόμεθα μόνο τις λέξεις οι οποίες δεν είναι ιδιαίτερες τής Δωρικής διαλέκτου. Με άλλα λόγια, δεχόμεθα τις λέξεις οι οποίες είναι κοινές στις διαλέκτους, τις λέξεις τής Αττικής, Ιωνικής διαλέκτου καθώς και τις λέξεις σε Επικό ύφος. Ο λόγος που αποκλείσθηκε η Δωρική διάλεκτος είναι γιατί εισάγει πολλή ασάφεια στο σύστημα, ώστε θα ήταν καλύτερο ως να εκπαιδευτεί ως ξεχωριστή γλώσσα. Ως ένα παράδειγμα ασαφείας θα πάρουμε τις μορφές τής μετοχής «βλέπουσα». Η μορφή «βλεπούσας» στις άλλες διαλέκτους είναι αιτιατική πληθυντικού, όμως στην Δωρική είναι και γενική ενικού. Η μορφή «βλέπουσαν» στην Δωρική διάλεκτο δέν είναι μόνο αιτιατική ενικού όπως είναι σε όλες τις άλλες διαλέκτους αλλά είναι και γενική πληθυντικού. Στο αντίστοιχο λοιπόν κομμάτι τού αρχείου Setup.xaml δηλώθηκαν οι αποδεκτές διάλεκτοι:

```
<ts:TrainingSet.TaggedWordTrainingSources>

  <perseus:PerseusTaggedWordTrainingSource
    MorphologyFilename="...\Training sets\Perseus\greek.punctuation.xml" />

  <perseus:PerseusTaggedWordTrainingSource
    MorphologyFilename="...\Training sets\Perseus\hebrew.interjections.xml" />

  <perseus:PerseusTaggedWordTrainingSource
    MorphologyFilename="...\Training sets\Perseus\greek.morph.xml">

    <perseus:PerseusTaggedWordTrainingSource.AllowedDialects>
      <perseus:Dialect Name="attic" />
      <perseus:Dialect Name="ionic" />
      <perseus:Dialect Name="epic" />
    </perseus:PerseusTaggedWordTrainingSource.AllowedDialects>
  </perseus:PerseusTaggedWordTrainingSource>

</ts:TrainingSet.TaggedWordTrainingSources>
```

Με γνώμονα τις παραπάνω διαλέκτους, ως πηγές σεσημασμένων προτάσεων προσαρμόσθηκαν τα εξής κείμενα, παρέχοντα 33414 προτάσεις εν συνόλω:

- Ἰλιὰς καὶ Ὀδύσσεια τοῦ Ὁμήρου
- Εὐθύφρων τοῦ Πλάτωνος
- Παλαιὰ Διαθήκη (πλὴν Μακκαβαίων)
- Καινὴ Διαθήκη (Σιναϊτικὸς κώδιξ ὑπὸ Tischendorf)

4.2.3 Ρυθμίσεις εκπαιδεύσεως για τα Αρχαία Ελληνικά

Η εκπαίδευση την οποία θα αξιολογήσουμε στα επόμενα έγινε με τις ρυθμίσεις που φαίνονται στο Σχήμα 4-1.

Η δεδομένη παράμετρος “Dropout” στην ομάδα “Word Sampling” συντελεί ώστε να επιλεγούν 10292 εκ τών 140 χιλιάδων χαρακτηριστικών τών λέξεων ως «συχνά» και να ανατεθεί η αναγνώρισή τους σε SVM ταξινομητές.

Επίσης, παρατηρούμε ότι ως μέθοδος εκπαιδεύσεως τού δευτέρου σταδίου, δηλαδή τού Conditional Random Field, έχει επιλεγεί η “Online Training”, δηλαδή η στοχαστική κατάβαση κλίσεως. Η αιτία είναι ότι αυτή η μέθοδος, αν και πρώτης τάξεως, συγκλίνει πολύ γρήγορα σε σχέση με την άλλη διαθέσιμη μέθοδο “Offline Training”, δηλαδή την συζυγούς κλίσεως με αναζήτηση σ΄ ευθεία, η οποία είναι δευτέρας τάξεως. Το φαινομενικώς παράδοξο γεγονός αυτό επιβεβαιώνεται και μελετάται διεξοδικώς για τα Conditional Random Fields από τους Vishwanathan, Schraudolph, Schmidt και Murphy [27].

4.2.4 Συνολική αποτίμηση αποδόσεως

Ο μέσος όρος ακριβείας τών 10292 SVM ταξινομητών χαρακτηριστικών τών λέξεων, οι οποίοι συνιστούν το πρώτο στάδιο, είναι 87%, με μέτρο “balanced accuracy” (BAC). Κατ’ αυτό το μέτρο, το 50% διαμερίζεται για τα αρνητικά δείγματα και το 50% για τα θετικά, και είναι το κατάλληλο, επειδή όπως είδαμε τα αρνητικά δείγματα είναι κατά τάξεις μεγέθους περισσότερα τών θετικών. Αν είχαμε περιορισθεί στο απλό μέτρο «σωστών επί συνόλου» χωρίς την παραπάνω διαμέριση, τότε ένας ταξινομητής που απαντά σε όλα «όχι» θα είχε μεν επιτυχία 97%, ελαχίστη δε αξία θα προσέφερε.

Ως σωστή αναγνώριση εκλαμβάνεται κάθε «δικαιολογημένη» αναγνώριση όταν υπάρχουν ασάφειες. Για παράδειγμα, ένας ταξινομητής για την κλάση «οὐδέτερον οὐσιαστικόν, πτῶσις αἰτιατικὴ πληθυντικοῦ, ἀκολουθία ἐντολῶν πρὸς λῆμμα: ἀφαίρεσις ληγούσης» ο οποίος αναγνωρίζει θετικώς την λέξη «πράγματα» εκλαμβάνεται ως σωστός, παρά τα εξ ίσου δυνατά ενδεχόμενα να είναι η λέξη στην ονομαστική ή την κλητική πτώση. Η επίλυση τών ασαφειών είναι δουλειά τού δευτέρου σταδίου.

Το δεύτερο στάδιο έχει ακρίβεια σημάνσεως λέξεων 75% και λημματισμού 84%. Αλλά το ενδιαφέρον είναι ότι μεγάλο μέρος τών λανθασμένων σημάνσεων είναι «δικαιολογημένο» ή «συγγνωστό», γιατί πολλές εξ αυτών είναι διφορούμενες ακόμη και για ένα άνθρωπο ή απαιτούν ανωτέρα γνώση τών εννοιών τού κειμένου. Άλλα σφάλματα είναι «αδικαιολόγητα» ή «μή συγγνωστά» και οφείλονται σε δύο αιτίες, πρώτον ότι η τοπολογία Linear Chain είναι πολύ απλουστευτική για μα γλώσσα όπως τα αρχαία ελληνικά όπου οι εξαρτήσεις τών λέξεων εκτείνονται συχνά σε μεγάλες αποστάσεις, δεύτερον ότι τα διαθέσιμα κείμενα δεν περιέχουν πολλές άγνωστες λέξεις ώστε να εκπαιδευτεί καλύτερα ο μηχανισμός αναγνωρίσεως αγνώστων λέξεων. Ως άγνωστες λέξεις εννοούμε αυτές που δέν υπάρχουν στα δεδομένα εκπαιδεύσεως τού πρώτου σταδίου, δηλαδή τις πηγές μεμονωμένων σεσημασμένων λέξεων. Θα δούμε τις αιτίες αναλυτικά στην επομένην ενότητα όπου θα αξιολογήσουμε συγκεκριμένα αντιπροσωπευτικά παραδείγματα. Εν συνόψει όμως, τα μέν «συγγνωστά» σφάλματα είναι τών εξής κατηγοριών:

- Αστοχία διφορουμένης κλίσεως, για παράδειγμα γένους επιθέτου ή μετοχής, επειδή απαιτείται προτέρα γνώση η κατανόηση τών εννοιών. Επί παραδείγματι, στην φράση «ἡ οὐσία τῶν ὄντων» η μετοχή «ὄντων» θα μπορούσε να είναι γένους είτε αρσενικού είτε ουδετέρου, και μόνο το θεματικό περιβάλλον «φιλοσοφία» καθορίζει ότι είναι ουδέτερο.
- Αστοχία ευρέσεως διφορουμένων μερών τού λόγου, για παράδειγμα όταν έχουμε ουσιαστικοποιημένα επίθετα ή μετοχές, όπως η λέξη «πρεσβύτερος» αλλού είναι ουσιαστικό, αλλού επίθετο θετικού βαθμού και αλλού συγκριτικός βαθμός τού επιθέτου «πρέσβυς». Ομοίως, στις λέξεις τού Περσέως τά «ὄντα» είναι σεσημασμένα καί ως μετοχή τού «εἰμί» και ως ουσιαστικό «ὄντα» στο πλαίσιο τής Φιλοσοφίας. Το σύστημα επιλέγει το λήμμα βάσει μόνο γραμματικών στοιχείων, δηλαδή αν κατά τα συμφραζόμενα η λέξη είναι πιθανότερον να χρησιμοποιείται ως ουσιαστικό. Συχνά επιλέγει ορθώς, όμως για περισσότερα σωστά αποτελέσματα απαιτείται γνώση τών εννοιών.

Τα δε «μη συγγνωστά» σφάλματα είναι τών επομένων κατηγοριών:

- Αστοχία πτώσεως, επειδή τα ουδέτερα ουσιαστικά, επίθετα, μετοχές έχουν κοινούς τύπους ονομαστικής, αιτιατικής και κλητικής, και η γενική τους συμπίπτει με τών αρσενικών, οπότε αν αυτό που τα διευκρινίζει είναι μακριά στην πρόταση και δέν διαδίδεται μέσω Linear Chain στην λέξη λόγω άλλης ασχέτου λέξεως που σκιάζει την μετάδοση (Markov blanket), τότε η απόφαση γίνεται με αφελές κριτήριο συχνότητος (naïve) ως αν οι λέξεις ήταν ανεξάρτητες.
- Αστοχία στην αναγνώριση αγνώστου λέξεως επειδή η κλάση της δεν βρισκόταν συχνά στις εκπαιδευτικές προτάσεις. Το φαινόμενο είναι πιο συχνό στα ρήματα, αφού τα διαθέσιμα κείμενα είχαν λίγα μόνο άγνωστα ρήματα.
- Παράλογος αστοχία στο μέρος τού λόγου επειδή η πρόταση έχει συντακτική δομή πολύ διαφορετική από αυτές τών εκπαιδευτικών κειμένων.
- Αστοχία λόγω εσφαλμένης σημάνσεως στα εκπαιδευτικά δεδομένα.

Επειδή τα «συγγνωστά σφάλματα» δεν απέχουν πολύ από την αλήθεια, σε πολύ μεγάλο βαθμό οδηγούν το σωστό λήμμα. Έτσι εξηγείται ότι ο λημματισμός έχει υψηλότερη απόδοση από την σήμανση. Στὸ επόμενο θα εξετασθούν παραδείγματα.

4.3 Ανάλυση παραδειγμάτων με το πρόγραμμα Gramma.Evaluator

Με το πρόγραμμα Gramma.Evaluator φορτώνουμε ή εισάγουμε κείμενα ώστε να δούμε με γραφικό τρόπο την αποτίμηση τών προτάσεων από το σύστημα.

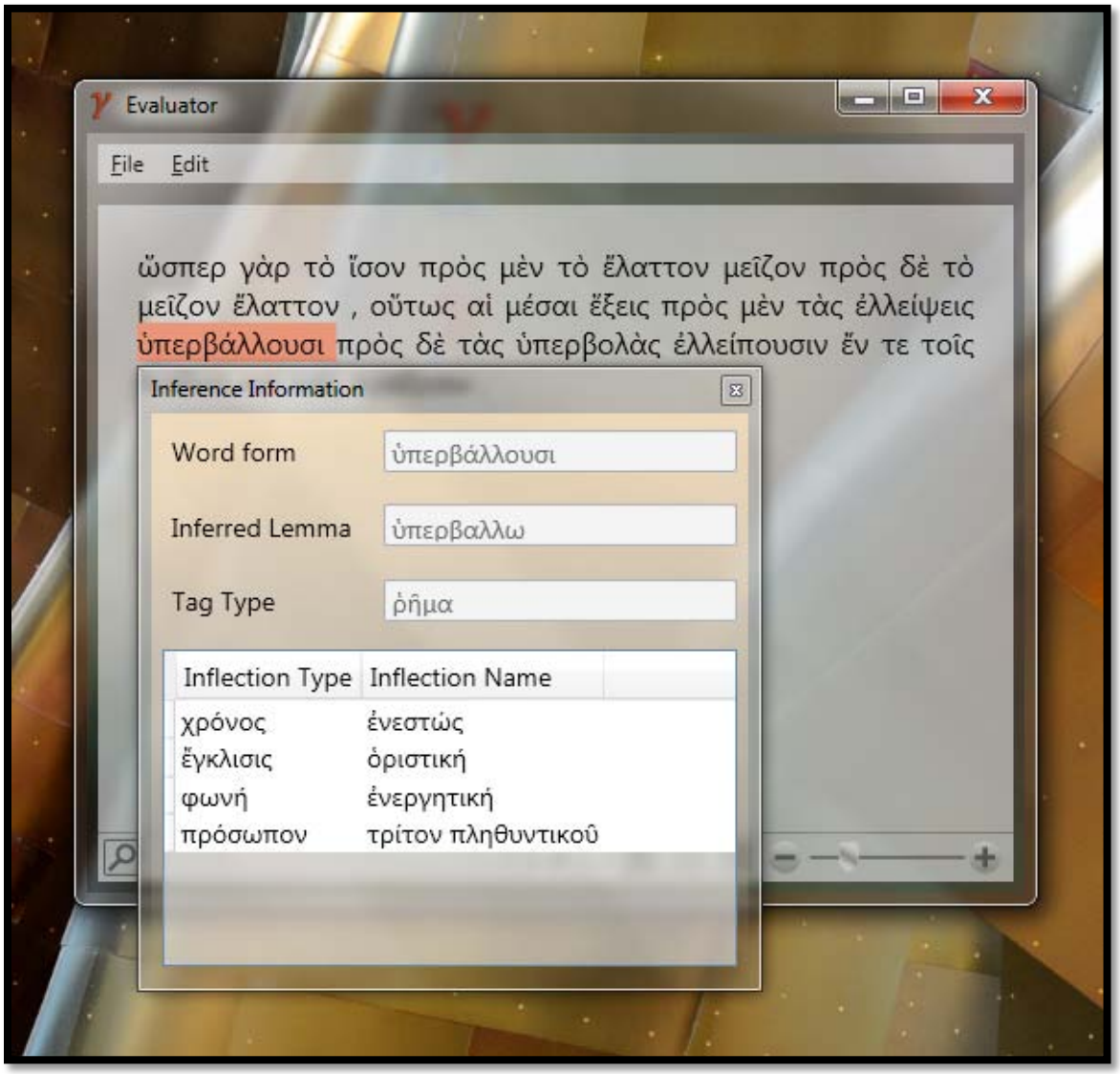
4.3.1 Ρυθμίσεις λειτουργίας

Οι ρυθμίσεις ορίζονται να βρίσκονται στο αρχείο Setup.xaml που συνοδεύει το εκτελέσιμο πρόγραμα, με όμοιο τρόπο όπως στο προηγούμενο πρόγραμμα Gramma.TrainingApplication. Ομοίως λοιπόν δηλώνονται εκεί οι υποστηριζόμενες γλώσσες υπό μορφήν υλοποιήσεων τής κλάσεως LanguageProvider. Αυτό που διαφέρει είναι ότι κάθε μία γλώσσα δέν συνδέεται πλέον

με πηγές εκπαιδεύσεως αλλά με μία αντίστοιχο αποτύπωση εκπαιδευμένων πόρων αποτιμήσεως σε αρχείο με κατάληξη “inference”. Τέτοια αρχεία σώζει το προηγούμενο πρόγραμμα εκπαιδεύσεως με την επιλογή “File → Save Whole Inference Resource”. Όλο αρχείο XAML καταλήγει να είναι το εξής απλό:

```
<Setup
  xmlns="clr-namespace:Gramma.Inference.Configuration;assembly=Gramma.Inference"
  xmlns:gp="clr-
namespace:Gramma.LanguageModel.Greek.Provision;assembly=Gramma.LanguageModel.Greek
.Provision"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  xmlns:i="clr-namespace:Gramma.Inference;assembly=Gramma.Inference">

  <Setup.LanguageProviders>

    <gp:GreekLanguageProvider x:Name="greek" />

  </Setup.LanguageProviders>

  <Setup.InferenceResourceProviders>

    <i:InferenceResourceProvider
      Path="\Gramma Training\10292 classes ... root decay.inference"
      LanguageProvider="{x:Reference greek}" />

  </Setup.InferenceResourceProviders>

</Setup>
```

4.3.2 Διάκριση σημάνσεως μιας λέξεως από τα συμφραζόμενα



Σχήμα 4-2: Ορθή σήμανση τής μορφής «ὑπερβάλλουσι» ως οριστικής εγκλίσεως ρήματος και όχι ως μετοχής

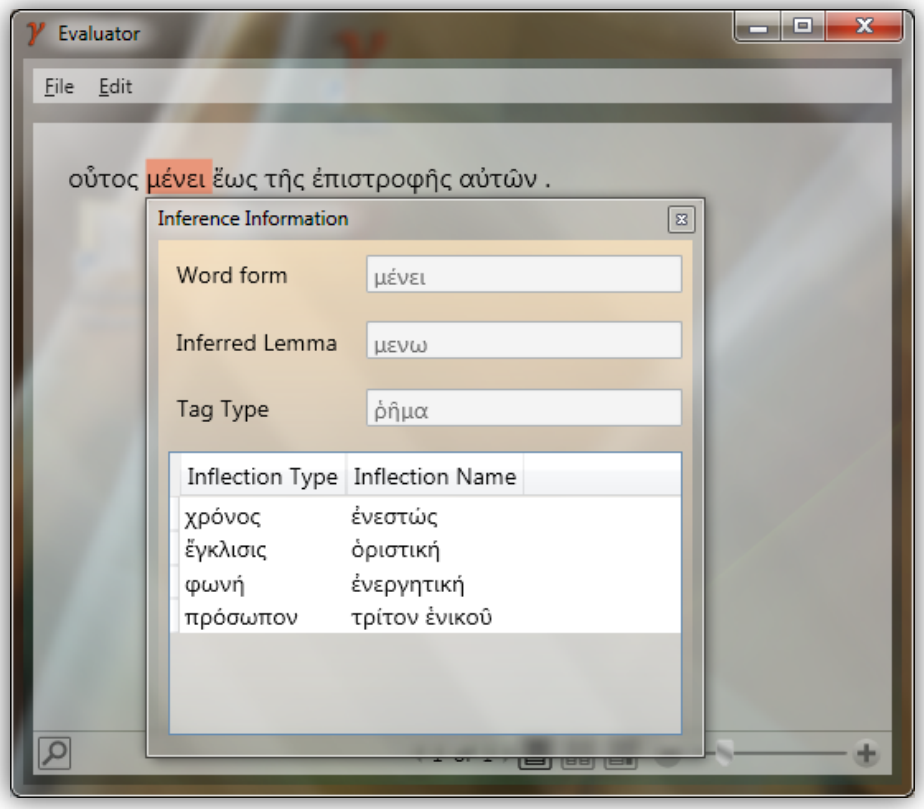
Στο πραγματικό κείμενο[2] στο Σχήμα 4-2, βλέπουμε ότι το σύστημα ορθώς επιλέγει ότι το «ὑπερβάλλουσι» είναι οριστική ρήματος, όχι μετοχή. Ομοίως πράττει και για το «ἐλλείπουσιν».

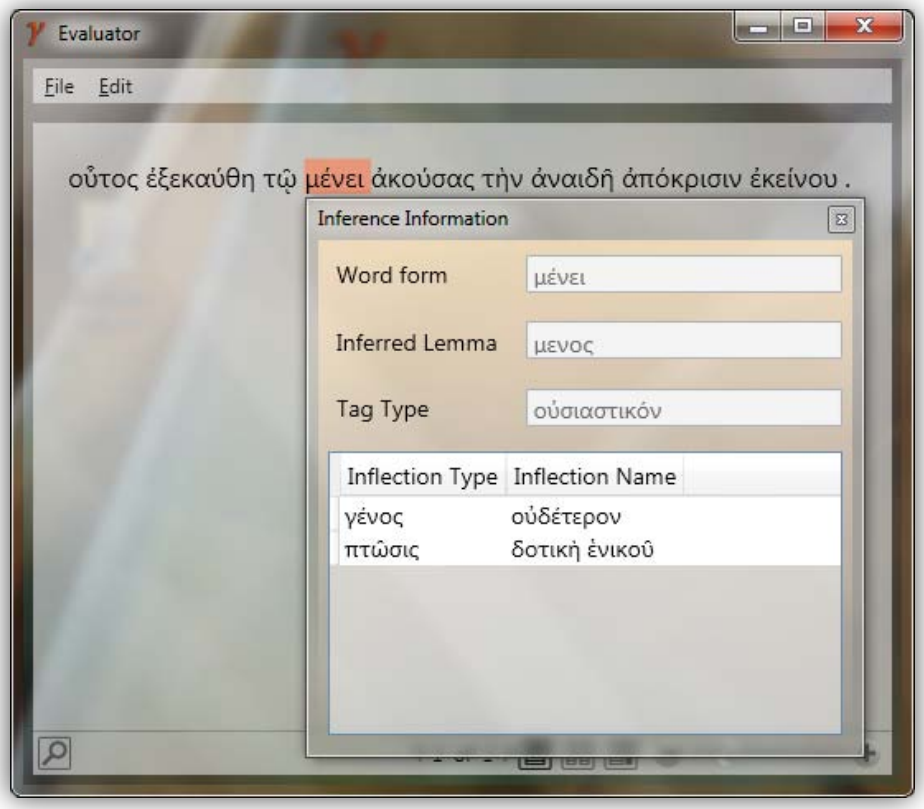
4.3.3 Διάκριση λήμματος από τα συμφραζόμενα

Στο επόμενο Σχήμα 4-3, βλέπουμε το παράδειγμα τής εισαγωγής, δηλαδή πώς το σύστημα διακρίνει το «μένω» από το «μένος» στις δύο προτάσεις που επινοήθηκαν για αυτόν τον σκοπό:

- οὗτος ἐξεκαύθη τῷ μένει ἀκούσας τὴν ἀναιδῆ ἀπόκρισιν ἐκείνου. (μένος)
- οὗτος μένει ἕως τῆς ἐπιστροφῆς αὐτῶν. (μένω)

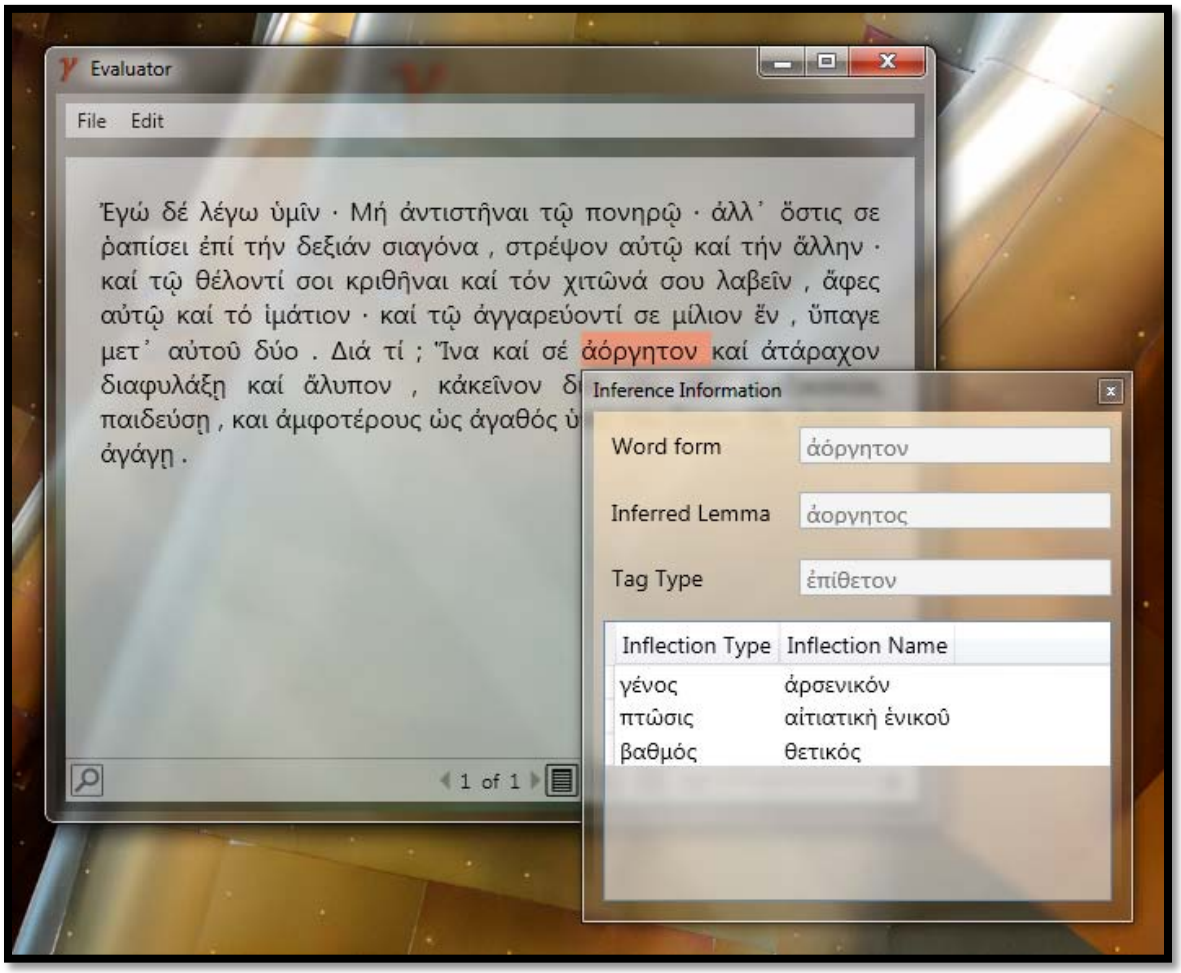
[2] Ἀριστοτέλους Ἠθικὰ Νικομάχεια 1108b 15





Σχήμα 4-3: Διάκριση τών λημμάτων «μένω» και «μένος» με την βοήθεια συμφραζομένων

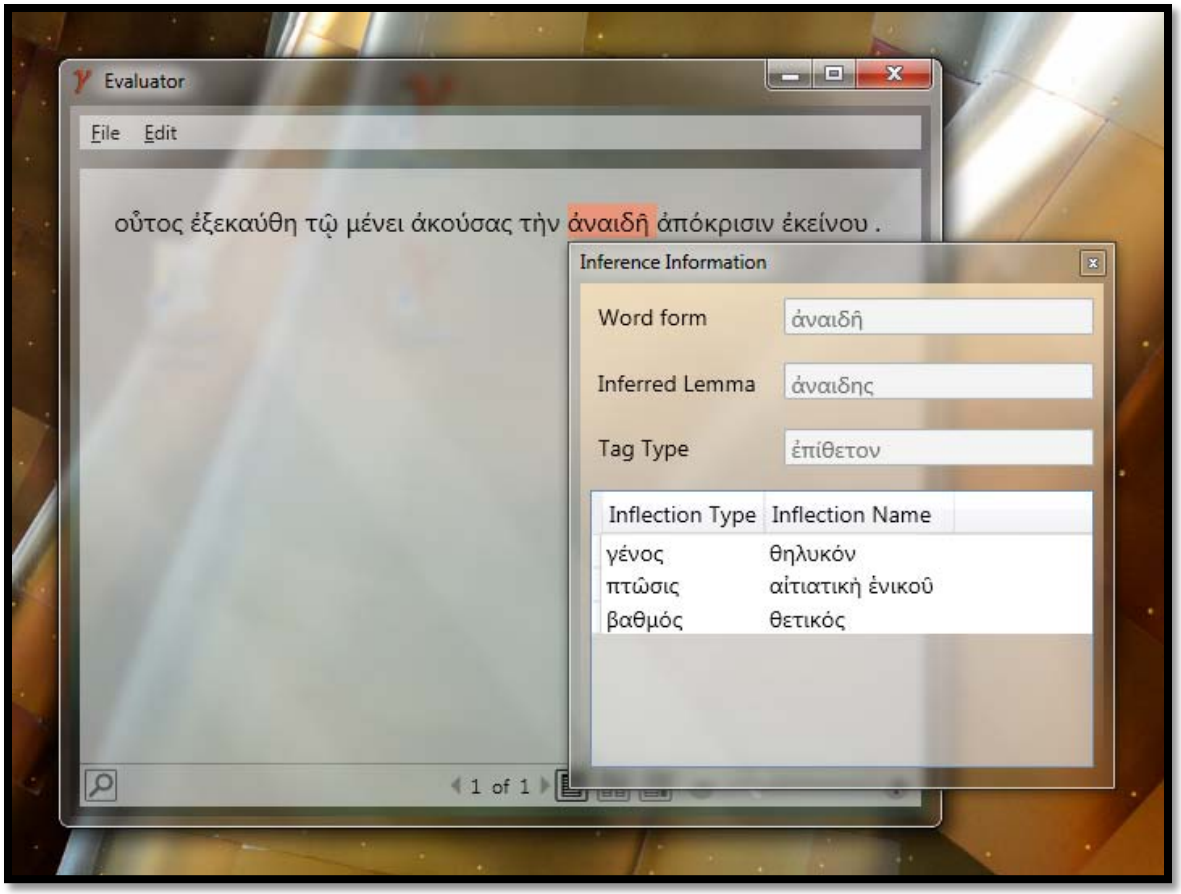
4.3.4 Επιλογή γένους επιθέτων



Σχήμα 4-4: Σωστός συμπερασμός ότι το «ἀόργητον» είναι αρσενικό επίθετο και όχι ουδέτερο

Βλέπουμε στο Σχήμα 4-4 ότι για πραγματικό παράδειγμα κειμένου[3] τού 7^{ου} αιώνος μ.Χ. η μορφή «ἀόργητον» είναι ορθώς σεσημασμένη από το σύστημα ως επίθετο αρσενικού γένους και όχι ουδετέρου, γιατί υποβοηθείται από την παρουσία τού «σέ». Στο ίδιο παράδειγμα μπορούμε να δούμε και την αδυναμία τού συστήματος: Το «ἀτάραχον» έχει σημανθεί ως ουδέτερο. Αυτό οφείλεται στην τοπολογία Linear Chain τού Conditional Random Field όπου η σωστή πληροφορία σημάνσεως τού «ἀόργητον» δεν μπορεί να διαδοθεί μέσω τού ακλίτου συνδέσμου «καί», ο οποίος λειτουργεί ως φράγμα “Markov blanket”, όπως λέγεται στα γραφικά πιθανοτικά μοντέλα.

[3] Μαξίμου Ὁμολογητοῦ Περὶ Ἀγάπης, πρώτη ἑκατοντάς, ξβ’



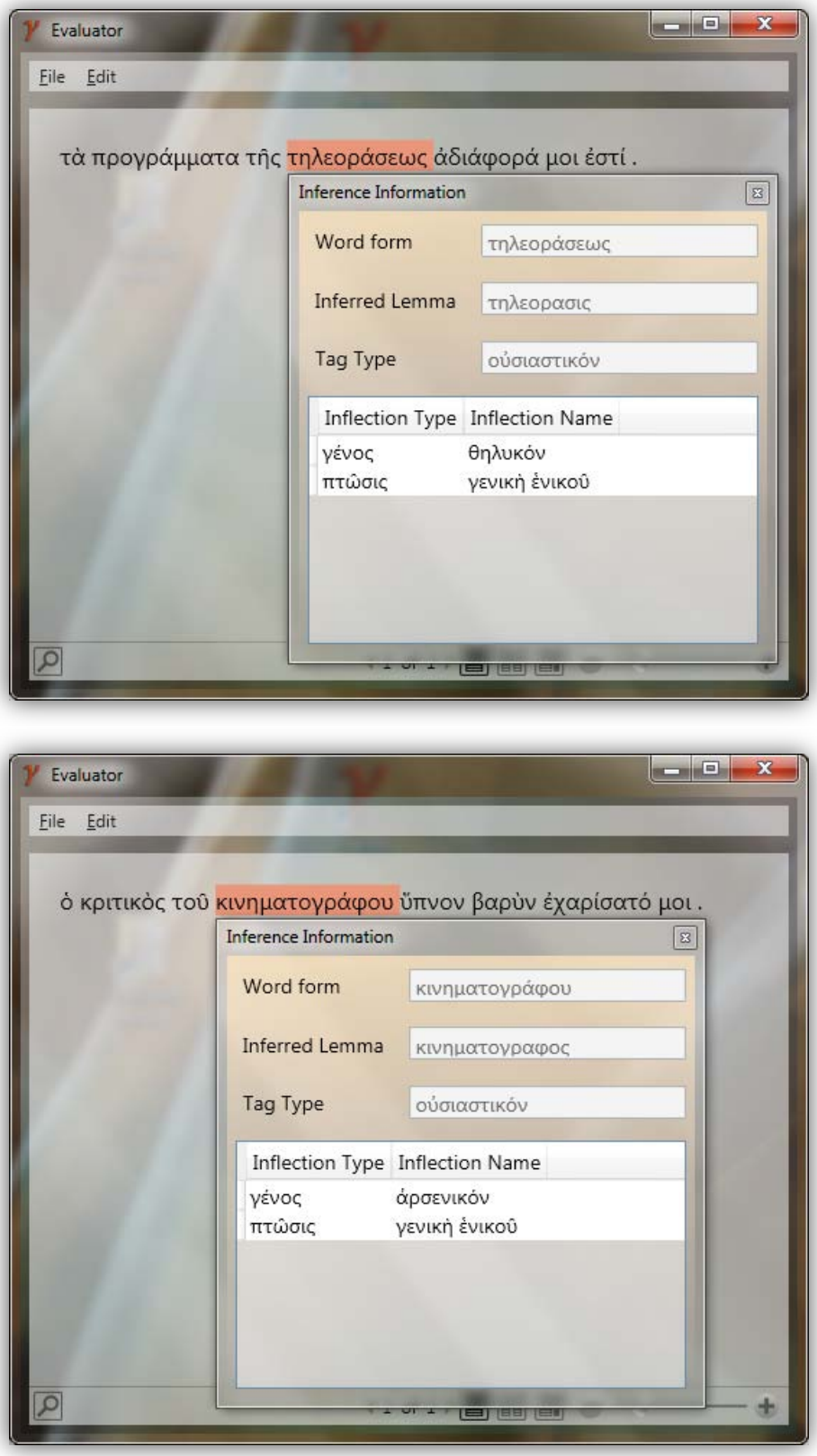
Σχήμα 4-5: Συμφωνία γένους τής μορφής επιθέτου «ἀναιδῆ»

Στο παραπάνω Σχήμα 4-5 έχουμε πάλι το παράδειγμα που είδαμε στο Σχήμα 4-3, όμως εστιάζουμε στην μορφή «ἀναιδῆ». Η μορφή αυτή μόνη της είναι μία περίπτωση ασαφούς κλίσεως. Μπορεί να είναι:

1. αἰτιατικὴ ἑνικοῦ, ἀρσενικόν
2. αἰτιατικὴ ἑνικοῦ, θηλυκόν
3. ὀνομαστικὴ πληθυντικοῦ, οὐδέτερον
4. αἰτιατικὴ πληθυντικοῦ, οὐδέτερον
5. κλητικὴ πληθυντικοῦ, οὐδέτερον

Βλέπουμε όμως ότι το σύστημα βάσει τών συμφραζομένων ορθώς επιλέγει την εκδοχή 2 από τις 5 δυνατές.

4.3.5 Αναγνώριση αγνώστων λέξεων

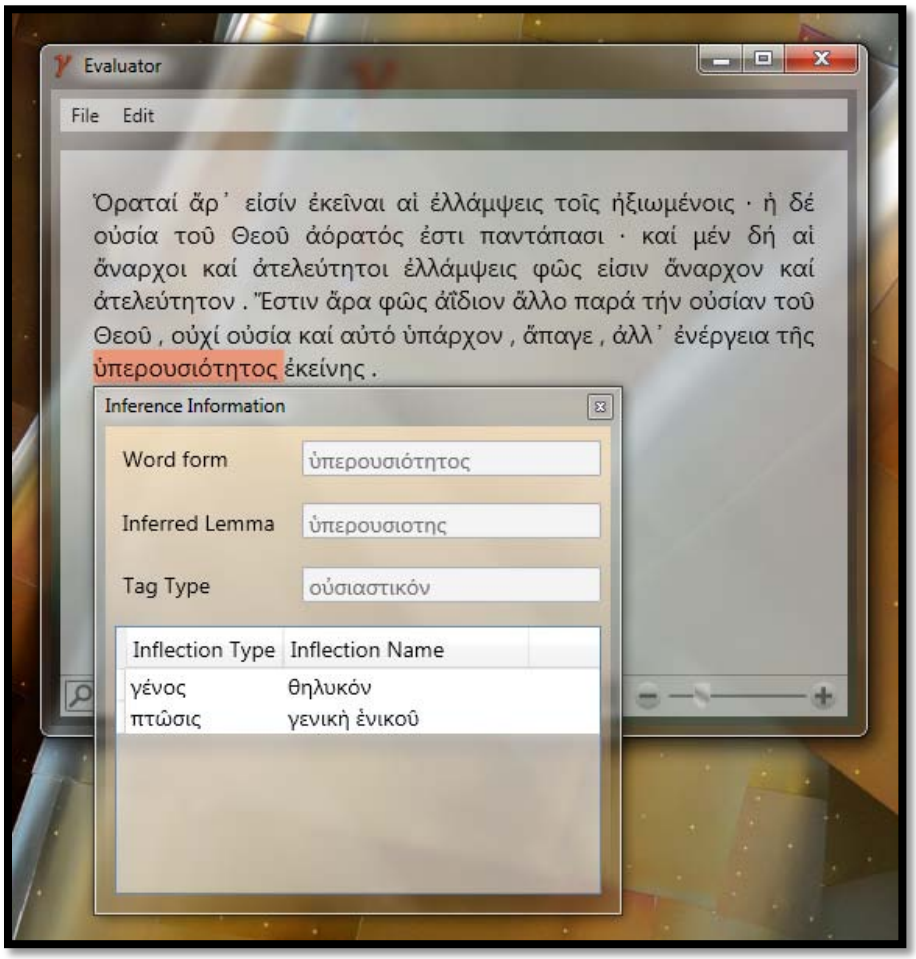


Σχήμα 4-6: Σήμανση και λημματισμός τών λέξεων «τηλεοράσεως» και «κινηματογράφου»

Στο Σχήμα 4-6 έχουμε δύο παραδείγματα προτάσεων με τις προφανώς άγνωστες κατά την αρχαιότητα μορφές «τηλεοράσεως» και «κινηματογράφου». Το σύστημα συνδυάζει με επιτυχία την μορφολογική πληροφορία που εξάγεται από τους ταξινομητές τού πρώτου σταδίου με την ανάλυση συμφραζομένων που επιτελεί το δεύτερο στάδιο ώστε να δώσει σωστή σήμανση και λήμμα.

Ειδικώς η περίπτωση «κινηματογράφου» έχει ενδιαφέρον. Από μόνη της η μορφή θα ήταν πιο πιθανή ως προστακτική ρήματος «κινηματογράφω» παρά γενική ουσιαστικού «κινηματογράφος», γιατί παρόμοια ουσιαστικά δευτέρας κλίσεως όπως «ζῳγράφος» υπάρχουν λίγα, ενώ παρόμοια ρήματα υπάρχουν πάρα πολλά, όπως «ἀντιγράφω», «συγγράφω», «ὑπογράφω», «συνυπογράφω», «ἀναγράφω», «ἀπογράφω», «μεταγράφω» και άλλα. Παρά ταύτα, το δεύτερο στάδιο εξέλεξε λόγω τών συμφραζομένων ορθώς ότι είναι ουσιαστικό δευτέρας κλίσεως.

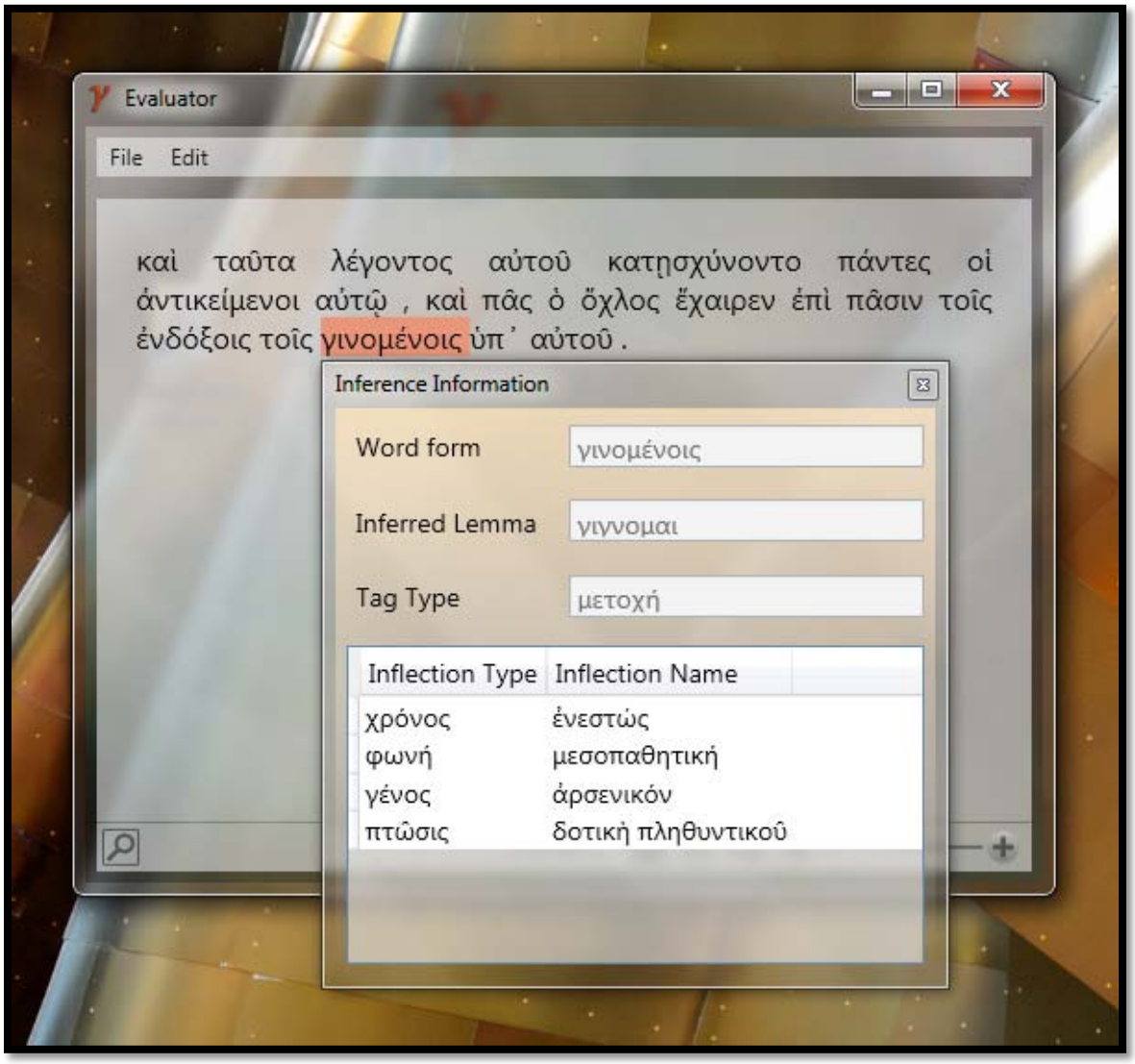
Στο Σχήμα 4-7 βλέπουμε σωστή αναγνώριση αγνώστου λέξεως σε κείμενο[4] τού 1335 μ.Χ.



Σχήμα 4-7: Ορθή αναγνώριση τής αγνώστου λέξεως «υπερουσιότητος»

[4] Γρηγορίου Παλαμᾶ Τριάδες, Λόγος Δεύτερος κατὰ τῶν Δευτέρων

4.3.6 Διφορούμενα γένη



Σχήμα 4-8: «Συγγνωστή» αστοχία γένους, καθώς απαιτείται ανωτέρα γνώση τού εννοιολογικού περιβάλλοντος για τον σωστό καθορισμό της

Με το παράδειγμα στο Σχήμα 4-8 βλέπουμε μία πρόταση[5] η για την οποία το σύστημα είχε πολύ χαμηλή απόδοση 58% στην σήμανση. Η βασική αιτία αστοχίας είναι ότι όλη η φράση «τοῖς ἐνδόξοις τοῖς γινομένοις» αποδόθηκε στο αρσενικό γένος, πράγμα το οποίο δεν είναι λάθος από απόψεως γραμματικής και συνακτικού. Για να αποδοθεί στο σωστό ουδέτερο γένος, χρειάζεται γνώση τών εννοιών. Αν κτιζόταν εννοιολογικό στρώμα επάνω από το σύστημα, αυτό το πρόβλημα θα λυνόταν ίσως με κάποιου είδους ανάδραση πίσω στο σύστημα.

Η άλλη απώλεια που υπάρχει είναι η μορφή «αὐτοῦ», η οποία αποδίδεται ως επίρρημα επειδή στις εκπαιδευτικές προτάσεις είναι πολύ συχνά σεσημασμένη ως τέτοια.

[5] κατὰ Λουκᾶν Εὐαγγέλιον ιγ᾽ 17

4.4 Συμπεράσματα και μελλοντική εργασία

Η διπλωματική εργασία αυτή ήταν μία κατασκευή η οποία εφήρμοσε νέες ιδέες για να επιτύχει ένα δύσκολο αποτέλεσμα. Τα προηγούμενα έδειξαν ότι το επέτυχε ως ένα βαθμό.

Είδαμε ότι τα πολλά σφάλματα είναι «λογικά» ή «συγγνωστά», τα περισσότερα δε εξ αυτών είναι κοντά στην σωστή απάντηση. Αυτή η συμπεριφορά είχε επιδιωχθεί εκ σχεδιασμού, ώστε ακόμη και αν δεν είναι τελεία η σήμανση μιας προτάσεως, να βγάζει κάποιο νόημα και, αν είναι δυνατόν, να οδηγεί στα σωστά λήμματα.

Υπάρχουν όμως και αρκετά σφάλματα τα οποία, εκ της γραματικής και τού συντακτικού μόνον, χωρίς άλλη γνώση, είναι «αδικαιολόγητα». Όπως ανεφέρθη, αυτό οφείλεται κυρίως στην απλουστευτικη τοπολογία “Linear Chain” που έχει το Conditional Random Field το οποίο αποτιμά τά μέρη τής προτάσεως. Αυτή η διαπίστωση μάς δείχνει πώς στο μέλλον μπορεί να βελτιωθεί το παρόν έργο. Μπορούμε να χρησιμοποιήσουμε μίαν επέκταση τής τοπολογίας “Linear Chain” ώστε μία λέξη να έχει και πιό μακρινές εξαρτήσεις εκτός τών γειτονικών της λέξεων. Οι Sutton και McCallum περιγράφουν μία τέτοια επέκταση ονομαζομένη “Skip Chain” [21]. Η υιοθέτηση τής τοπολογίας αυτής στο παρόν έργο εκτιμάται από τον γράφοντα ότι θα έδινε ουσιαστική βελτίωση αποτελεσμάτων. Συνεπώς, αυτό που θα έπρεπε να γίνει είναι να γενικευθούν προς την τοπολογία “Skip Chain” οι μηχανισμοί επιταχύνσεως τού Linear Chain Conditional Random Field οι οποίοι εκπονήθηκαν εδώ για τον αποδοτικό χειρισμό τού μεγάλου πλήθους σημάνσεων που συνεπάγεται μία γλώσσα με πλουσία μορφολογία όπως η Αρχαία Ελληνική.

Ευχαριστώ τους επιβλέποντες καθηγητές κ. Σταφυλοπάτη και κ. Σιόλα για την στήριξη.

5 Βιβλιογραφία

1. Hooker, J.T.: Linear B: an introduction. (1980).

2. Tschach, H.: Syllables and other String Kernel Extensions. (2004).

3. Meijer, E., Beckman, B., Bierman, G.: LINQ. Proceedings of the 2006 ACM SIGMOD international conference on Management of data - SIGMOD ’06. p. 706. ACM Press, New York, New York, USA (2006).

4. GARCIA, R., JARVI, J., LUMSDAINE, A., SIEK, J., WILLCOCK, J.: An extended comparative study of language support for generic programming. J. Funct. Program. 17, 145 (2006).

5. Packard, D.W.: Computer-assisted morphological analysis of ancient Greek. Proceedings of the 5th conference on Computational linguistics -. p. 343. Association for Computational Linguistics, Morristown, NJ, USA (1973).

6. Pantelia, M.: “Noûs, INTO CHAOS”: THE CREATION OF THE THESAURUS OF THE GREEK LANGUAGE. Int. J. Lexicogr. (2000).

7. Smith, D., Rydberg-Cox, J., Crane, G.: The Perseus Project: a digital library for the humanities. Lit. Linguist. Comput. 15, 15–25 (2000).

8. Shewchuk, J.: An introduction to the conjugate gradient method without the agonizing pain. (1994).

9. Bottou, L.: Stochastic gradient descent tricks. Neural Networks: Tricks of the Trade. 1, 1–16 (2012).

10. Navarro, G.: A guided tour to approximate string matching. ACM Comput. Surv. 33, 31–88 (2001).

11. Ukkonen, E.: On-line construction of suffix trees. Algorithmica. 14, 249–260 (1995).

12. Vishwanathan, S., Smola, A.: Fast kernels for string and tree matching. Kernel methods Comput. …. (2004).

13. Vapnik, V.: The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA (1995).

14. Chen, P., Lin, C., Sch, B.: A Tutorial on ν -Support Vector Machines.

15. Osuna, E., Freund, R., Girosi, F.: Training Support Vector Machines: an Application to Face Detection, citeulike-article-id:6694669, (1997).

16. Joachims, T.: Making large scale SVM learning practical. (1999).

17. Platt, J.C.: Fast Training of Support Vector Machines Using Sequential Minimal Optimization. Advances in Kernel Methods - Support Vector Learning. MIT Press (1999).

18. Liossis, E.: Efficient serial and parallel SVM training using coordinate descent. 2013 IEEE Symp. Comput. Intell. Eng. Solut. 76–83 (2013).

19. Mangasarian, O.L., Musicant, D.R.: Successive overrelaxation for support vector machines. IEEE Trans. Neural Netw. 10, 1032–7 (1999).

20. Lin, H., Lin, C., Weng, R.: A note on Platt's probabilistic outputs for support vector machines. Mach. Learn. 1–12 (2007).

21. Sutton, C., McCallum, A.: An introduction to conditional random fields. arXiv Prepr. arXiv1011.4088. (2010).

22. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE. 77, 257–286 (1989).

23. Rahimi, A.: An Erratum for "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," http://alumni.media.mit.edu/~rahimi/rabiner/rabiner-errata/rabiner-errata.html.

24. [MS-XAML-2009]: XAML Object Mapping Specification 2009, http://msdn.microsoft.com/en-us/library/ff629155.aspx.

25. David Bamman, F.M.G.C.: An Ownership Model of Annotation: The Ancient Greek Dependency Treebank.

26. James, T., Sandborg-Petersen, U.: MorphGNT - Linguistic Databases and Python Tools for the Greek New Testament, http://morphgnt.org/.

27. Vishwanathan, S.V.N., Schraudolph, N.N., Schmidt, M.W., Murphy, K.P.: Accelerated training of conditional random fields with stochastic gradient methods. Proc. 23rd Int. Conf. Mach. Learn. - ICML '06. 969–976 (2006).