

Σύστημα Απάντησης Ερωτήσεων
(Σ.Α.Ε.)

Αλέξανδρος Α. Πλέσσας

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΠΕΡΙΛΗΨΗ.....	5
ABSTRACT.....	6
1 ΕΙΣΑΓΩΓΗ.....	7
1.1 Περιγραφή του προβλήματος	7
1.2 Βιβλιογραφική αναφορά σε σχετικές εργασίες	7
1.3 Μέθοδο και τρόπο επίλυσης του προβλήματος	7
1.4 Οργάνωση κειμένου	8
2 ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ – ARTIFICIAL INTELLIGENCE (AI)	9
2.1 Εισαγωγή στην Τεχνητή Νοημοσύνη	9
2.2 Γιατί είναι σημαντική η Τεχνητή Νοημοσύνη.....	10
2.3 Που εφαρμόζεται η Τεχνητή Νοημοσύνη	12
2.4 Πώς ακριβώς λειτουργεί η Τεχνητή Νοημοσύνη	13
2.5 Διαχωρίζοντας την Τεχνητή Νοημοσύνη από τον ενθουσιασμό	14
2.6 Πότε θα έχουμε πραγματικά έξυπνη Τεχνική Νοημοσύνη	15
3 ΕΠΕΞΕΡΓΑΣΙΑ ΦΥΣΙΚΗΣ ΓΛΩΣΣΑΣ – NATURAL LANGUAGE PROCESSING (NLP)	16
3.1 Εισαγωγή στην Επεξεργασία της Φυσικής Γλώσσας	16
3.2 Γιατί είναι σημαντική η Επεξεργασία της Φυσικής Γλώσσας.....	17
3.3 Εργασίες της Επεξεργασία της Φυσικής Γλώσσας.....	17
3.4 Εφαρμογές NLP στην καθημερινότητα μας	19

4	ΣΥΣΤΗΜΑΤΑ ΑΠΑΝΤΗΣΗΣ ΕΡΩΤΗΣΕΩΝ – QUESTION ANSWERING (QA)	20
4.1	Τι είναι ένα σύστημα Απάντησης Ερωτήσεων	20
4.2	Τι είδη ερωτήσεων υπάρχουν	20
4.3	Μέθοδοι Απάντησης Ερωτήσεων	21
4.4	Μερικά συστήματα-μέθοδοι απάντησης ερωτήσεων	22
5	ΕΡΓΑΛΕΙΑ/ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ.....	24
5.1	Γλώσσας προγραμματισμού Python.....	24
5.1.1	Γιατί επιλέγω την Εκδόση Python 3.5	25
5.2	Περιβάλλον ανάπτυξης PyCharm	27
5.3	Βιβλιοθήκη NLTK (Natural Language ToolKit).....	28
5.3.1	Εγκατάσταση του NLTK για Windows	29
5.4	Βιβλιοθήκη Stanford CoreNLP	29
5.4.1	Εγκατάσταση του CoreNLP Server	30
5.5	Βιβλιοθήκη TextBlob.....	33
5.6	Γραφικό περιβάλλον χρήστη tkinder	33
6	Η ΔΟΜΗ ΜΙΑΣ ΕΡΩΤΗΣΕΙΣ (ΑΓΓΛΙΚΗ ΓΡΑΜΜΑΤΙΚΗ)	35
6.1	Οι τύποι ερωτήσεων που υπάρχουν.....	35
6.2	Κλειστές ερωτήσεις (Closed Questions) - Οι ερωτήσεις ‘τύπου’ Ναι / Όχι (Yes/No)	35
6.3	Ανοιχτές ερωτήσεις (Open Questions) - Οι ερωτήσεις ‘τύπου’ Wh – ερωτήσεις	36
6.4	Τι δομή έχει μια ερώτηση	36
7	ΠΩΣ ΧΡΗΣΙΜΟΠΟΙΩ ΤΟ ΣΥΣΤΗΜΑ ΑΠΑΝΤΗΣΗΣ ΕΡΩΤΗΣΕΩΝ (ΣΑΕ)	38
7.1	Αρχική οθόνη προγράμματος.....	38

7.2	Παράδειγμα τρεξίματος του προγράμματος	39
8	Η ΚΑΤΑΣΚΕΥΗ - ΕΞΗΓΗΣΗ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΑΠΑΝΤΗΣΗΣ ΕΡΩΤΗΣΕΩΝ (ΣΑΕ)	42
8.1	Οι Είσοδοι του Συστήματος Απάντησης Ερωτήσεων (ΣΑΕ)	42
8.2	Η βασική προ-επεξεργασία της εισόδου	44
8.3	Απάντηση ερωτήσεων	45
8.3.1	Διαδικές ερωτήσεις (Ναι/Όχι)	46
8.3.2	Ερωτήσεις που έχουν να κάνουν με οντότητες (NERs)	47
8.3.3	Όλες οι υπόλοιπες Ερωτήσεις	49
8.4	Έξοδος του προγράμματος	51
9	ΠΩΣ ΧΡΗΣΙΜΟΠΟΙΩ ΤΟ ΣΥΣΤΗΜΑ ΑΠΑΝΤΗΣΗΣ ΕΡΩΤΗΣΕΩΝ (ΣΑΕ) ΜΕ ΔΕΔΟΜΕΝΑ ΤΟΥ ΧΡΗΣΤΗ	52
9.1	Τροποποίηση του αρχείου εισόδου από τον χρήστη	52
9.2	Τρέξιμο του προγράμματος με δικά μου δεδομένα	53
9.3	Γιατί επιλέγω την τροποποίηση του αρχείου εισόδου από την εισαγωγή μέσω της διεπαφής	53
10	ΜΕΤΡΙΚΕΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ ΑΠΑΝΤΗΣΗΣ ΕΡΩΤΗΣΕΩΝ (ΣΑΕ)	54
10.1	Ταυτότητα δεδομένων - στατιστικά	54
10.2	Τα αποτελέσματα των μετρικών	55
10.3	Ανάλυση κλίμακας των τιμών των μετρικών	55
11	ΣΥΜΠΕΡΑΣΜΑΤΑ - ΜΕΛΛΟΝΤΙΚΕΣ ΚΑΤΕΥΘΥΝΣΕΙΣ	56
11.1	Συμπεράσματα	56
11.2	Μελλοντικές κατευθύνσεις	56
	ΒΙΒΛΙΟΓΡΑΦΙΑ	58

ΠΑΡΑΡΤΗΜΑΤΑ.....	61
ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ	62
ΑΠΟΔΟΣΗ ΟΡΩΝ	63
ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ.....	64

Περίληψη

Στην παρούσα εργασία θα μελετήσουμε την κατασκευή ενός συστήματος απάντηση ερωτήσεων, τα συστήματα απάντησης ερωτήσεων ανέκαθεν ήταν ένας βασικός κλάδος της ΑΙ και του NLP. Το σύστημα μου παίρνει σαν είσοδο ένα αρχείο κειμένου και τις ερωτήσεις (κλειστού και ανοιχτού τύπου) και σαν έξοδο επιστέφει τις απαντήσεις των ερωτήσεων και ένα ποσοστό ακρίβειας-επιτυχίας, εάν μαζί με τις ερωτήσεις έχουν δοθεί και οι σωστές απαντήσεις.

Αρχικά χωρίζω τις ερωτήσεις σε κατηγορίες και εφαρμόζω ανάλογα με την κατηγορία έναν ειδικό αλγόριθμο απάντησης της εκάστοτε ερώτησης. Στην εργασία χρησιμοποίησα Python, NLP τεχνολογίες και γλωσσολογικούς – γραμματικούς κανόνες. Τέλος, έγιναν αρκετές δοκιμές και αναλύσεις σε 97 άρθρα – αρχεία κειμένου και 1910 ερωτήσεις και ο χρήστης έχει την δυνατότητα να δοκιμάσει το σύστημα και με δικά του δεδομένα. Η διεπαφή με τον χρήστη γίνεται με χρήση παραθυριακού περιβάλλοντος.

Λέξεις κλειδιά: σύστημα απάντησης ερωτήσεων, ερωτήσεις κλειστού τύπου, ερωτήσεις ανοιχτού τύπου, δυαδικές ερωτήσεις, καταφατικές ερωτήσεις, αρνητικές ερωτήσεις, γλωσσολογία, NER, NPL, CoreNLP.

Abstract

In this work we will study the construction of a system of answering questions, question answering systems have always been a major branch of AI and NLP. My system takes a text file and questions (closed and open type) as an input, and returns answers to questions as a result and a percentage of accuracy – success, if the correct answers are given along with the questions.

First, I divide the questions into classes and apply a specific answer algorithm for each question according to the category. At postgraduate work I used Python, NLP technologies and linguistic - grammatical rules. Finally, several tests and analyzes were performed in 97 articles - text files and 1910 questions, the user has the possibility to test the system on his own data. The user interface is made using a window environment.

Keywords: question-answering system, closed-ended questions, open-ended questions, binary questions, affirmative questions, negative questions, linguistics, NER, NPL, CoreNLP.

1 Εισαγωγή

1.1 Περιγραφή του προβλήματος

Η μεταπτυχιακή εργασία μου αναφέρεται στην δημιουργία ενός συστήματος το οποίο δύναται να απαντήσει σε ερωτήσεις. Τα συστήματα απάντησης ερωτήσεων ανέκαθεν ήταν ένας βασικός κλάδος της IR και του NLP η οποία έχει να κάνει με συστήματα αυτόματης απάντησης σε ερωτήσεις η οποίες είναι σε ανθρώπινη γλώσσα. Αρχικά ξεκίνησα με την εύρεση ενός corpus το οποίο περιέχει συλλογές από κείμενα και ζευγάρια ερωτήσεων – απαντήσεων, έπειτα δημιούργησα έναν ταξινομητή ερωτήσεων και τέλος την απάντηση των ερωτήσεων ανάλογα με την κατηγορία της με χρήση Python, NLP και γλωσσολογικών – γραμματικών κανόνων.

1.2 Βιβλιογραφική αναφορά σε σχετικές εργασίες

Ενδεικτικά κάποιες από τις πιο σημαντικές και βασικές αναφορές είναι οι: Theorem-proving by resolution as a basis for question-answering systems (Green, 1969) Natural language question-answering systems: 1969 (Simmons, 1970), Overview of the TREC-9 Question Answering Track (Voorhees, 2001) και Question Answering using a large NLP System (Elworthy, 2001)

1.3 Μέθοδο και τρόπο επίλυσης του προβλήματος

Οι μέθοδοι που χρησιμοποίησα για την επίλυση του προβλήματος ήταν η **υπόθεση, ο ολιστικός και ο αναλυτικός τρόπος**.

Η μέθοδος της **υπόθεσης** χρησιμοποιήθηκε για την δημιουργία των αλγόριθμων απάντηση των ερωτήσεων, όπου στηρίζομαι σε υποθέσεις οι οποίες βελτίωναν την απόδοση του συστήματος, ειδικά στον αλγόριθμο απάντησης ερωτήσεων που δεν ήταν κλειστές ή είχαν να κάνουν με οντότητες.

Ο **ολιστικός τρόπος επεξεργασίας** αναφέρεται στην αντίληψη του έργου ή του αντίστοιχου κειμένου ως όλου, δίχως διάκριση των επιμερών στοιχείων ή χαρακτηριστικών του (Foard & Kemler, 1984) όπου στην συγκεκριμένη περίπτωση είναι το σύστημα απάντησης ερωτήσεων γενικά.

Ο **αναλυτικός τρόπος επεξεργασίας** αναφέρεται στην παρατήρηση των λεπτομερειών και στην τελική σύνθεση τους για την συγκρότηση μιας συνολικής αναπαράστασης του έργου ή του αντικειμένου (Foard & Kemler, 1984) δηλαδή στον επιμέρους καταμερισμό των διεργασιών μια προ μια μέχρι την τελική σύνθεση τους και την δημιουργία του τελικού συστήματος μου.

1.4 Οργάνωση κειμένου

Αρχικά γίνεται μια εισαγωγή και απάντηση κάποιων βασικών ερωτημάτων που έχουν να κάνουν με την τεχνική νοημοσύνη στο Κεφάλαιο 2. Στο Κεφάλαιο 3, θα ασχοληθούμε και να απαντήσουμε κάποια βασικά ερωτήματα που έχουν να κάνουν με την επεξεργασία της φυσικής γλώσσας (NLP). Στο Κεφάλαιο 4, θα ασχοληθούμε και θα απαντήσουμε σε κάποια βασικά ερωτήματα που έχουν να κάνουν με τα συστήματα απάντησης ερωτήσεων. Στο Κεφάλαιο 5, θα ασχοληθούμε με τα εργαλεία και τις γλώσσες που χρησιμοποιήσα για την ανάπτυξη του προγράμματος. Στο Κεφάλαιο 6, θα αναφερθούμε στο τι χρειαζόμαστε για την απάντηση μιας ερώτησης δηλαδή να καταλάβουμε τα είδη και την δομή που μπορεί να έχει μια ερώτηση. Στο Κεφάλαιο 7, υπάρχουν οι οδηγίες που πρέπει να ακολουθήσει ο χρήστης για να χρησιμοποιήσει το πρόγραμμα απάντησης ερωτήσεων. Στο Κεφάλαιο 8, περιγράφονται και εξηγούνται οι βασικοί κορμοί του συστήματος απάντησης ερωτήσεων. Στο Κεφάλαιο 9, υπάρχουν οι οδηγίες προς τον χρήστη για να εκτελέσει με δικά του δεδομένα το πρόγραμμα απάντησης ερωτήσεων. Στο Κεφάλαιο 10, αναφέρεται η αξιολόγηση και η ανάλυση της αξιολόγησης του συστήματος. Τέλος, στο Κεφάλαιο 11 υπάρχουν κάποια συμπεράσματα αλλά και κάποιες μελλοντικές κατευθύνσεις της εργασίας.

2 Τεχνητή νοημοσύνη – Artificial Intelligence (AI)

Σε αυτήν την ενότητα θα προσπαθήσουμε να ασχοληθούμε και να απαντήσουμε κάποια βασικά ερωτήματα που έχουν να κάνουν με την τεχνική νοημοσύνη και πιο συγκεκριμένα με τα εξής:

- Εισαγωγή στην Τεχνητή Νοημοσύνη
- Γιατί είναι σημαντική η Τεχνητή Νοημοσύνη
- Που εφαρμόζεται η Τεχνητή Νοημοσύνη
- Πως ακριβώς δουλεύει η Τεχνητή Νοημοσύνη
- Διαχωρίζοντας την Τεχνητή Νοημοσύνη από τον ενθουσιασμό

Στις παρακάτω υποενότητες θα απαντηθούν διεξοδικά, τα παραπάνω ερωτήματα.

2.1 Εισαγωγή στην Τεχνητή Νοημοσύνη

Ο όρος τεχνητή νοημοσύνη αναφέρεται στον κλάδο της πληροφορικής ο οποίος ασχολείται με τη σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων που **μιμούνται στοιχεία της ανθρώπινης συμπεριφοράς** τα οποία υπονοούν έστω και στοιχειώδη ευφυΐα: μάθηση, προσαρμοστικότητα, εξαγωγή συμπερασμάτων, κατανόηση από συμφραζόμενα, επίλυση προβλημάτων κλπ. Ο Τζον Μακάρθι όρισε τον τομέα αυτόν ως «επιστήμη και μεθοδολογία της δημιουργίας νοούντων μηχανών».

Η τεχνητή νοημοσύνη αποτελεί σημείο τομής μεταξύ πολλαπλών επιστημών όπως της πληροφορικής, της ψυχολογίας, της φιλοσοφίας, της νευρολογίας, της γλωσσολογίας και της επιστήμης μηχανικών, με στόχο τη σύνθεση ευφυούς συμπεριφοράς, με στοιχεία συλλογιστικής, μάθησης και προσαρμογής στο περιβάλλον, ενώ συνήθως εφαρμόζεται σε μηχανές ή υπολογιστές ειδικής κατασκευής. Διαιρείται στη **συμβολική τεχνητή νοημοσύνη**, η οποία επιχειρεί να εξομοιώσει την ανθρώπινη νοημοσύνη αλγοριθμικά χρησιμοποιώντας σύμβολα και λογικούς κανόνες υψηλού επιπέδου, και στην **υποσυμβολική τεχνητή νοημοσύνη**, η οποία προσπαθεί να αναπαράγει την ανθρώπινη ευφυΐα χρησιμοποιώντας στοιχειώδη αριθμητικά μοντέλα που συνθέτουν επαγωγικά νοήμονες συμπεριφορές με τη

διαδοχική αυτοοργάνωση απλούστερων δομικών συστατικών («**συμπεριφορική τεχνητή νοημοσύνη**»), προσομοιώνουν πραγματικές βιολογικές διαδικασίες όπως η εξέλιξη των ειδών και η λειτουργία του εγκεφάλου («**υπολογιστική νοημοσύνη**»), ή αποτελούν εφαρμογή στατιστικών μεθοδολογιών σε προβλήματα ΑΙ.

Αρχικά, η έρευνα γύρω από το ΑΙ επικεντρώθηκε σε θέματα όπως η επίλυση προβλημάτων και οι συμβολικές μέθοδοι. Τη **δεκαετία του '60**, το Υπουργείο Άμυνας των ΗΠΑ ενδιαφέρθηκε για αυτόν τον τύπο εργασίας και ξεκίνησε την εκπαίδευση των υπολογιστών στη μίμηση της βασικής ανθρώπινης συλλογιστικής. Για παράδειγμα, η Υπηρεσία Προηγμένων Ερευνητικών Προγραμμάτων Άμυνας (DARPA) ολοκλήρωσε τα προγράμματα χαρτογράφησης δρόμων τη δεκαετία του '70. επίσης, η DARPA παρήγαγε ευφυείς προσωπικούς βοηθούς το 2003, πολύ πριν η Siri, Alexa και Cortana γίνουν πασίγνωστες. Ουσιαστικά από το **1950** ως **1970** **είχαμε ανάπτυξη των Neural Networks** (Haykin, 1994), οι αρχικές εργασίες με neural networks προκαλεί ενθουσιασμό γύρω από τις μηχανές που "σκέφτονται". Από το **1980** ως το **2010** **είχαμε ανάπτυξη των Machine Learning**, το Machine learning αρχίζει να ανθίζει. Ενώ **σήμερα έχουμε το Deep Learning** (LeCun, et al., 2015), οι ανακαλύψεις γύρω από το Deep Learning οδηγούν στην έκρηξη του ΑΙ συμπεριλαμβανομένων των συστημάτων υποστήριξης λήψης αποφάσεων και των έξυπνων συστημάτων αναζήτησης που μπορούν να σχεδιαστούν ώστε να συμπληρώνουν και να βελτιώνουν τις ανθρώπινες ικανότητες.

Ενώ **οι ταινίες του Χόλυγουντ και τα μυθιστορήματα επιστημονικής φαντασίας** απεικονίζουν το ΑΙ ως ανθρωπόμορφα ρομπότ που καταλαμβάνουν τον κόσμο, **η τρέχουσα εξέλιξη του ΑΙ δεν είναι τόσο τρομακτική – ούτε τόσο έξυπνη**. Αντίθετα, το ΑΙ έχει εξελιχθεί ώστε να παρέχει συγκεκριμένα οφέλη σε κάθε βιομηχανικό κλάδο.

2.2 Γιατί είναι σημαντική η Τεχνητή Νοημοσύνη

Το ΑΙ αυτοματοποιεί την επαναληπτική μάθηση και την ανακάλυψη μέσω δεδομένων. Το ΑΙ διαφέρει, όμως, από την καθοδηγούμενη από το hardware ρομποτική αυτοματοποίηση. Αντί των αυτοματοποιημένων χειροκίνητων έργων, το ΑΙ εκτελεί συχνά, μεγάλου όγκου μηχανογραφημένα έργα, αξιόπιστα και χωρίς κόπο. Για αυτόν τον τύπο της

αυτοματοποίησης, η ανθρώπινη έρευνα εξακολουθεί να είναι καίριας σημασίας για να εγκατασταθεί το σύστημα και να τεθούν οι κατάλληλες ερωτήσεις.

Το AI προσθέτει ευφυΐα στα υπάρχοντα προϊόντα. Στις περισσότερες περιπτώσεις, το AI δεν πωλείται ως μεμονωμένη εφαρμογή. Πιθανότερο είναι οι ικανότητες του AI να βελτιώνουν τα προϊόντα που ήδη χρησιμοποιείτε, όπως η πρόσθεση της **ψηφιακής βοηθού Siri** ως λειτουργία στα προϊόντα Apple νέας γενιάς. Η αυτοματοποίηση, οι πλατφόρμες συνομιλίας, τα bots και οι έξυπνες μηχανές μπορούν να συνδυαστούν με μεγάλες ποσότητες δεδομένων προκειμένου να βελτιώνουν πολλές τεχνολογίες στο σπίτι και τον χώρο εργασίας από την ασφάλεια πληροφοριών έως την ανάλυση επενδύσεων.

Το AI προσαρμόζεται μέσω προοδευτικών αλγορίθμων εκμάθησης (learning algorithms) ώστε να αφεθούν τα δεδομένα να κάνουν τον προγραμματισμό. Το AI βρίσκει δομή και κανονικότητες στα δεδομένα οπότε ο **αλγόριθμος αποκτά μια δεξιότητα: Ο αλγόριθμος ταξινομεί ή κατηγοριοποιεί.** Έτσι, όπως ο αλγόριθμος μπορεί να διδάξει τον εαυτό του τον τρόπο να παίζει σκάκι, μπορεί και να τον διδάξει ποιο προϊόν να συστήσει στη συνέχεια διαδικτυακά. Και τα μοντέλα προσαρμόζονται όταν τους δίνονται νέα δεδομένα. Η **οπισθοδιάδοση (back-propagation)** είναι μια τεχνική AI που επιτρέπει στο μοντέλο να προσαρμόζεται, μέσω εκπαίδευσης και προστιθέμενων δεδομένων όταν η πρώτη απάντηση δεν είναι η ενδεδειγμένη.

Το AI αναλύει περισσότερα και βαθύτερα δεδομένα με χρήση neural networks (νευρωνικά δίκτυα) που διαθέτουν πολλά κρυφά επίπεδα. Η κατασκευή ενός συστήματος ανίχνευσης απάτης με πέντε κρυφά επίπεδα ήταν σχεδόν αδύνατη λίγα χρόνια νωρίτερα. Όλα αυτά έχουν αλλάξει με την απίστευτη ισχύ των υπολογιστών και το μέγεθος των δεδομένων. Χρειάζεσαι **μια πληθώρα δεδομένων για να εκπαιδεύσεις μοντέλα μάθησης σε βάθος, επειδή μαθαίνουν απευθείας από τα δεδομένα.** Με όσο περισσότερα δεδομένα τα τροφοδοτείς, τόσο πιο ακριβή γίνονται.

Το AI επιτυγχάνει απίστευτη ακρίβεια μέσω Deep neural networks – κάτι που προηγουμένως ήταν αδύνατο. Για παράδειγμα, οι αλληλεπιδράσεις σας με το **Alexa, το Google Search και το Google Photos** βασίζονται στη στο Deep learning – και συνεχίζονται να γίνονται πιο ακριβείς όσο περισσότερο τα χρησιμοποιείτε. Στον **ιατρικό τομέα**, τεχνικές AI όπως το Deep learning, η ταξινόμηση εικόνων και η αναγνώριση αντικειμένων μπορούν

τόρα να χρησιμοποιηθούν για την ανίχνευση καρκίνου σε απεικονίσεις με μαγνητική τομογραφία, με ακρίβεια παρόμοια με αυτή καταρτισμένων ακτινολόγων.

Το ΑΙ αξιοποιεί στο έπακρο τα δεδομένα. Όταν οι αλγόριθμοι είναι αυτο-εκπαιδευόμενοι, τα ίδια τα δεδομένα μπορούν να καταστούν πνευματική ιδιοκτησία. Οι απαντήσεις βρίσκονται στα δεδομένα, απλά, πρέπει να εφαρμόσετε μεθόδους τεχνητής νοημοσύνης για να τις ανακτήσετε. Επειδή ο ρόλος των δεδομένων είναι πλέον πιο σημαντικός από ποτέ άλλοτε, μπορούν να δημιουργήσουν ένα ανταγωνιστικό πλεονέκτημα. **Εάν διαθέτετε τα καλύτερα δεδομένα σε έναν ανταγωνιστικό βιομηχανικό κλάδο, ακόμη και αν όλοι εφαρμόζουν παρόμοιες τεχνικές, τα καλύτερα δεδομένα θα νικήσουν.**

2.3 Που εφαρμόζεται η Τεχνητή Νοημοσύνη

Φροντίδα υγείας: Οι εφαρμογές του ΑΙ μπορούν να παρέχουν εξατομικευμένη διαχείριση φαρμάκων και αναγνώσεις ακτινογραφιών. Οι προσωπικοί βοηθοί φροντίδας υγείας μπορούν να ενεργήσουν ως σύμβουλοι διαβίωσης, να σας υπενθυμίζουν να παίρνετε τα φάρμακά σας, να ασκείστε ή να τρώτε πιο υγιεινά.

Λιανικό εμπόριο: Το ΑΙ παρέχει εικονικές (virtual) ικανότητες αγορών που προσφέρουν εξατομικευμένες προτάσεις και τη δυνατότητα συζήτησης των επιλογών αγοράς με τον πελάτη. Η διαχείριση αποθεμάτων και η διάταξη του χώρου επίσης βελτιώνονται.

Παραγωγή: Το ΑΙ μπορεί να αναλύει εργοστασιακά δεδομένα μέσα από εφαρμογές διαδικτύου των πραγμάτων (IoT), καθώς τα δεδομένα ρέουν από τον διασυνδεδεμένο εξοπλισμό για να προβλέψουν το αναμενόμενο φορτίο ζήτησης με τη χρήση recurrent neural networks, ενός τύπου Deep Learning Network που χρησιμοποιείται για ακολουθίες δεδομένων.

Αθλητισμός: Η τεχνητή νοημοσύνη χρησιμοποιείται για τη σύλληψη εικόνων αθλητικών παιχνιδιών και για να παρέχει στους προπονητές αναφορές σχετικά με τον τρόπο καλύτερης οργάνωσης του παιχνιδιού, συμπεριλαμβανομένης της καλύτερης τοποθέτησης των παιχτών στο γήπεδο και της στρατηγικής.

2.4 Πώς ακριβώς λειτουργεί η Τεχνητή Νοημοσύνη

Η τεχνητή νοημοσύνη λειτουργεί με συνδυασμό μεγάλων ποσοτήτων δεδομένων με γρήγορους, επαναληπτικής διαδικασίας και ευφυείς αλγορίθμους, επιτρέποντας στο λογισμικό να μαθαίνει αυτόματα από μορφές ή χαρακτηριστικά των δεδομένων. Η τεχνητή νοημοσύνη είναι ένα πεδίο μελέτης που περιλαμβάνει πολλές θεωρίες, μεθόδους και τεχνολογίες, καθώς και τα παρακάτω κύρια υπο-επίπεδα:

Machine learning: αυτοματοποιεί την κατασκευή αναλυτικών μοντέλων. Χρησιμοποιεί μεθόδους από τα νευρωνικά δίκτυα (neural networks), τη στατιστική, την επιχειρησιακή έρευνα (operational research) και τη φυσική για την εύρεση κρυφών γνώσεων εντός των δεδομένων χωρίς να έχει προγραμματιστεί εμφανώς για το πού να εξετάσει ή τι να συμπεράνει.

Neural network: είναι ένας τύπος μηχανικής μάθησης που αποτελείται από αλληλοσυνδεόμενες μονάδες (όπως οι νευρώνες) που επεξεργάζονται τις πληροφορίες ανταποκρινόμενο σε εξωτερικές εισαγωγές δεδομένων, προωθώντας πληροφορίες μεταξύ κάθε μονάδας. Η διαδικασία απαιτεί πολλαπλές διελεύσεις στα δεδομένα προκειμένου να βρεθούν συνδέσεις και να γίνει εξαγωγή νοήματος από ακαθόριστα δεδομένα.

Deep learning: Χρησιμοποιεί τεράστια neural networks με πολλά επίπεδα μονάδων επεξεργασίας, αξιοποιώντας τις εξελίξεις στην υπολογιστική ισχύ και τις βελτιωμένες τεχνικές εκπαίδευσης για την μάθηση πολύπλοκων μορφών σε μεγάλες ποσότητες δεδομένων. Οι κοινές εφαρμογές της περιλαμβάνουν την αναγνώριση εικόνας και ομιλίας.

Cognitive computing: είναι ένα υπο-επίπεδο του ΑΙ που στοχεύει σε μια φυσική, ανθρωπόμορφη αλληλεπίδραση με μηχανές. Κατά τη χρήση τεχνικών ΑΙ και cognitive computing, ο απώτατος **στόχος είναι η προσομοίωση ανθρώπινων αλληλεπιδράσεων από μια μηχανή** μέσω της ικανότητας να ερμηνεύσουν εικόνες και ομιλία – και να υπάρξει κανονική απάντηση από την μηχανή η οποία έχει ειρμό.

Computer vision : βασίζεται στην αναγνώριση μορφών (pattern recognition) και στο Deep learning ώστε να αναγνωρίζεται τι υπάρχει σε μια εικόνα ή ένα βίντεο. Όταν οι μηχανές μπορούν να επεξεργαστούν, να αναλύσουν και να κατανοήσουν εικόνες μπορούν να συλλάβουν εικόνες ή βίντεο σε πραγματικό χρόνο και να ερμηνεύσουν τα περιβάλλοντά τους.

Natural Language Processing (NLP): είναι η ικανότητα των υπολογιστών να αναλύουν, να κατανοούν και να παράγουν ομιλούμενη γλώσσα, συμπεριλαμβανομένης της ομιλίας. Το επόμενο στάδιο στην ΕΦΓ είναι η φυσική γλωσσική αλληλεπίδραση, η οποία επιτρέπει στους ανθρώπους να επικοινωνούν με υπολογιστές χρησιμοποιώντας την κανονική, καθημερινή γλώσσα για την εκτέλεση καθηκόντων.

Συνοπτικά, ο στόχος της τεχνητής νοημοσύνης είναι να παρέχει ένα λογισμικό που μπορεί να αναλύει τα εισαγόμενα δεδομένα και να εξηγεί τα εξερχόμενα. **Η τεχνητή νοημοσύνη παρέχει ανθρωπόμορφες αλληλεπιδράσεις με λογισμικό και θα προσφέρει υποστήριξη στη λήψη αποφάσεων για συγκεκριμένα καθήκοντα**, αλλά δεν αποτελεί υποκατάστατο των ανθρώπων – και δεν αναμένεται να γίνει. Εμείς σε αυτήν την εργασία θα αναλύσουμε περαιτέρω την **Επεξεργασία Φυσικής Γλώσσας** που σχετίζεται με το θέμα της εργασίας στην συνέχεια.

2.5 Διαχωρίζοντας την Τεχνητή Νοημοσύνης από τον ενθουσιασμό

Ζούμε σε συναρπαστικές περιόδους. Οι σχέσεις μας με τις μηχανές πράγματα αλλάζουν γρήγορα. Τα ποντίκια και τα πληκτρολόγια μας κάνουν ακριβώς ότι τους λέμε και συσκευές όπως η Amazon Echo μπορούν να μας βοηθήσουν να κάνουμε απλές εργασίες όπως το άνοιγμα των φώτων ή πιο σύνθετα καθήκοντα, όπως η απάντηση σε ερωτήσεις που τους κάνουμε.

Εάν ανησυχείτε για τα μηχανήματα που θα καταλάβουν τον κόσμο, μπορείτε να κοιμάστε ήσυχα. Δεν θα συμβεί με βάση την τεχνολογία που χρησιμοποιείται σήμερα. **Η τάση είναι να επισημάνουμε ότι όλα είναι ΑΙ οτιδήποτε κάνει κάτι εξωπραγματικά έξυπνο ή απροσδόκητο, αλλά στην πραγματικότητα δεν είναι ΑΙ. Το κομπιουτεράκι χειρός μου είναι καλύτερος στην αριθμητική από ότι θα είμαι ποτέ - δεν είναι ΑΙ. Ένα δέντρο απόφασης δεν είναι ΑΙ. Ένα έξτρα SQL Clauses σε ένα ερώτημα SQL δεν είναι ΑΙ.**

Έχουμε δει απίστευτες προόδους στην δημιουργία των αλγορίθμων με εκπληκτική ακρίβειας στις εργασίες που θα μπορούσε να κάνει ένας άνθρωπος. Μέχρι πρόσφατα πιστεύαμε ότι το παιχνίδι Go δεν μπορούσε να αυτοματοποιηθεί (μηχανογραφηθεί), και τώρα ένα μηχάνημα μας νικά σε αυτό και μας ξεπέρασε. Ή στον τομέα της υγειονομικής

περίθαλψης, οι αλγόριθμοι μπορούν να ανιχνεύσουν μορφές καρκίνου από ιατρικές εικόνες το ίδιο καλά με τους ακτινολόγους – κάτι που αλλάζει την ζωή μας.

Αυτοί οι αλγόριθμοι έχουν υπεράνθρωπες ικανότητες επειδή κάνουν την εργασία τους αξιόπιστα, με ακρίβεια, επανειλημμένα και όλο το εικοσιτετράωρο. **Ωστόσο, απέχουμε πολύ από τη δημιουργία μηχανών που μπορούν να σκέπτονται ή να συμπεριφέρονται σαν άνθρωποι.** Τα τρέχοντα συστήματα ΑΙ εκπαιδεύονται για να επιτελούν ένα ανθρώπινο καθήκον με έξυπνο, μηχανογραφημένο τρόπο, αλλά εκπαιδεύονται να κάνουν ένα έργο - και ένα μόνο έργο. **Το σύστημα που μπορεί να παίζει Go δεν μπορεί να παίζει πασιέντζα ή πόκερ και δεν θα αποκτήσει δεξιότητες για να το κάνει.** Το λογισμικό που οδηγεί ένα αυτόνομο όχημα δεν μπορεί να χειριστεί τα φώτα στο σπίτι σας.

Αυτό δεν σημαίνει ότι αυτή η μορφή ΑΙ δεν είναι ισχυρή. Έχει τη δυνατότητα να μετασχηματίζει πολλές βιομηχανίες - ίσως κάθε βιομηχανία στο μέλλον. Τα συστήματα που μαθαίνουν υπό εποπτεία, από την κορυφή προς τα κάτω με βάση τα δεδομένα εκπαίδευσης δεν μπορούν να αναπτυχθούν πέρα από το περιεχόμενο των δεδομένων. Δηλαδή **δεν μπορούν να δημιουργήσουν ή να καινοτομήσουν ή να αιτιολογήσουν.**

Η ψευδαίσθηση της νοημοσύνης είναι το μόνο που μπορούμε να χειριστούμε και είναι αυτό που πρέπει να χειριστούμε για τώρα. **Θέλουμε να ξεγελάσουμε από το μηχάνημα, με έξυπνο τρόπο. Το υπόλοιπο είναι διαφημιστικές εκστρατείες και ενθουσιασμός.**

2.6 Πότε θα έχουμε πραγματικά έξυπνη Τεχνική Νοημοσύνη

Η νοημοσύνη απαιτεί κάποια μορφή δημιουργικότητας, καινοτομίας, διαίσθησης, ανεξάρτητης επίλυσης προβλημάτων και ευαισθησίας. Τα συστήματα που οικοδομούμε με βάση το Deep Learning δεν μπορούν να έχουν αυτά τα χαρακτηριστικά. Δεν θέλω να βάλω ένα χρονικό πλαίσιο για το πότε θα είναι έξυπνο το ΑΙ. Φανταστείτε ότι πριν από αρκετές δεκαετίες πιστεύαμε ότι οι μηχανές θα ενεργούν και θα σκέφτονται σαν τους ανθρώπους πράγμα που μέχρι τώρα, δεν έχει γίνει. Η τεχνολογία που διαθέτουμε σήμερα δεν μπορεί να λύσει αυτό το πρόβλημα. **Πρέπει να υπάρξει μια απρόσμενη τεχνολογική στροφή για να φτάσουμε στην αληθινή ΑΙ.** Δεν νομίζω ότι ακόμα έχει έρθει αυτή η απρόσμενη τεχνολογική ανατροπή, όμως το ψάχνουμε.

3

Επεξεργασία φυσικής γλώσσας – Natural Language Processing (NLP)

Σε αυτήν την ενότητα θα προσπαθήσουμε να ασχοληθούμε και να απαντήσουμε κάποια βασικά ερωτήματα που έχουν να κάνουν με την επεξεργασία της φυσικής γλώσσας και πιο συγκεκριμένα με τα εξής:

- Εισαγωγή στην Επεξεργασία της Φυσικής Γλώσσας
- Γιατί είναι σημαντική η Επεξεργασία της Φυσικής Γλώσσας
- Εργασίες της Επεξεργασία της Φυσικής Γλώσσας
- Εφαρμογές NLP στην καθημερινότητα μας

Στις παρακάτω υποενότητες θα απαντηθούν, τα παραπάνω ερωτήματα.

3.1 Εισαγωγή στην Επεξεργασία της Φυσικής Γλώσσας

Η Επεξεργασία της Φυσικής Γλώσσας είναι ένα πεδίο της επιστήμης των υπολογιστών και της γλωσσολογίας που ασχολείται με τις **αλληλεπιδράσεις μεταξύ των υπολογιστών και της φυσικής γλώσσας**. Η ανάπτυξή της ξεκίνησε σαν ένα μέρος της **τεχνητής νοημοσύνης**. Ουσιαστικά, αναφέρεται στη δυνατότητα χειρισμού της φυσικής γλώσσας από τους υπολογιστές, σε επίπεδο κατανόησης και επικοινωνίας με τους ανθρώπους. Πρωταρχικός στόχος αποτελούσε η κατανόηση των εκφράσεων της φυσικής γλώσσας και αργότερα η απάντηση αυτών με εύστοχες και χρήσιμες απαντήσεις, ανάλογα με την περίπτωση κάθε φορά.

Οι τεχνολογίες που βασίζονται στον τομέα της Επεξεργασίας της Φυσικής Γλώσσας αυξάνονται ραγδαία καθώς παρατηρείται όλο και περισσότερο η ανάγκη για τη χρησιμοποίησή τους. Με την παροχή όλο και περισσότερων διεπαφών μεταξύ ανθρώπου-μηχανής και την εξελιγμένη πρόσβαση σε αποθηκευμένες πληροφορίες, η επεξεργασία της φυσικής γλώσσας παίζει πλέον **ένα σημαντικό ρόλο στην πολύγλωσση κοινωνία της πληροφορίας**.

Οι προκλήσεις της **Επεξεργασία της Φυσικής Γλώσσας** συχνά εμπλέκονται με κατανόηση της φυσικής γλώσσας, παραγωγή/γέννηση φυσικής γλώσσας, συστήματα διαλόγου ή κάποιο συνδυασμό των παραπάνω. Η **Επεξεργασία της Φυσικής Γλώσσας** έχει να κάνει με τη δημιουργία πραγματικών εφαρμογών χρησιμοποιώντας τεχνικές NLP. Σε ένα πρακτικό πλαίσιο είναι ανάλογο με τη διδασκαλία μιας γλώσσας σε ένα παιδί.

3.2 Γιατί είναι σημαντική η Επεξεργασία της Φυσικής Γλώσσας

Η Επεξεργασία της Φυσικής Γλώσσας (NLP) είναι μια δεξιότητα που απαιτείται όλο και περισσότερο στον κλάδο. Μετά την έλευση των μεγάλων δεδομένων (big data), η μεγάλη πρόκληση είναι ότι χρειαζόμαστε περισσότερους ανθρώπους που είναι **καλοί όχι μόνο με τα δομημένα δεδομένα** αλλά και με τα **ημι-δομημένα ή τα μη-δομημένα** δεδομένα.

Δημιουργούμε petabytes ιστολογίων, tweets, feeds στο Facebook, συζητήσεις, emails και κριτικές. **Οι εταιρείες συλλέγουν όλα αυτά τα διαφορετικά είδη δεδομένων για καλύτερη στόχευση των πελατών και ουσιαστικές γνώσεις.** Για να επεξεργαστούμε όλες αυτές τις αδόμητες πηγές δεδομένων χρειαζόμαστε ανθρώπους που κατέχουν σε βάθος την Επεξεργασία της Φυσικής Γλώσσας (NLP).

Είμαστε στην εποχή των πληροφοριών! Δεν μπορούμε ούτε να φανταστούμε τη ζωή μας χωρίς την Google. Χρησιμοποιούμε **Speech engines** (όπως την Siri, Cortana, Google Voice, Alexa) για τα περισσότερα βασικά πράγματα. Χρησιμοποιούμε φίλτρα ανεπιθύμητης αλληλογραφίας (**Spam classifiers**) για το φιλτράρισμα ανεπιθύμητων ηλεκτρονικών μηνυμάτων. Χρειαζόμαστε τον **ορθογραφικό έλεγχο** για τα έγγραφα του Word. Υπάρχουν πολλά παραδείγματα πραγματικών εφαρμογών NLP γύρω μας.

Το 90% των παγκόσμιων δεδομένων, δημιουργήθηκαν τα τελευταία δύο χρόνια. Ο κόσμος είναι γεμάτος από αδόμητα, πλούσια σε κείμενο δεδομένα. Από μηνύματα ηλεκτρονικού ταχυδρομείου μέχρι tweets πελατών. Οι πληροφορίες που έχουν ταφεί σε όλο αυτό το κείμενο έχουν τη δυνατότητα να παρέχουν **πολύτιμες επιχειρηματικές ιδέες.**

3.3 Εργασίες της Επεξεργασία της Φυσικής Γλώσσας

Μερικές από τις πιο συνηθισμένες εργασίες, όπως η κατανόηση λέξεων, φράσεων και ο σχηματισμός γραμματικών προτάσεων που είναι γραμματικά σωστές, είναι πολύ φυσικές

για τον άνθρωπο για τον υπολογιστή δεν είναι. Στην Επεξεργασία της Φυσικής Γλώσσας, μερικές από αυτές τις εργασίες είναι οι εξής:

- **μεταφράζονται σε χώρισμα του κειμένου σε λεκτικές μονάδες (tokenization)**
- **αναγνώριση μερών του λόγου (part of speech tagging)**
- **σε συντακτική ανάλυση (parsing)**
- σε μηχανική μετάφραση (machine translation)
- αναγνώριση ομιλίας (speech recognition)
- **σε αναγνώριση ονομαστικών οντοτήτων (Named Entity Recognition-NER)**
- σε παραγωγή φυσικής γλώσσας (natural language generation)
- σε κατανόηση της φυσικής γλώσσας (natural language understanding)
- **απάντηση ερωτήσεων (question answering),**
- εξαγωγή σχέσεων (relationship extraction),
- ανάλυση συνθήματος (sentiment analysis)
- διαίρεση του γραπτού κείμενου σε μονάδες με ουσιαστικό νόημα (topic segmentation)
- η αποσαφήνιση της έννοιας μια λέξης (word sense disambiguation)
- δημιουργία περίληψης αυτόματα (automatic summarization)
- εύρεση όλων των εκφράσεων που αναφέρονται σε μια ίδια οντότητα (coreference resolution)
- αναγνώριση φωνής (speech recognition),
- κείμενο σε ομιλία (text-to-speech)

Τα περισσότερα από αυτά είναι ακόμα δύσκολες προκλήσεις για τους υπολογιστές αλλά σε αυτή την εργασία θα **επικεντρωθούμε στην απάντηση ερωτήσεων (question answering)** και σε όποιες άλλες εργασίες απαιτούνται για την επίτευξη αυτού.

3.4 Εφαρμογές NLP στην καθημερινότητα μας

Μερικά παραδείγματα εφαρμογών Επεξεργασία Φυσικής Γλώσσας τα οποία χρησιμοποιούμε καθημερινά αλλά δεν ξέρουμε ότι είναι εφαρμογές της Επεξεργασία της Φυσικής Γλώσσας :

- διόρθωση ορθογραφίας, υπάρχει στο Microsoft Word ή οποιαδήποτε άλλο κειμενογράφο
- μηχανές αναζήτησης όπως Google, Bing, Yahoo και **wolframalpha**.
- μηχανές φωνής όπως Siri , Cortana, Google Voice & Now, Alexa - Amazon Echo
- φίλτρα ανεπιθύμητη αλληλογραφίας που υπάρχουν σε όλες τις υπηρεσίες e-mail
- οι ροές ειδήσεων όπως το Google News και το Yahoo! News
- μεταφραστές κειμένων όπως το Google Translate
- το σύστημα απάντησης ερωτήσεων **IBM Watson** που μπορεί να απαντήσει σε ερωτήσεις που τίθενται στη φυσική γλώσσα και αναπτύχθηκε από την IBM στα πλαίσια του DeepQA
- η πλατφόρμα **Google Cloud Natural Language API** (<https://cloud.google.com/natural-language/>)
- η πλατφόρμα **Amazon Lex** (<https://aws.amazon.com/lex/>)

4 Συστήματα Απάντησης Ερωτήσεων – Question Answering (QA)

Σε αυτήν την ενότητα θα ασχοληθούμε και θα απαντήσουμε σε κάποια βασικά ερωτήματα που έχουν να κάνουν με τα συστήματα απάντησης ερωτήσεων και πιο συγκεκριμένα με τα εξής:

- Τι είναι ένα σύστημα Απάντησης Ερωτήσεων
- Τι είδη ερωτήσεων υπάρχουν
- Μέθοδοι Απάντησης Ερωτήσεων
- Μερικά συστήματα-μέθοδοι απάντησης ερωτήσεων

Στις παρακάτω υποενότητες θα απαντηθούν, τα παραπάνω ερωτήματα.

4.1 Τι είναι ένα σύστημα Απάντησης Ερωτήσεων

Η απάντηση ερωτήσεων (QA) είναι ένας κλάδος του τομέα της πληροφορικής στους τομείς της ανάκτησης πληροφοριών και της **επεξεργασίας φυσικής γλώσσας (NLP)**, η οποία ασχολείται με **την κατασκευή συστημάτων που απαντούν αυτόματα σε ερωτήσεις που θέτουν οι άνθρωποι σε μια φυσική γλώσσα.**

4.2 Τι είδη ερωτήσεων υπάρχουν

Εάν δούμε τα είδη των ερωτήσεων σε υψηλό επίπεδο υπάρχουν δυο μεγάλες κατηγορίες:

- Η **ερωτήσεις κλειστού πεδίου** (Anon., 2018) ασχολείται με ερωτήσεις κάτω από έναν συγκεκριμένο τομέα (για παράδειγμα ιατρική) και μπορεί να εκμεταλλευτεί ειδικές γνώσεις για το **συγκεκριμένο τομέα**, οι οποίες συχνά τυποποιούνται στις οντολογίες. Εναλλακτικά, ο τομέας κλειστού χώρου μπορεί να αναφέρεται σε μια

κατάσταση όπου μόνο περιορισμένο είδος ερωτήσεων είναι αποδεκτές, όπως ερωτήσεις που ζητούν περιγραφικές και όχι διαδικαστικές πληροφορίες.

- Η ερώτησης ανοιχτού τομέα (Anon., 2018) ασχολείται με ερωτήσεις για σχεδόν οτιδήποτε και μπορεί να βασιστεί μόνο σε γενικές οντολογίες και την γνώση του κόσμου. Από την άλλη πλευρά, αυτά τα συστήματα έχουν συνήθως πολύ περισσότερα διαθέσιμα δεδομένα από τα οποία μπορούν να εξάγουν την απάντηση.

Στα πλαίσια της εργασίας να ασχοληθούμε με τις ερωτήσεις ανοιχτού τομέα.

4.3 Μέθοδοι Απάντησης Ερωτήσεων

Ένα σύστημα απάντησης ερωτήσεων εξαρτάται πολύ από ένα καλό corpus (δηλαδή μια συλλογή από έγγραφα που περιέχουν την απάντηση) είναι το ελάχιστο που χρειαζόμαστε για ένα τέτοια σύστημα. Επομένως, έχει νόημα να έχουμε ένα μεγάλο μέγεθος συλλογής γενικά προσδίδει στην καλή επίδοση του συστήματος, εκτός εάν ο τομέας του ερωτήματος είναι ανεξάρτητος της συλλογής.

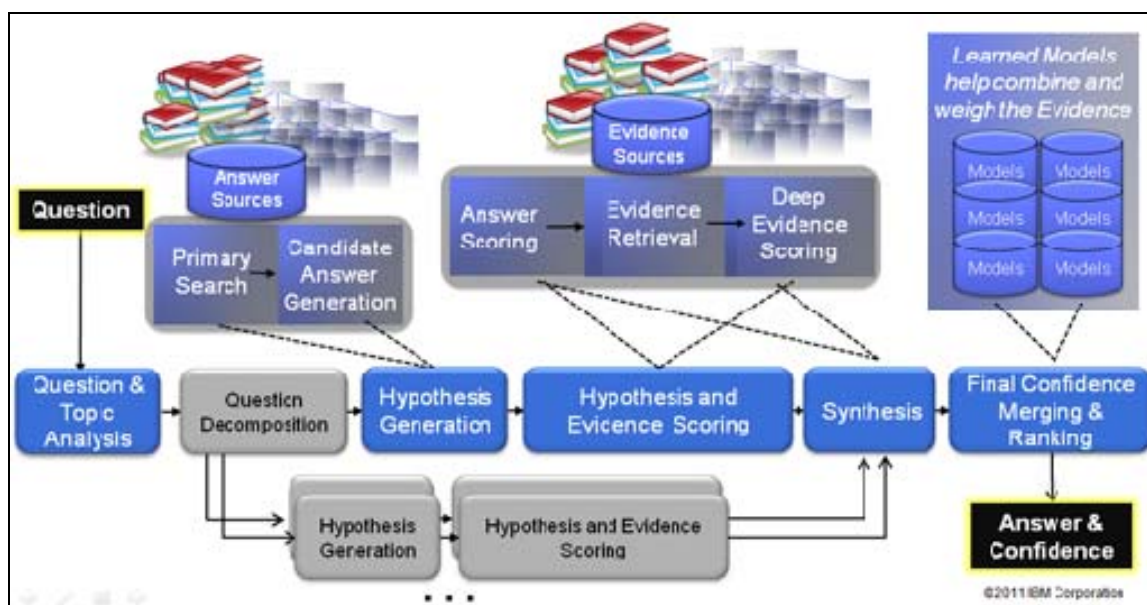
Έπειτα ακολουθεί η εξαγωγή λέξεων-κλειδιών για τον προσδιορισμό του τύπου της ερώτησης, σε ορισμένες περιπτώσεις, υπάρχουν σαφείς λέξεις που υποδεικνύουν τον τύπο ερωτήματος άμεσα πχ. "Ποιός (Who)", "Πού (Where)" ή "Πόσες (How many) ", αυτές οι λέξεις λένε στο σύστημα ότι οι απαντήσεις πρέπει να είναι τύπου "Πρόσωπο", "Θέση" ή "Αριθμός" αντίστοιχα. Για παράδειγμα, η λέξη "Όταν (When)" υποδεικνύει ότι η απάντηση πρέπει να είναι τύπου "Ημερομηνίας (Date)", οπότε έπειτα από NER Parsing (θα εξηγηθεί στο κεφάλαιο που ακολουθεί) επιλέγω μια οντότητα τύπου "DATE".

Δυστυχώς, μερικές ερωτηματικές λέξεις όπως οι "Ποιά (Which)", "Τι (What)" ή "Πώς (How)" δεν δίνουν σαφείς τύπους απαντήσεων. Κάθε μία από αυτές τις λέξεις μπορεί να αντιπροσωπεύει περισσότερους από έναν τύπους. Σε καταστάσεις όπως αυτό, πρέπει να εξεταστούν και άλλες λέξεις στο ερώτημα. Το πρώτο πράγμα που πρέπει να κάνετε είναι να βρείτε τις λέξεις που μπορούν να υποδηλώνουν το νόημα της ερώτησης.

Τέλος, βρίσκω το σύνολο από προτάσεις που περιέχουν τις λέξεις που υποδηλώνουν το νόημα της ερώτησης και βρίσκω αυτή που περιέχει την απάντηση και έπειτα την φράση της πρότασης που είναι η απάντηση.

4.4 Μερικά συστήματα-μέθοδοι απάντησης ερωτήσεων

Το **DeepQA** (Kalyanpur, et al., 2012), είναι μια αρχιτεκτονική λογισμικού για βαθιά ανάλυση του περιεχομένου και μια συλλογιστική βασισμένη στην απόδειξη - αξιολόγηση, η οποία ενσαρκώνει αυτήν την φιλοσοφία. Η αρχιτεκτονική DeepQA βλέπει το πρόβλημα της αυτόματης απάντησης ερωτήσεων ως **εργασία παραγωγής και αξιολόγησης μαζικά παράλληλων υποθέσεων**. Ως αποτέλεσμα, το DeepQA δεν είναι απλώς ερώτημα / απάντηση - μάλλον μπορεί να θεωρηθεί ως ένα σύστημα που εκτελεί διαφορετική διάγνωση: **παράγει ένα ευρύ φάσμα πιθανοτήτων και για το καθένα αναπτύσσει ένα επίπεδο εμπιστοσύνης** συγκεντρώνοντας, αναλύοντας και αξιολογώντας στοιχεία βάσει διαθέσιμων δεδομένων (Βλέπε την Εικόνα 1).



Εικόνα 1: Η αρχιτεκτονική του συστήματος DeepQA

Το **SpaRQL** (από το SPARQL πρωτόκολλο και το RDF Query Language) είναι μια γλώσσα ερωτημάτων για RDF (είναι ένα πλαίσιο για την περιγραφή πόρων στον ιστό). Ουσιαστικά πρόκειται για μια γλώσσα η οποία κάνει ερωτήσεις σε μια βάση με δεδομένα αποθηκευμένα σε μορφή RDF.

Το **START** (Barbieri, et al., 2009) (<http://start.csail.mit.edu/index.php>) το παγκόσμια πρώτο διδακτικό σύστημα απάντησης ερωτήσεων, έχει λειτουργία on-line συνεχόμενη από

τον Δεκέμβριο του 1993. Έχει αναπτυχθεί από τον Boris Katz και το Group InfoLab (εργαστηρίου Computer Science και Τεχνητής Νοημοσύνης του MIT). Σε αντίθεση με τα συστήματα ανάκτησης πληροφοριών (πχ. μηχανές αναζήτησης), **το START επιδιώκει να παρέχει στους χρήστες "μόνο τις σωστές πληροφορίες", αντί να παρέχει απλά μια λίστα με επιτυχίες.** Επί του παρόντος, το σύστημα μπορεί να απαντήσει σε εκατομμύρια αγγλικών ερωτήσεων σχετικά με μέρη (πχ. πόλεις, χώρες, λίμνες, συντεταγμένες, καιρικές συνθήκες, χάρτες, δημογραφικά στοιχεία, πολιτικά και οικονομικά συστήματα), ταινίες (πχ. τίτλους, ηθοποιούς, σκηνοθέτες), άτομα (πχ. ημερομηνίες γέννησης, βιογραφίες), ορισμοί λεξικών και πολλά άλλα.

Το **AskMSR** (Brill, et al., 2002), είναι μια αρχιτεκτονική απάντησης ερωτήσεων όπου αξιολογούμε συστηματικά την συμβολή των διαφόρων στοιχείων του συστήματος με ακρίβεια. Το σύστημα διαφέρει από τα περισσότερα συστήματα απαντήσεων ερωτήσεων στην εξάρτησή του από τον πλεονασμό δεδομένων παρά από εξελιγμένες γλωσσικές αναλύσεις είτε των ερωτήσεων είτε των υποψηφίων απαντήσεων. Επειδή **μια λανθασμένη απάντηση είναι συχνά χειρότερη από την μη απάντηση**, διερευνάμε επίσης στρατηγικές για την **πρόβλεψη του πότε είναι πιθανό το σύστημα απάντησης ερωτήματος να δώσει λανθασμένη απάντηση.**

5 Εργαλεία/Γλώσσα προγραμματισμού που χρησιμοποιήθηκαν

Για την εργασία χρησιμοποιήθηκαν διάφορα εργαλεία. Πιο αναλυτικά χρησιμοποιήθηκαν τα εξής:

- Γλώσσας προγραμματισμού Python
- Περιβάλλον ανάπτυξης PyCharm
- Πλατφόρμα NLTK
- Βιβλιοθήκη Stanford CoreNLP
- Βιβλιοθήκη TextBlob
- Γραφικό περιβάλλον χρήστη tKinter

Στις παρακάτω υποενότητες θα αναλυθούν διεξοδικά, αναφορικά με τις δυνατότητες τους και την χρήση τους.

5.1 Γλώσσας προγραμματισμού Python

Η Python (<https://www.python.org/>) είναι μια interpreted και αντικειμενοστρεφής γλώσσα προγραμματισμού υψηλού επιπέδου με δυναμική σημασιολογία. Οι υψηλού επιπέδου ενσωματωμένες δομές δεδομένων, σε συνδυασμό με τη δυναμική πληκτρολόγηση και τη δυναμική σύνδεση, το καθιστούν πολύ ελκυστικό για την **ταχεία ανάπτυξη εφαρμογών**, καθώς και για τη χρήση ως γλώσσα δέσμης ενεργειών ή κόλλας για τη σύνδεση των υπαρχόντων στοιχείων μαζί. Η απλή και εύκολη σύνταξη της Python δίνει έμφαση στην αναγνωσιμότητα και επομένως μειώνει το κόστος συντήρησης του προγράμματος. Η Python υποστηρίζει modules και πακέτα, τα οποία ενθαρρύνουν τη διαμόρφωση του προγράμματος και την επαναχρησιμοποίηση του κώδικα. Ο interpreter της Python και η εκτεταμένη τυποποιημένη βιβλιοθήκη της είναι διαθέσιμα είτε σαν πηγή είτε σε δυαδική μορφή για χρήση σε όλες τις μεγάλες πλατφόρμες και μπορούν να διανεμηθούν ελεύθερα.

Συχνά, οι προγραμματιστές ερωτεύονται την Python λόγω της **αυξημένης παραγωγικότητας που προσφέρει**. Δεδομένου ότι δεν υπάρχει κανένα βήμα για τη μεταγλώττιση, ο κύκλος επεξεργασίας-δοκιμής-εντοπισμού σφαλμάτων είναι απίστευτα

γρήγορος. Τα προγράμματα **εντοπισμού σφαλμάτων Python** είναι εύκολα: ένα σφάλμα ή κακή είσοδος δεν θα προκαλέσει ποτέ σφάλμα κατάτμησης. Αντίθετα, όταν ο διερμηνέας ανακαλύψει ένα σφάλμα, εγείρει μια εξαίρεση. Όταν το πρόγραμμα δεν καλύψει την εξαίρεση, ο διερμηνέας εκτυπώνει ένα ίχνος στοίβας. Ένα εργαλείο εντοπισμού σφαλμάτων σε επίπεδο πηγής επιτρέπει την επιθεώρηση των τοπικών και παγκόσμιων μεταβλητών, την αξιολόγηση των αυθαίρετων εκφράσεων, τη ρύθμιση των σημείων διακοπής, τη μετάβαση από τον κώδικα σε μια γραμμή τη φορά και ούτω καθεξής. Το πρόγραμμα εντοπισμού σφαλμάτων γράφεται στην ίδια την Python, μαρτυρώντας την **ενδοσκοπική δύναμη του Python**. Από την άλλη πλευρά, συχνά ο ταχύτερος τρόπος για τον εντοπισμό σφαλμάτων σε ένα πρόγραμμα είναι να προσθέσετε μερικές εκτυπώσεις στον κώδικα: ο γρήγορος κύκλος επεξεργασίας-δοκιμής-εντοπισμού καθιστά αυτή την απλή προσέγγιση πολύ αποτελεσματική.

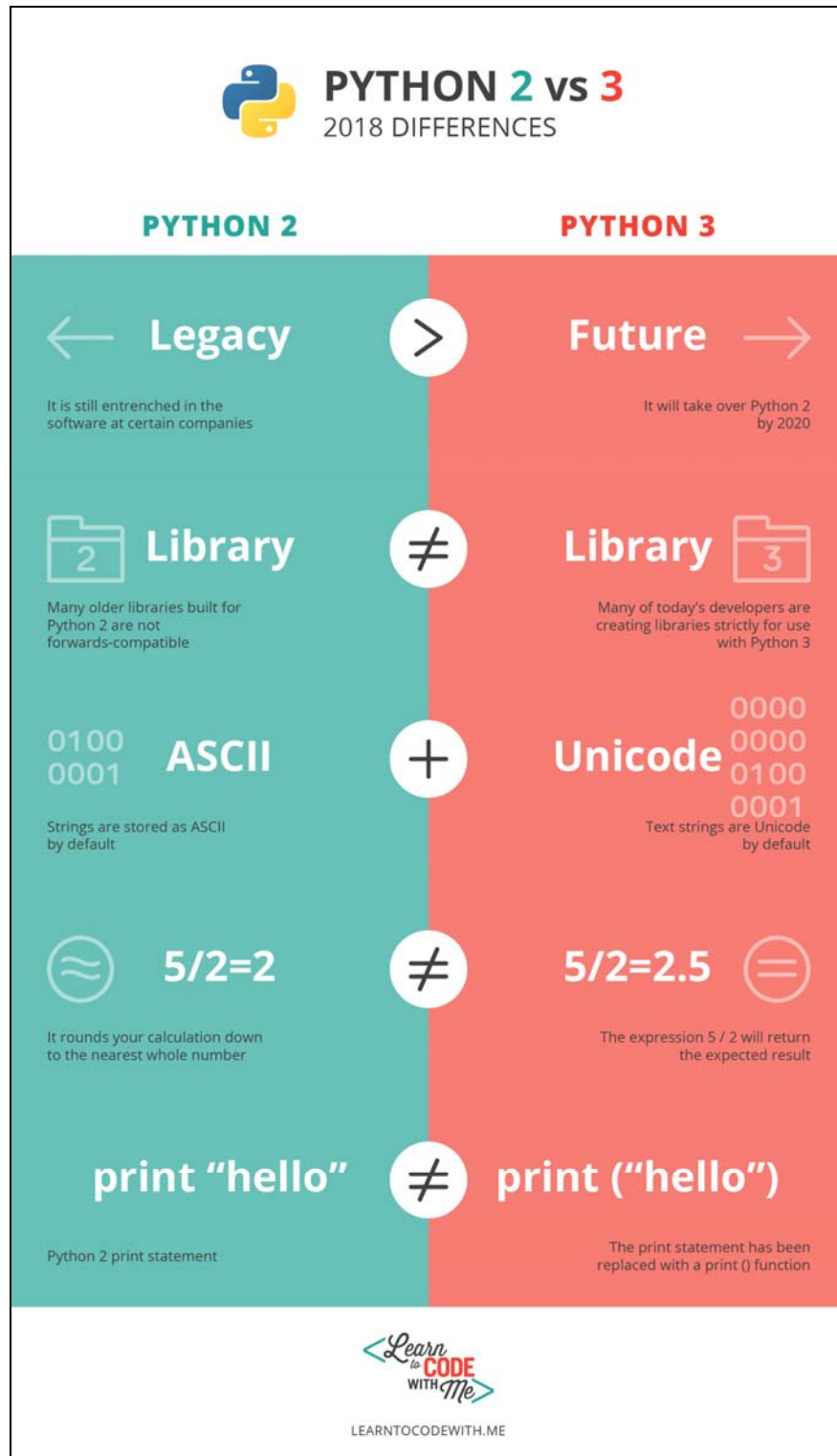
5.1.1 Γιατί επιλέγω την Εκδόση Python 3.5

Στην Python υπάρχει το ρητό: **Η Python 2.x είναι κληρονομιά, η Python 3.x είναι το παρόν και το μέλλον της γλώσσας.**

Η έκδοση Python 3.0 κυκλοφόρησε το 2008. Η τελική έκδοσης της 2.x ήταν η έκδοση 2.7 βγήκε στα μέσα του 2010, με δήλωση ότι αυτή η έκδοσης σημαίνει και το τέλος της ζωής και πως η έκδοση 3.x βρίσκεται σε ενεργό ανάπτυξη και έχει ήδη δει πάνω από πέντε χρόνια σταθερών εκδόσεων, συμπεριλαμβανομένης της έκδοσης 3.3 το 2012, 3.4 το 2014, **3.5 το 2015** και 3.6 το 2016. Αυτό σημαίνει ότι όλες οι πρόσφατες τυπικές βελτιώσεις βιβλιοθήκης, για παράδειγμα, θα είναι διαθέσιμες μόνο από την Python 3.x.

Ο Guido van Rossum (ο αρχικός δημιουργός της γλώσσας Python) αποφάσισε να σταματήσει η υποστήριξη στην Python 2.x και να υποστηρίζεται πλέον μόνο η Python 3.x. Η πιο δραστική βελτίωση είναι η **καλύτερη υποστήριξη Unicode** (με όλες τις συμβολοσειρές κειμένου να είναι το Unicode από προεπιλογή) καθώς και η απομάκρυνση των bytes/Unicode.

Εκτός αυτού, πολλές πτυχές της βασικής γλώσσας έχουν προσαρμοστεί ώστε να είναι **πιο εύκολο για τους νεοεισερχόμενους να μάθουν** και να είναι πιο συνεπείς με την υπόλοιπη γλώσσα.



Εικόνα 2: Οι διαφορές ανάμεσα στην Python 2 και Python 3.

Ο κυριότερος λόγος όμως για την επιλογή της έκδοσης Python 3.5 είναι το NLTK μια βιβλιοθήκη την οποία θα αναλύσουμε στην συνέχεια η οποία παρότι υποστηρίζει τις εκδόσεις Python 2.7, 3.4, 3.5, 3.6 ή 3.7 από προσωπική εμπειρία και έπειτα από πολλές δοκιμές για την ώρα **μόνο στην έκδοση Python 3.5 δεν παρουσιάζει κανένα πρόβλημα** και είναι μια από τις πιο δημοφιλείς βιβλιοθήκες της κοινότητας του NLP (Επεξεργασίας της Φυσικής Γλώσσας).

5.2 Περιβάλλον ανάπτυξης PyCharm

Το PyCharm (<https://www.jetbrains.com/pycharm/>) είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που χρησιμοποιείται στον προγραμματισμό υπολογιστών, ειδικά για τη γλώσσα Python. Αναπτύσσεται από την τσεχική εταιρεία **JetBrains**. Παρέχει ανάλυση κώδικα, γραφικό εργαλείο εντοπισμού σφαλμάτων, ολοκληρωμένο έλεγχο συσκευών, ολοκλήρωση με συστήματα ελέγχου εκδόσεων (VCSes) και υποστηρίζει την ανάπτυξη ιστού με το Django. Το PyCharm είναι πολλαπλής πλατφόρμας, με εκδόσεις Windows, macOS και Linux. Η Κοινοτική (Community) Έκδοση κυκλοφορεί με την Άδεια Apache και είναι δωρεάν, επίσης υπάρχει η Επαγγελματική Έκδοση με επιπλέον χαρακτηριστικά, η οποία κυκλοφορεί με άδεια ιδιοκτησίας.

Οι λόγοι που το επέλεξα ήταν ο **έξυπνος κειμενογράφος κώδικα** ο επιτρέπει στους προγραμματιστές να διαβάζουν εύκολα τον κώδικα μέσω των χρωμάτων, να επιλέγουν το κατάλληλο στυλ κωδικοποίησης και να επωφελούνται από προτάσεις για την ολοκλήρωση κώδικα με γνώμονα το περιβάλλον. Οι **επιλογές πλοήγησης έξυπνου κώδικα** που παρέχονται από τον PyCharm βοηθούν τους προγραμματιστές να επεξεργάζονται και να βελτιώνουν τον κώδικα χωρίς να βάζουν επιπλέον χρόνο και προσπάθεια. Η **υποστήριξη για τις επιστημονικές βιβλιοθήκες της Python**, που βοηθά τους προγραμματιστές να χρησιμοποιούν πιο αποτελεσματικά την Python σε μεγάλα έργα δεδομένων και της επιστήμης δεδομένων. Το **οπτικό πρόγραμμα εντοπισμού σφαλμάτων** και το **ενσωματωμένο τοπικό τερματικό για Windows** που επιτρέπει στους προγραμματιστές να συνεχίσουν να κωδικοποιούν και να δοκιμάζουν χωρίς να αφήνουν το IDE.

Τέλος, δεν χρειάζεστε το PyCharm για την ανάπτυξη Python. Αλλά αν θέλετε να **αυξηθεί η παραγωγικότητά σας** τότε το PyCharm είναι μια πολύ καλή επιλογή. Για την υλοποίηση της εργασίας χρησιμοποίησα την **Έκδοση JetBrains PyCharm Edu 4.0.2**.

5.3 Βιβλιοθήκη NLTK (Natural Language ToolKit)

Το NLTK (Natural Language ToolKit) (Loper & Bird, 2002) είναι μια ανοικτού κώδικα βιβλιοθήκη συναρτήσεων, που έχει αναπτυχθεί σε γλώσσα Python, με στόχο την επεξεργασία της φυσικής γλώσσας και ανάπτυξη ανάλογων εφαρμογών. Η ανάπτυξή του ξεκίνησε το **2001**, σαν ένα μέρος του μαθήματος της **Υπολογιστικής Γλωσσολογίας** του τμήματος Υπολογιστών και Επιστήμης της Πληροφορίας στο πανεπιστήμιο της Πενσυλβανία, κυρίως για την Αγγλική γλώσσα.

Περιλαμβάνει components, δομές δεδομένων και interfaces και συνοδεύεται από οδηγούς, αναφορές και τεχνικές οδηγίες που εξηγούν την ακριβή λειτουργία του αλλά και **καθοδηγούν για τη χρήση και την επέκτασή του**. Βασικά του χαρακτηριστικά είναι ότι ενώ παρέχει ένα ευρύ σύνολο από λειτουργίες, **παραμένει ένα εργαλείο που ασχολείται με το πεδίο της επεξεργασίας της φυσικής γλώσσας (NLP) και δεν μετατρέπεται σε σύστημα**.

Το NLTK προορίζεται για να υποστηρίξει την έρευνα και τη διδασκαλία της Επεξεργασίας της Φυσικής Γλώσσας. Συνδέεται στενά με τους τομείς της γλωσσολογίας, της γνωστικής επιστήμης, της τεχνητής νοημοσύνης (AI), της ανάκτησης πληροφοριών (IR) και της μηχανικής μάθησης (Machine Learning).

Για να υλοποιηθούν κάποιες από τις προαναφερθείσες εφαρμογές και άλλες βασικές επεξεργασίες της φυσικής γλώσσας, υπάρχουν πολλά διαθέσιμα εργαλεία. Ορισμένα από αυτά αναπτύσσονται από οργανισμούς (π.χ. Google Cloud Platform / Machine Learning / Cloud Natural Language API) για να δημιουργήσουν τις δικές τους εφαρμογές NLP, ενώ ορισμένες από αυτές είναι **ανοιχτές όπως το NLTK**. Τα περισσότερα από τα εργαλεία είναι γραμμένα σε Java και έχουν παρόμοιες λειτουργίες. Ορισμένα από αυτά είναι ισχυρά και διαθέτουν διαφορετική ποικιλία εργαλείων NLP.

Ωστόσο, όταν πρόκειται για την **ευκολία χρήσης** και την εξήγηση των εννοιών, τα αποτελέσματα του NLTK είναι πολύ θετικά. Το NLTK είναι επίσης πολύ καλό για την εκμάθηση επειδή **η καμπύλη μάθησης της Python (στην οποία είναι γραμμένο το NLTK) είναι πολύ γρήγορη**. Το NLTK έχει ενσωματώσει τις περισσότερες εργασίες της Επεξεργασίας της Φυσικής Γλώσσας, είναι πολύ κομψό και εύκολο στη χρήση. Για όλους

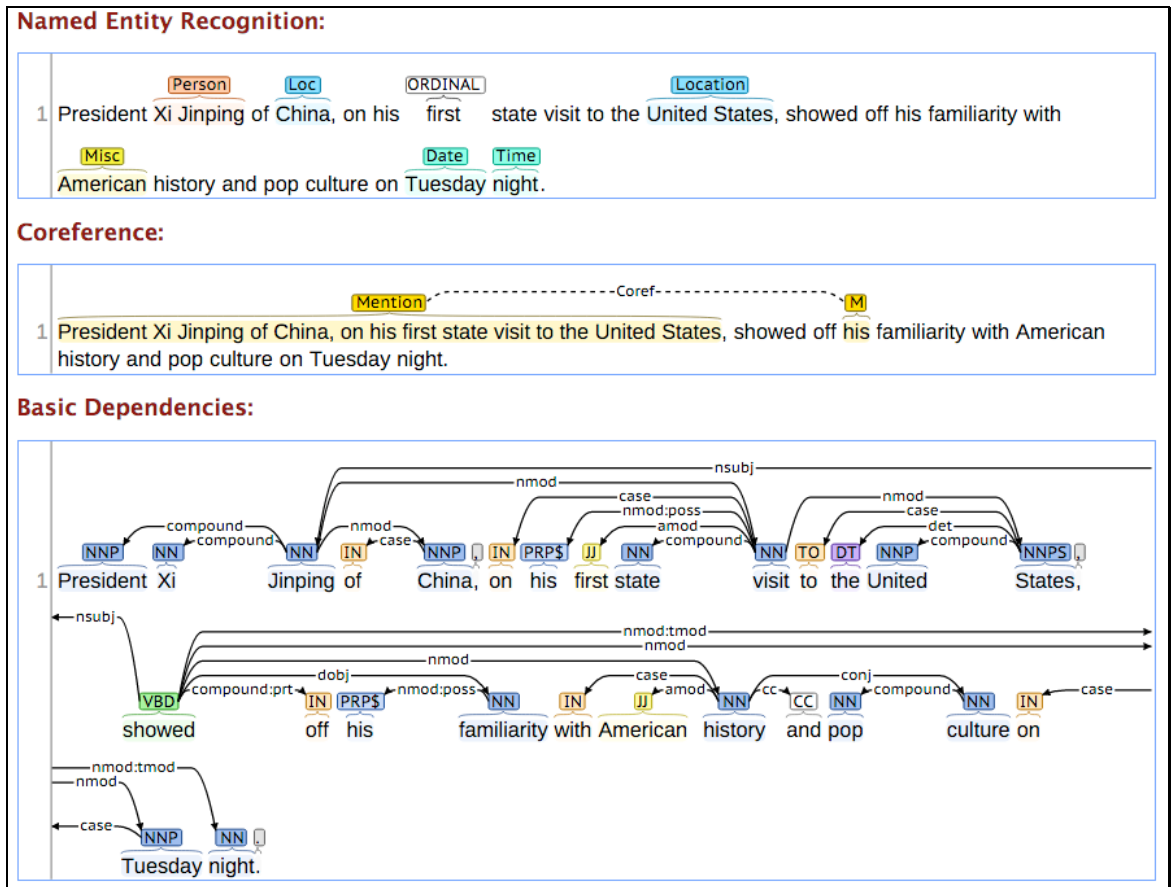
αυτούς τους λόγους, το NLTK έχει γίνει μια από τις πιο δημοφιλείς βιβλιοθήκες της κοινότητας και για αυτό την επέλεξα για την ανάπτυξη της εργασίας μου.

5.3.1 Εγκατάσταση του NLTK για Windows

Αρχικά κατεβάζουμε και εγκαθιστούμε την 32-bit έκδοση της Python 3.5 εάν έχουμε λειτουργικό σύστημα Windows αλλιώς την 64-bit, κατά την εγκατάσταση επιλέγουμε την επιλογή 'Path' για να δημιουργηθεί μια μεταβλητή περιβάλλοντος Python στο Windows συστήμα μας . Αφού η εγκατάσταση ολοκληρωθεί ανοίγουμε το Command Prompt και εκτελούμε την εντολή `'python -m pip install nltk'`, η οποία καλεί το πακέτο διαχείρισης πακέτων που χρησιμοποιείται για την εγκατάσταση και διαχείριση πακέτων λογισμικού γραμμένα σε Python στην περίπτωση μας το NLTK. Εάν η εντολή δεν τρέξει τότε θα χρειαστεί να εγκαταστήσουμε το πακέτο pip χειροκίνητα το οποίο θα βρούμε online με την ονομασία `'get-pip.py'` και θα επαναλάβουμε την εκτέλεση της προηγούμενης εντολής. Τέλος, ανοίγουμε το τερματικό της Python στο σύστημα μας και τρέχουμε την εντολή `'import nltk'` για να ελέγχουμε εάν έχει εγκατασταθεί σωστά εάν δεν δούμε τίποτα τότε ισχύει η ρήση του Linux καθόλου νέα καλά νέα (no news good news) και το έχουμε εγκαταστήσει σωστά. Η έκδοση που χρησιμοποίησα για την εργασία μου είναι η `nltk-3.2.5`.

5.4 Βιβλιοθήκη Stanford CoreNLP

Το Stanford CoreNLP (Manning, et al., 2014) παρέχει ένα σύνολο τεχνολογικών εργαλείων που έχουν να κάνουν με την ανθρώπινη γλώσσα. Μπορεί να μας δώσει την ρίζα μιας λέξης, τα μέρη το λόγου (πχ. ρήμα, ουσιαστικό κοκ.), αναγνωρίζει οντότητες όπως εταιρείες, ανθρώπους, ημερομηνίες, ώρες και άλλα. Επισημάνει τη δομή μιας πρότασης σε φράσεις και βρίσκει τις συντακτικές εξαρτήσεις. Ο στόχος του είναι να **κάνει πολύ εύκολη την εφαρμογή την γλωσσολογικής ανάλυση ενός κειμένου**. Το CoreNLP έχει σχεδιαστεί για να είναι **ιδιαίτερα ευέλικτο και επεκτάσιμο**. Το Stanford CoreNLP ενσωματώνει πολλά εργαλεία NLP, συμπεριλαμβανομένου του tagger για τα τμήμα ομιλίας (POS), την αναγνώρισή ονομαστικών οντοτήτων (NER), του parser, του συστήματος ανάλυσης coreference, της ανάλυσης συναίσθημα και την εξαγωγής πληροφοριών.



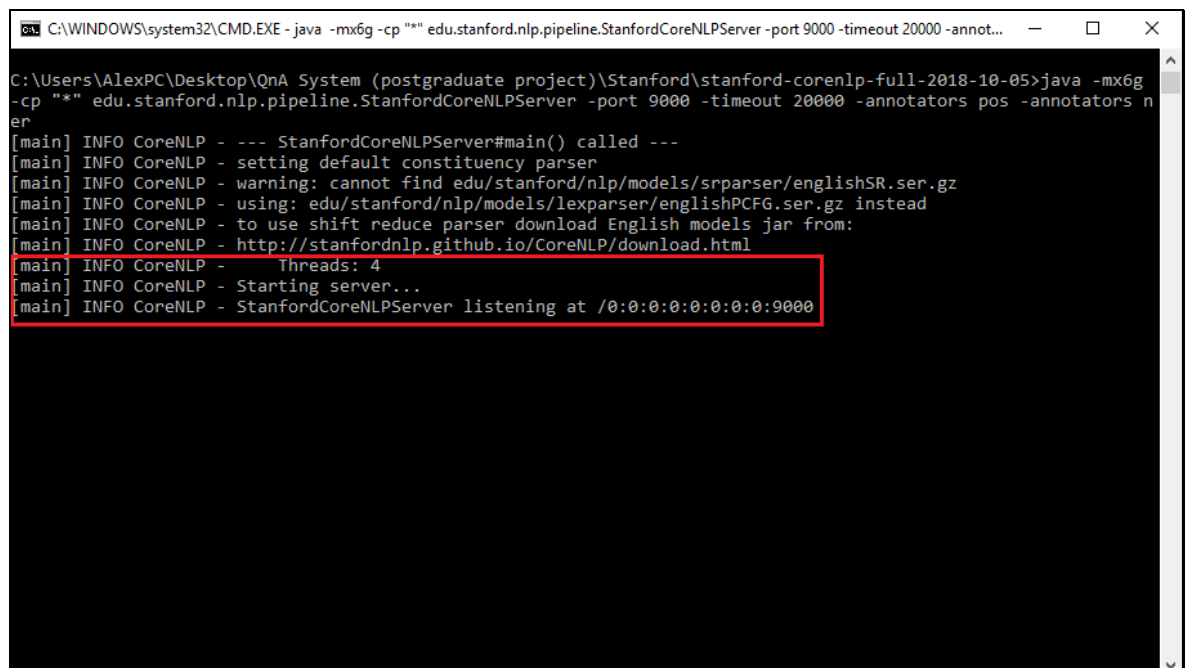
Εικόνα 3: Παραδείγματα επισήμανσης του Stanford CoreNLP.

Η βασική διανομή παρέχει αρχεία μοντέλων για την ανάλυση καλά επεξεργασμένων αγγλικών, αλλά είναι συμβατός και με μοντέλα για άλλες γλώσσες όπως αραβικά, τα κινέζικα, τα γαλλικά, τα γερμανικά και τα ισπανικά. Στην εργασία μου όλες οι είσοδοι θα είναι στα Αγγλικά οπότε **θα χρησιμοποιήσω το Αγγλικό πακέτο του Stanford CoreNLP**. Η χρήση του θα γίνει μέσω ενός ειδικού wrapper που έχει η NLTK βιβλιοθήκη και χρειάζεται Java 1.8+ έκδοση για να τρέξει (το CoreNLP) μέσω NLTK που τρέχει σε Python.

5.4.1 Εγκατάσταση του CoreNLP Server

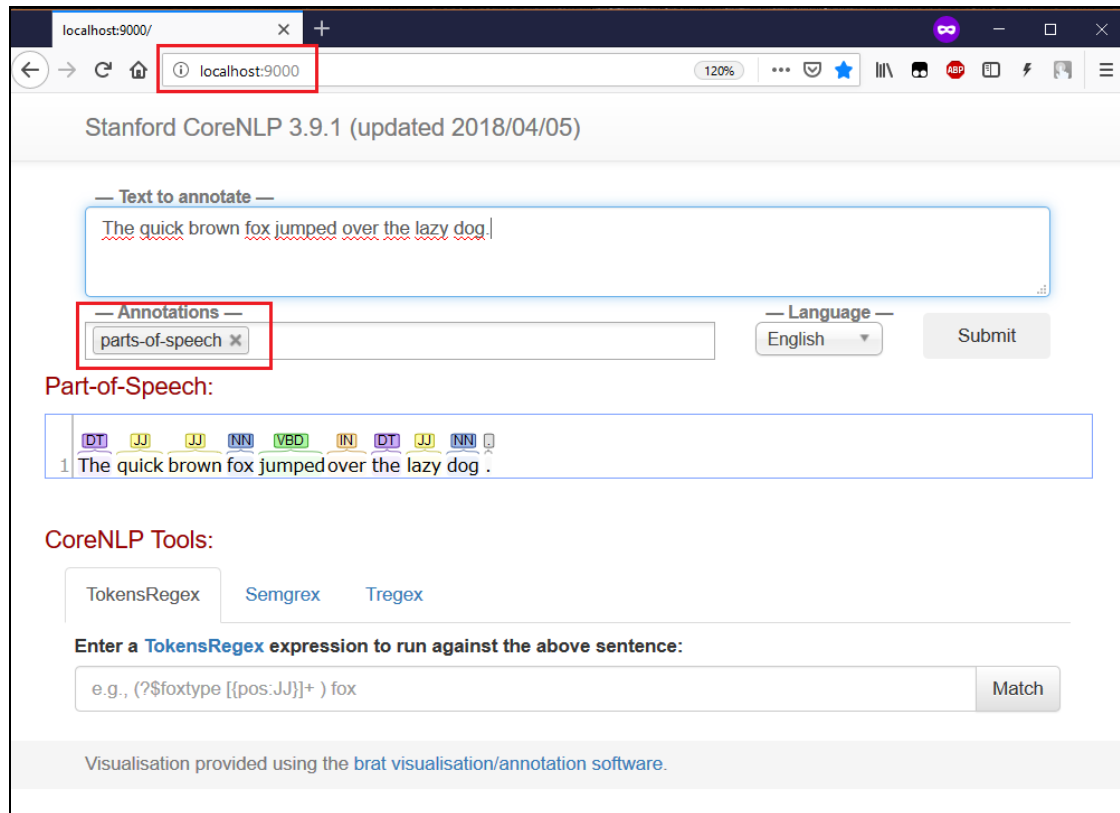
Για την χρήση του CoreNLP αρχικά θα χρειαστεί να κατεβάσουμε την τελευταία έκδοση του CoreNLP από την επίσημη σελίδα του Stanford: <https://stanfordnlp.github.io/>, για την εργασία μου **χρησιμοποίησα την έκδοση 3.9.2 (stanford-corenlp-full-2018-10-05)** και να κάνουμε αποσυμπίεσή του φακέλου. Έπειτα πρέπει να κατεβάσουμε μια έκδοση της

γλώσσας Java μεγαλύτερη από την 1.8 από την επίσημη σελίδα: <https://www.java.com/en/download/> ανάλογα με το λειτουργικό σύστημα που έχουμε για να μπορέσουμε αργότερα να τρέξουμε τον Server του CoreNLP, για την εργασία μου **χρησιμοποίησα την έκδοση Java 8 update 201 (jdk1.8.0_112)**. Τέλος, για να λειτουργήσει ο Server ανοίγουμε το Command Prompt και τρέχουμε την εντολή (βλέπε **Εικόνα 3**): **'java -mx6g -cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer -port 9000 -timeout 20000 -annotators pos -annotators ner'** στον φάκελο του CoreNLP που αποσυμπίεσαμε πιο πριν, ουσιαστικά η εντολή αυτή καλεί την Java με μέγιστη(-mx) δέσμευση μνήμης τα 6GB γενικά χρειαζόμαστε αρκετή μνήμη για την επεξεργασία μεγάλων προτάσεων ώστε να μην μας 'σκάσει' το πρόγραμμα, και εκκινεί τον StanfordCoreNLPServer στην θύρα (port) 9000 με μέγιστο όριο τα 20000 χιλιοστά του δευτερολέπτου (milliseconds) για κάθε request και την χρήση annotators για POS και NER που θα. Εάν όλα έχουν γίνει σωστά θα δούμε στην σελίδα **localhost: 9000** του Browser μας στην **Εικόνα 4**.

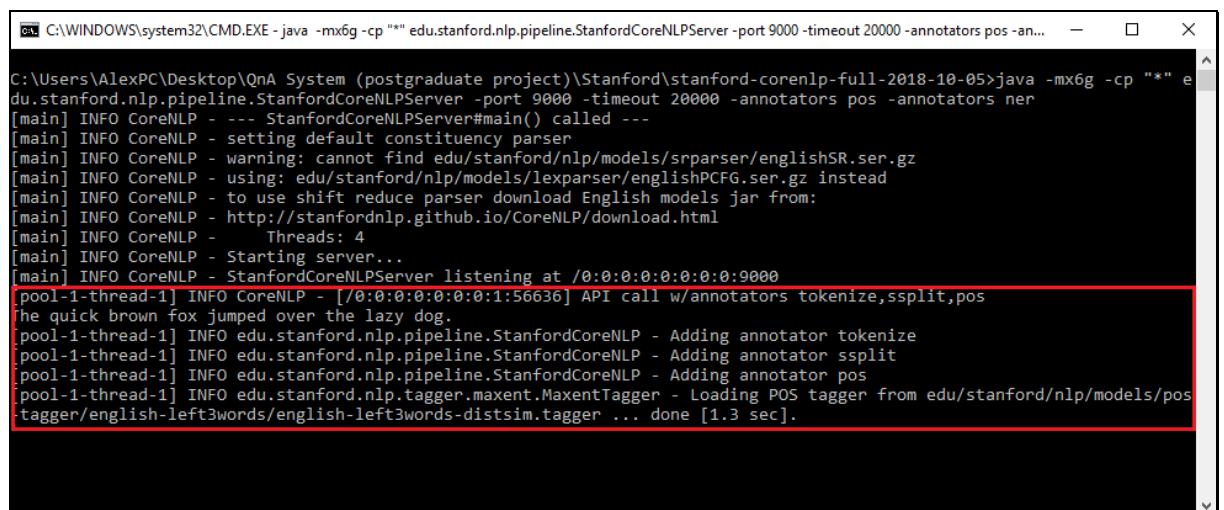


```
C:\WINDOWS\system32\CMD.EXE - java -mx6g -cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer -port 9000 -timeout 20000 -annot...
C:\Users\AlexPC\Desktop\QnA System (postgraduate project)\Stanford\stanford-corenlp-full-2018-10-05>java -mx6g
-cp "*" edu.stanford.nlp.pipeline.StanfordCoreNLPServer -port 9000 -timeout 20000 -annotators pos -annotators n
er
[main] INFO CoreNLP - --- StanfordCoreNLPServer#main() called ---
[main] INFO CoreNLP - setting default constituency parser
[main] INFO CoreNLP - warning: cannot find edu/stanford/nlp/models/srparser/englishSR.ser.gz
[main] INFO CoreNLP - using: edu/stanford/nlp/models/lexparser/englishPCFG.ser.gz instead
[main] INFO CoreNLP - to use shift reduce parser download English models jar from:
[main] INFO CoreNLP - http://stanfordnlp.github.io/CoreNLP/download.html
[main] INFO CoreNLP - Threads: 4
[main] INFO CoreNLP - Starting server...
[main] INFO CoreNLP - StanfordCoreNLPServer listening at /0:0:0:0:0:0:0:0:9000
```

Εικόνα 4: Εκκίνηση του Server μέσω του τερματικού.



Εικόνα 5: Στην σελίδα localhost:9000 του Browser μας τρέχει ο CoreNLP Server.



Εικόνα 6: Βλέπουμε ότι ο Server εκτελεί Part of Speech για την πρόταση της προηγούμενης Εικόνας.

Στην συνέχεια θα αναλύσω πως γίνεται η κλήση του CoreNLP Server στο πρόγραμμα μου μέσω της βιβλιοθήκης NLTK. Αξίζει να αναφερθεί ότι υπάρχει και άλλος τρόπος να

χρησιμοποιήσουμε απευθείας τον parser του Stanford με πιο απλό τρόπο ο οποίος όμως στις επόμενες εκδόσεις του NLTK θα καταργηθεί. Η λύση του **Server** αν και είναι λίγο χρονοβόρα μας λύνει τα χέρια διότι αφενός **θα δουλεύει και για τις επόμενες εκδόσεις του NLTK αλλά κάνει και πολύ πιο εύκολο η χρήση του μέσω του Browser μας για testing.**

5.5 Βιβλιοθήκη TextBlob

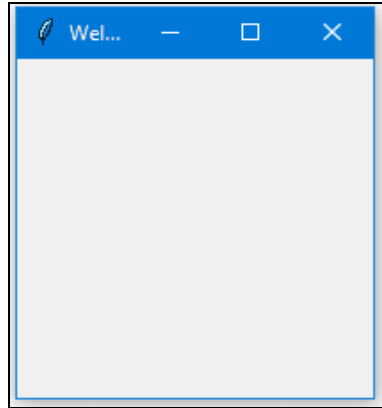
Το TextBlob είναι μια βιβλιοθήκη για Python 2.x και Python 3.x για την **επεξεργασία δεδομένων κειμένου**. Παρέχει ένα απλό API για κοινές εργασίες επεξεργασίας φυσικής γλώσσας (NLP), όπως part-of-speech tagging, εξαγωγή ουσιαστικών (noun) φράσεων, ανάλυση συναισθημάτων, ταξινόμηση, μετάφραση, ταξινόμηση κειμένων, γλωσσική μετάφραση και ανίχνευση που υποστηρίζεται από το Google Translate, διαχωρισμός κειμένου σε λέξεις και προτάσεις, συχνότητες λέξεων και φράσεων, καμπύλη λέξεων και λεμματοποίηση, διόρθωση ορθογραφίας και αλλά πολλά.

5.6 Γραφικό περιβάλλον χρήστη tKinter

Το tKinter είναι η τυπική βιβλιοθήκη GUI για την Python. Η Python όταν συνδυάζεται με το tKinter παρέχει έναν γρήγορο και εύκολο τρόπο δημιουργίας εφαρμογών GUI. Ακόμα το **tKinter είναι μέρος της Python**, περιλαμβάνεται τώρα σε οποιαδήποτε διανομή Python οπότε δεν χρειάζεται να γίνει λήψη κάποιου ξεχωριστού πακέτου. Έχει πολύ απλό συντακτικό. Δουλεύει σε Windows και Mac είναι cross-platform **δεν απαιτεί μετατροπές για να τρέχει από την μια πλατφόρμα στην άλλη**. Επιπλέον, η Tkinter θα παρέχει τη φυσική εμφάνιση και την αίσθηση της συγκεκριμένης πλατφόρμας στην οποία τρέχει. ακολουθεί ένα απλό παράδειγμα:

```
from Tkinter import *
root = Tk()
root.title("Welcome a simple application")
root.mainloop ()
```

Στην **Εικόνα 7** φαίνεται το GUI του παραπάνω κώδικα. Οι πρώτες 2 γραμμές επιτρέπουν τη δημιουργία ενός πλήρους παραθύρου. Σε σχέση με τον προγραμματισμό Microsoft Foundation Classes (MFC), δεν αμφιβάλει ότι το Tkinter είναι απλό στη χρήση. Η τρίτη γραμμή ορίζει τη λεζάντα του παραθύρου και η τέταρτη το κάνει να εισέλθει στον βρόχο εκδήλωσης.



Εικόνα 7: Ένα απλό παράθυρο σχεδιασμένο με tkinder.

6 Η δομή μιας Ερωτήσεως (Αγγλική γραμματική)

Για την απάντηση μιας ερώτησης πρέπει να καταλάβουμε τα είδη και την δομή που μπορεί να έχει:

- Οι τύποι ερωτήσεων που υπάρχουν
- Κλειστές ερωτήσεις (Closed Questions) - Οι ερωτήσεις ‘τύπου’ Ναι / Όχι (Yes/No)
- Ανοιχτές ερωτήσεις (Open Questions) - Οι ερωτήσεις ‘τύπου’ Wh – ερωτήσεις
- Τι δομή έχει μια ερώτηση

Στις παρακάτω υποενότητες θα αναλυθούν διεξοδικά και θα απαντηθούν τα παραπάνω ερωτήματα.

6.1 Οι τύποι ερωτήσεων που υπάρχουν

Υπάρχουν δύο τύπους ερωτήσεων (Anon., 2018). Τις ερωτήσεις του απαντώνται με ένα Ναι / Όχι και τις Wh-ερωτήσεις.

6.2 Κλειστές ερωτήσεις (Closed Questions) - Οι ερωτήσεις ‘τύπου’ Ναι / Όχι (Yes/No)

Οι Ναι / Όχι ερωτήσεις καλούνται επίσης **κλειστές ερωτήσεις** επειδή υπάρχουν μόνο δύο πιθανές απαντήσεις: Ναι ή Όχι. Όταν σχηματίζεται ένα Ναι / Όχι ερώτημα, πρέπει να περιλαμβάνει ένα από αυτά τα **ρήματα ή κλίσεις αυτών: be** (είναι), **do** (σημασία ανάλογα με την φράση), **have** (έχω) ή ένα μεταβατικό ρήμα (modal verb: **can, could, may, might, shall, should, will, would, must**). Είναι αδύνατο να ζητήσετε ένα Ναι / Όχι ερώτημα χωρίς ένα από αυτά τα ρήματα.

Π.χ. Ερώτηση: Are these islands Greek? Απάντηση: Yes. ή Yes, they are. ή Yes, these islands are Greek.

6.3 Ανοιχτές ερωτήσεις (Open Questions) - Οι ερωτήσεις 'τύπου' Wh – ερωτήσεις

Οι Wh-ερωτήσεις, είναι όλες οι ερωτήσεις οι οποίες ξεκινούν με λέξεις (της Αγγλικής γλώσσας) που αρχίζουν με -Wh και ονομάζονται ανοικτές ερωτήσεις επειδή ο αριθμός των πιθανών απαντήσεων είναι απεριόριστος, άρα πρέπει να απαντηθούν με περισσότερες πληροφορίες από απλά "ναι" ή "όχι".

Οι **-wh** λέξεις που υποστηρίζονται από το πρόγραμμα είναι οι εξής: when (πότε), how long (πόσο καιρό), how many (πόσα), how much (πόσο), how old (πόσο χρονών), who (ποιος), whom (ποιόν), whose (τίνος ή ποιανού), where (που), what (τι), how (πώς), why (γιατί) και which (ποιο). Χρησιμοποιούμε τις λέξεις when, how long, how many, how much και how old όταν ρωτάμε για κάτι σχετικό με **ημερομηνία , ώρα ή κάποιο αριθμό**. Χρησιμοποιούμε τις λέξεις who, whom και whose όταν ρωτάμε κάτι σχετικό με την **ταυτότητα κάποιου**. Χρησιμοποιούμε τη λέξη where όταν ρωτάμε κάτι σχετικό με κάποια **τοποθεσία, πόλη θέση ή χώρα** (στην γενική περίπτωση). Χρησιμοποιούμε τις λέξεις what, how, why και which όταν ρωτάμε για κάτι που χρειάζεται περεταίρω διερεύνηση ή μέθοδο ή διεργασία ή εξήγηση.

Π.χ. Ερώτηση: When does Anna arrive? Απάντηση: She arrives at 10:30.

6.4 Τι δομή έχει μια ερώτηση

Δομή ερώτησης κλειστού τύπου: Ρήμα (υπο-ενότητα 6.2) + υποκείμενο (πχ. you, he, his, she κ.α.) + ρήμα + (αντικείμενο) + ? .

Π.χ. Does he live in New York? ή Have you seen that film?

Δομή ερώτησης ανοιχτού τύπου: wh λέξη (υπο-ενότητα 6.3) + βοηθητικό ρήμα + υποκείμενο (πχ. you, he, his, she κ.α.) + ρήμα + ? .

Π.χ Who is coming to the party next week? ή How long has your teacher worked at this school?

7 Πως χρησιμοποιώ το Σύστημα Απάντησης Ερωτήσεων (ΣΑΕ)

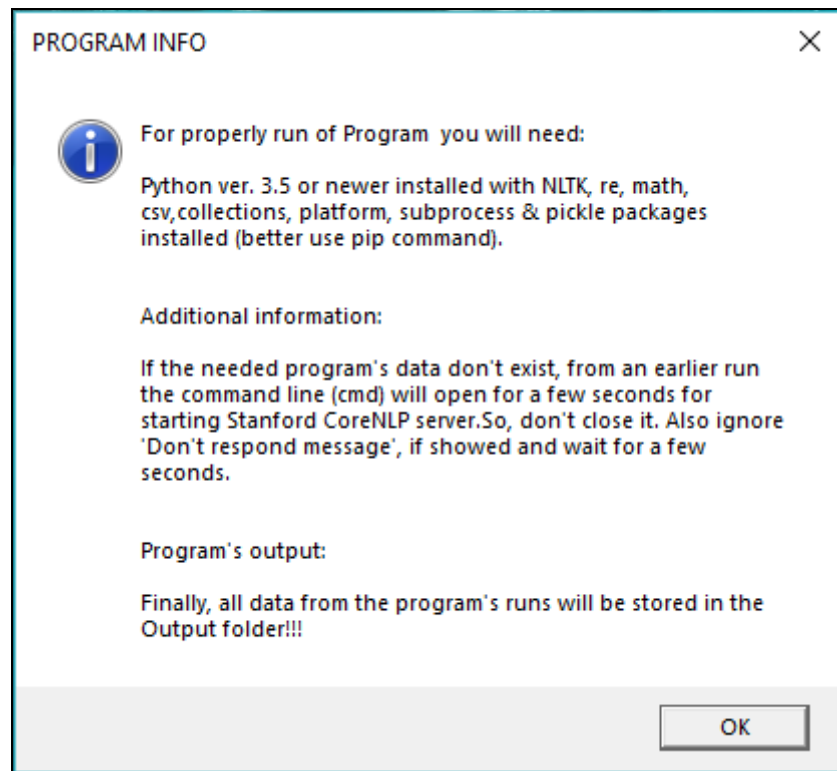
Για να χρησιμοποιηθεί το πρόγραμμα απάντησης ερωτήσεων ο χρήστης θα πρέπει να λάβει υπόψιν του τις οδηγίες των ενοτήτων:

- Αρχική οθόνη προγράμματος
- Παράδειγμα τρεξίματος του προγράμματος

Στις παρακάτω υποενότητες θα αναλυθούν διεξοδικά και θα απαντηθούν τα παραπάνω ερωτήματα.

7.1 Αρχική οθόνη προγράμματος

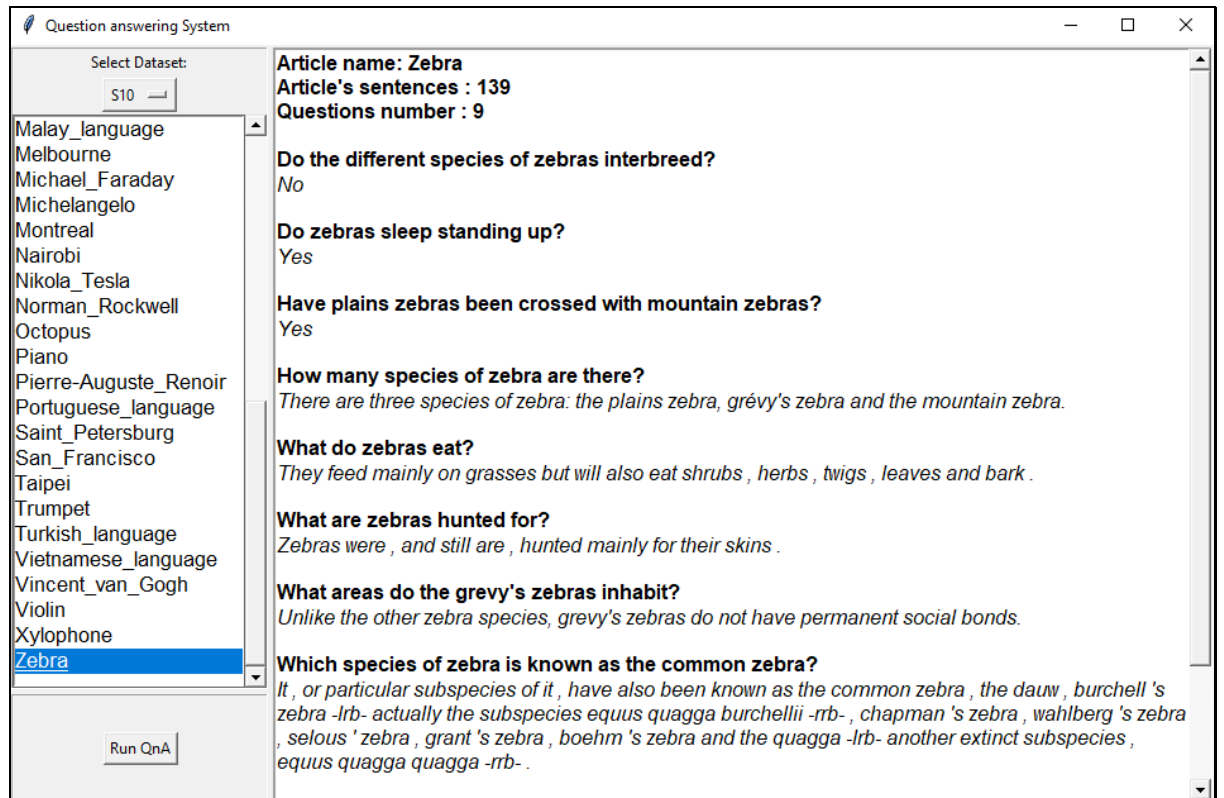
Αφού έχουμε ακολουθήσει τα προηγούμενα βήματα ανοίγουμε τον Φάκελο ‘**QnA System (postgraduate project)**’ και εκτελούμε το αρχείο ‘**QnA_System START.bat**’ το οποίο εκκινεί το γραφικό περιβάλλον του προγράμματος. Αρχικά φαίνεται μια προειδοποίηση (**Εικόνα 8**) η οποία μας ενημερώνει για τα πακέτα τα οποία πρέπει να έχουμε εγκαταστήσει ήδη και υπάρχουν οδηγίες για την εγκατάστασή τους στο **Κεφάλαιο 5** και ότι μπορεί να ανοίξει ένα command line το οποίο θα τρέχει για λίγα δευτερόλεπτα τον CoreNLP Server.



Εικόνα 8: Μήνυμα ειδοποίησης για την σωστή λειτουργία του προγράμματος.

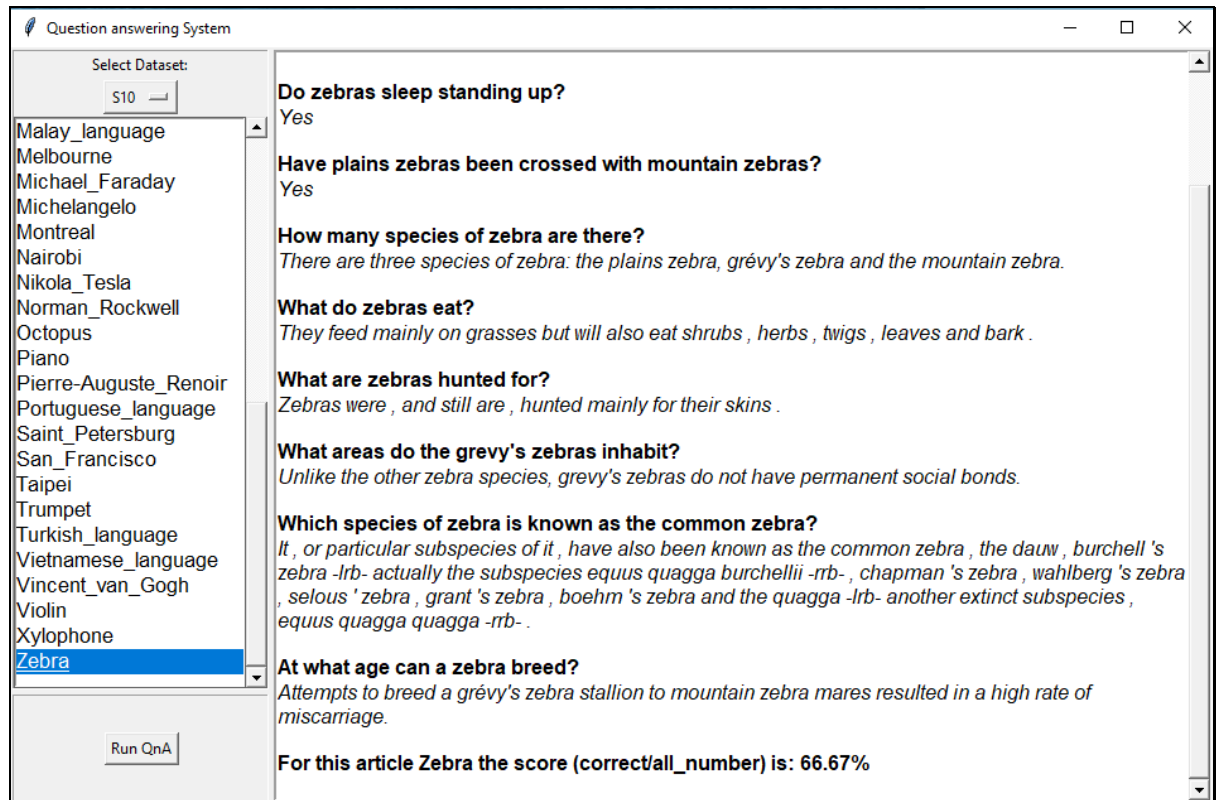
7.2 Παράδειγμα τρεξίματος του προγράμματος

Αφού πατήσουμε OK στην οθόνη με τις προειδοποιήσεις θα μας εμφανιστεί η οθόνη του προγράμματος **Εικόνα 9** η οποία χωρίζεται σε 3 μέρη. Στο **πάνω αριστερά μέρος** υπάρχει μια λίστα με ονόματα αρχείων τα οποία έχουν χρησιμοποιεί για το τεστάρισα της εφαρμογής και αποτελούν τίτλους από άρθρα της Wikipedia του **Dataset S10**, στο **κάτω αριστερό μέρος** υπάρχει το πλήκτρο 'Run QnA' το οποίο εκκινεί την διαδικασία της απάντησης των ερωτήσεων και στο τέλος όλο το **δεξιό μέρος** της οθόνης μας εμφανίζει την έξοδο του προγράμματος για το εκάστοτε άρθρο που έχει επιλεγθεί και πατηθεί το πλήκτρο 'Run QnA'.



Εικόνα 9: Η εκτέλεση του προγράμματος για το άρθρο Zebra (α).

Στις Εικόνες 9 και 10 βλέπουμε ότι έχει επιλεγθεί το άρθρο με όνομα 'Zebra' για να γίνει η εκτέλεση για αυτό και στο δεξιό μέρος της οθόνης βλέπουμε κάποιες πληροφορίες όπως ότι το άρθρο έχει 139 προτάσεις, οι ερωτήσεις που θα κληθεί το πρόγραμμα να απαντήσει είναι 9 στον αριθμό και έπειτα σε κάθε σειρά υπάρχει ένα ζεύγος από ερώτηση - απάντηση συστήματος και στο τέλος μας δείχνει το ποσοστό επιτυχίας δηλαδή τον λόγο των σωστών απαντήσεων ερωτήσεων προς το συνολικό αριθμό των ερωτήσεων σε ποσοστό π.χ. εδώ έχουμε το ποσοστό σωστών απαντήσεων 66,67 %.



Εικόνα 10: Η εκτέλεση του προγράμματος για το άρθρο Zebra (β).

Στα επόμενο Κεφάλαια θα γίνει εκτενής αναφορά στον μηχανισμό που χρησιμοποιείτε για την απάντηση στην κάθε ερώτηση και θα καλυφθούν τα όλα τα κενά που υπάρχουν μέχρι στιγμής.

8 Η κατασκευή - εξήγηση του Συστήματος Απάντησης Ερωτήσεων (ΣΑΕ)

Οι βασικοί κορμοί του συστήματος απάντησης ερωτήσεων στηρίζεται είναι οι εξής:

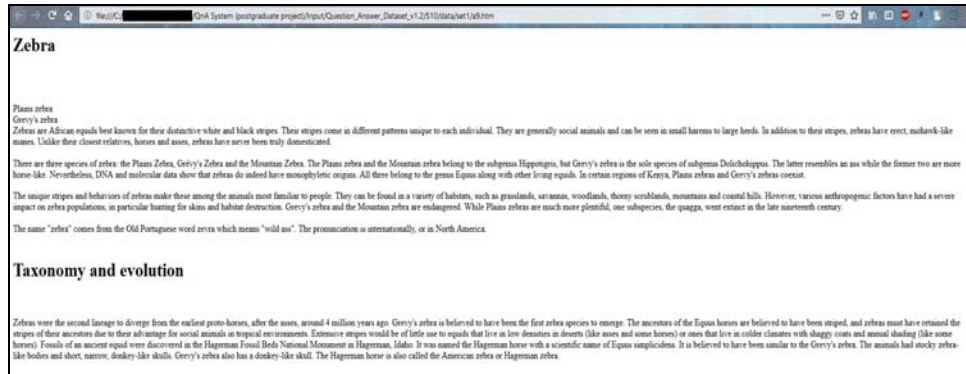
- Οι Είσοδοι του Συστήματος Απάντησης Ερωτήσεων (ΣΑΕ)
- Η βασική προ-επεξεργασία της εισόδου
- Απάντηση ερωτήσεων
- Έξοδος του προγράμματος

Στις παρακάτω υποενότητες θα αναλυθεί διεξοδικά πως έγινε η κατασκευή και η δημιουργία των παραπάνω διεργασιών του προγράμματος.

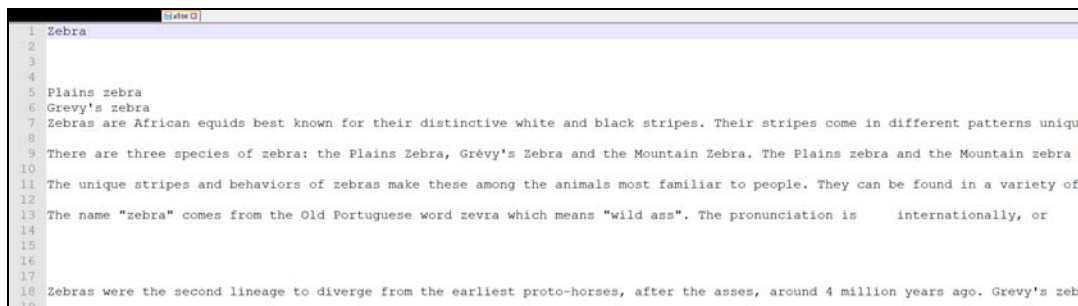
8.1 Οι Είσοδοι του Συστήματος Απάντησης Ερωτήσεων (ΣΑΕ)

Στον Φάκελο **‘Input/Question_Answer_Dataset_v1.2’** θα βρούμε μια ομάδα από Datasets (Smith, et al., 2008) τα οποία έχουν δημιουργηθεί από τον Noah Smith του Πανεπιστημίου του Carnegie Mellon και της Rebecca Hwa του Πανεπιστημίου του Pittsburgh. Ουσιαστικά πρόκειται για **3 datasets το S08, το S09 και το S10** τα οποία δημιουργήθηκαν την Άνοιξη του 2008, 2009 και 2010 αντίστοιχα και το καθένα έχει ένα διαφορετικό σύνολο από Άρθρα τα οποία ο χρήστης μπορεί να επιλέξει.

Ο κάθε φάκελος του κάθε dataset αποτελείται από έναν φάκελο, τον **‘data’** και ένα αρχείο, το **‘question_answer_pairs.txt’**. Στον φάκελο data έχουμε τα sets τα οποία περιέχουν τα αρχεία με τα **άρθρα της Wikipedia** σε μορφή ηλεκτρονικής σελίδας (.htm), κειμένου (.txt) και καθαρού κειμένου (.txt.clean) βλέπε Εικόνες 11 & 12 και το αρχείο **‘question_answer_pairs.txt’** που περιέχει τις ερωτήσεις και τις απαντήσεις.



Εικόνα 11: Το αρχείο a9.htm που αντιστοιχεί στο άρθρο Zebra.



Εικόνα 12: Το αρχείο a9.txt που αντιστοιχεί στο άρθρο Zebra.

Το αρχείο ‘question_answer_pairs.txt’ στην πρώτη του σειρά (Εικόνα 13) έχουμε την κεφαλίδα του η οποία περιέχει:

- Το όνομα του άρθρου (**ArticleTitle**) της Wikipedia από τις οποίες έχουν προκύψει οι ερωτήσεις και οι απαντήσεις.
- Την ερώτηση (**Question**),
- Την απάντηση (**Answer**),
- Τον βαθμό δυσκολίας από τον ερωτών (**DifficultyFromQuestioner**) δηλαδή από το άτομο που κλήθηκε να γράψει την ερώτηση.
- Τον βαθμό δυσκολίας από το άτομο που έχει κληθεί να αξιολόγησει και να απάντησε την ερώτηση (**DifficultyFromAnswerer**), ο οποίος βαθμός μπορεί να διαφέρει από τον DifficultyFromQuestioner.

- Την απόλυτη διαδρομή του άρθρου (**ArticleFile**) χωρίς την κατάληξη .htm, .txt ή .txt.clean ώστε να μπορεί να χρησιμοποιηθεί όποια κατάληξη χρειάζεται κατά το δοκούν.

1 ArticleTitle Question Answer DifficultyFromQuestioner DifficultyFromAnswerer ArticleFile
--

Εικόνα 13: Η κεφαλίδα του αρχείου question_answer_pairs.txt

Στην Εικόνα 14 υπάρχει ένα παράδειγμα για το πώς είναι το αρχείο με τις ερωτήσεις για το άρθρο με τίτλο Zebra του Dataset S10. Μπορούμε να διακρίνουμε ότι κάποιες ερώτησης εμφανίζονται δις, αυτό προκύπτει συνήθως λόγω της απάντησής η οποία παρότι είναι μια μπορεί να ερμηνευτεί με 2 τρόπους, στο πρόγραμμα εμφανίζονται μόνο μια φορά όμως.

```
1161 Zebra|Do the different species of zebras interbreed?|No|easy|easy|data/set1/a9
1162 Zebra|Do zebras sleep standing up?|yes|easy|easy|data/set1/a9
1163 Zebra|Do zebras sleep standing up?|Yes|easy|easy|data/set1/a9
1164 Zebra|Have plains zebras been crossed with mountain zebras?|yes|easy|easy|data/set1/a9
1165 Zebra|Have plains zebras been crossed with mountain zebras?|Yes|easy|easy|data/set1/a9
1166 Zebra|How many species of zebra are there?|three|medium|medium|data/set1/a9
1167 Zebra|How many species of zebra are there?|Three|medium|medium|data/set1/a9
1168 Zebra|What do zebras eat?|mainly grass|medium|hard|data/set1/a9
1169 Zebra|What do zebras eat?|Grasses, shrubs, herbs, twigs, leaves, and bark|medium|hard|data/set1/a9
1170 Zebra|What are zebras hunted for?|mainly for their skins|medium|medium|data/set1/a9
1171 Zebra|What are zebras hunted for?|Skins|medium|medium|data/set1/a9
1172 Zebra|What areas do the Grevy's Zebras inhabit?|semi-arid grasslands of Ethiopia and northern Kenya|hard|hard|da
1173 Zebra|What areas do the Grevy's Zebras inhabit?||hard||data/set1/a9
1174 Zebra|Which species of zebra is known as the common zebra?|Plains Zebra (Equus quagga, formerly Equus burchelli)
1175 Zebra|Which species of zebra is known as the common zebra?|Plains Zebra|hard|medium|data/set1/a9
1176 Zebra|At what age can a zebra breed?|five or six|hard|medium|data/set1/a9
1177 Zebra|At what age can a zebra breed?|5 or 6|hard|hard|data/set1/a9
```

Εικόνα 14: Το περιεχόμενο του αρχείου question_answer_pairs.txt

8.2 Η βασική προ-επεξεργασία της εισόδου

Αφού ο χρήστης έχει επιλέξει ένα συγκεκριμένο άρθρο π.χ. το **Zebra**, το σύστημα ελέγχει εάν έχει εκτελεστεί ήδη, από παλιότερη εκτελεστή στο QnA_System.py και διαβάζει τα απαραίτητα δεδομένα μέσω του data_parsing.py και της συνάρτησης read_parse_data αλλιώς ξεκινάει η απαραίτητη προ-επεξεργασία.

Ξεκινάμε με την δημιουργία των **συντακτικών δένδρων** για κάθε μια **πρόταση** του αρχικού κείμενου του άρθρου με την βοήθεια του CoreNPL, αρχείο data_parsing.py συνάρτηση create_parse_trees και χρήση του CoreNLPOSTagger (εκτελεί tokenization, tagging και δημιουργία συντακτικού δένδρου) το οποίο προαναφέραμε σε προηγούμενη

ενότητα και αντίστοιχα για κάθε **ερώτηση** του άρθρου. Έπειτα **βρισκω όλες τις οντότητες (NERs)** της κάθε πρότασης του κειμένου και τις αποθηκεύω για μελλοντική χρήση με τη βοήθεια του πάλι του CoreNLP (αρχείο `data_parsing.py` συνάρτηση `find_all_ner` και χρήση του `CoreNLPNERTagger`).

Αφού έχουν γίνει οι δυο (2) απαραίτητες διεργασίες αποθηκεύω τα αποτελέσματα, αρχείο `data_parsing.py` συνάρτηση `write_parse_data` τους στον φάκελο **Output/[το όνομα του αντίστοιχου Dataset]/[το όνομα του αντίστοιχου Άρθρου]** με τα ονόματα:

- **context_parsed_trees.txt**: pickle (δυαδικό αρχείο) με την λίστα των συντακτικών δέντρων της κάθε πρότασης του άρθρου, πιο συγκεκριμένα είναι μια λίστα με δέντρα όπου το κάθε δέντρο αντιστοιχεί σε μια πρόταση του άρθρου, όπου το κάθε υπο δέντρο της (πρότασης) αποτελείτε από δυάδες (ετικέτα μέρος του λόγου, λέξη).
- **context_sentences_ners.txt**: pickle (δυαδικό αρχείο) με την λίστα των οντοτήτων της κάθε πρότασης του άρθρου, όπου κάθε πρόταση είναι μια λίστα από δυάδες (λέξη, NER ετικέτα).
- **questions_parsed_trees.txt**: pickle (δυαδικό αρχείο) με την λίστα των συντακτικών δέντρων της κάθε ερωτήσεως του άρθρου, πιο συγκεκριμένα είναι μια λίστα με δέντρα όπου το κάθε δέντρο αντιστοιχεί σε μια πρόταση του άρθρου, όπου το κάθε υπο δέντρο της (πρότασης) αποτελείτε από δυάδες (ετικέτα μέρος του λόγου, λέξη).

Είναι προτιμότερο η προ-επεξεργασία να γίνει μια και καλή πριν αρχίσει η διαδικασία της απάντησης των ερωτήσεων και η αποθήκευση των δεδομένων της (προ-επεξεργασίας) ώστε να κερδίζουμε χρόνο στις επανεκτελέσεις.

8.3 Απάντηση ερωτήσεων

Αρχικά γίνεται ένα **φιλτράρισμα της κάθε ερώτησης** ανάλογα με το είδος της ερωτήσεως γενικά εξετάζουμε τρεις (3) μεγάλες κατηγορίες: Δυαδικές ερωτήσεις (Ναι/Όχι), Ερωτήσεις που έχουν να κάνουν με οντότητες (NERs) και Όλες οι υπόλοιπες Ερωτήσεις.

8.3.1 Δυαδικές ερωτήσεις (Ναι/Όχι)

Δυαδικές ερωτήσεις (Ναι/Όχι) είναι οι ερωτήσεις στις οποίες **απαντούμε με Ναι ή Όχι**. Έχουμε δυο (2) μεγάλες υποκατηγορίες τις **Καταφατικές** και της **Αρνητικές** ερωτήσεις. Γίνεται έλεγχος για το εάν η ερώτηση είναι Καταφατική ή Αρνητική στο αρχείο question_answering.py και την **συνάρτηση yes_no** η οποία εξετάζει εάν έχω Καταφατική ή Αρνητική ερώτηση ελέγχοντας εάν υπάρχει κάποιο ρήμα από τα αρχεία **‘Yes No Words (18).txt’** ή **‘Negative Yes No Words (15).txt’** αντίστοιχα (linguistic – γλωσσολογική προσέγγιση) του φάκελου Utilities. Αυτή είναι μια σημαντική εργασία διότι στην περίπτωση της Αρνητικής ερώτησης θα χρειαστεί αντιστροφή της απάντησης της ερώτησης.

Π.χ. Are they working hard? -> Καταφατική Ερώτηση

Π.χ. Aren't you coming? -> Αρνητική Ερώτηση

Αξίζει να σημειωθεί πως υπάρχει ένα μεγάλο **σημασιολογικό ζήτημα στις Αρνητικές ερωτήσεις** δηλαδή ένα κάποιος ρωτήσει **‘Δεν έχεις αδέρφια ?’** μπορεί να έχει σαν απάντηση το **Ναι** **οπότε δεν υπάρχει θέμα** που σημαίνει **‘Ναι δεν έχω (άρα είμαι μοναχοπαίδι)’** ενώ το **Όχι είναι διφορούμενο** που μπορεί να ερμηνευτεί σαν **‘Όχι δεν έχω αδέρφια (άρα είμαι μοναχοπαίδι).’** ή **‘Όχι έχω αδέρφια (άρα δεν είμαι μοναχοπαίδι).’**. Αυτό μπορεί να ερμηνευτεί μόνο από τα συμφραζόμενα μια συζήτησης ή ενός κειμένου ή από τον τόνο της φωνής αυτού που απαντά την ερώτηση.

Εδώ λοιπόν κάνω την **ΠΑΡΑΔΟΧΗ** σαν δημιουργός-σχεδιαστής του προγράμματος που για εμένα είναι **‘λογική’** (καθαρά υποκειμενικό και έχει να κάνει με την δική μου λογική) να **μετατρέπω όλες τις ερωτήσεις σε Καταφατικές** στις μεν **πραγματικά καταφατικές να κρατάω το αποτέλεσμα** είτε Ναι είτε Όχι ως έχει ενώ στις **αρχικά Αρνητικές να το αντιστρέφω** δηλαδή να αλλάξω το Ναι σε Όχι και το Όχι σε Ναι.

Αφού έχω εξετάσει εάν έχω Καταφατική ή Αρνητική ερώτηση καλώ την συνάρτηση binary_question_answering από το αρχείο question_answering.py οπού **μετατρέπω την ερώτηση (ανεξαρτήτως εάν είναι Καταφατική ή Αρνητική) σε δήλωση (δηλαδή Κατάφαση)** και εξετάζω εάν αυτή η δήλωση υπάρχει σε κάποια από τις προτάσεις του άρθρου. **Εάν υπάρχει η απάντηση είναι Ναι αλλιώς Όχι** και δημιουργώ επίσης ένα θετικό

ή αρνητικό κείμενο απάντησης ακόμα περνάω σας **παράμετρο negative_flag** εάν True έχουμε μια Αρνητική δυαδική ερώτηση, ώστε να αντιστραφεί η απάντηση.

Η διεργασία της απάντησης έχει να κάνει με την **μετατροπή της ερωτήσεως σε δήλωση** αυτό επιτυγχάνεται με το φόρτωμα του αρχείου 'Negative Yes No Words (15).txt' που προανάφερα, της **αφαίρεσης του βοηθητικού ρήματος** (έχουμε αναφερθεί σε αυτό σε προηγούμενο κεφάλαιο), την **αφαίρεση του χαρακτήρα του ερωτηματικού**, την εφαρμογή **tokenization** (δηλαδή σπάσιμο της πρότασης σε λίστα λέξεων), την **αφαίρεση των stopwords** (δηλαδή την αφαίρεση τετριμμένων λέξεων και λέξεων χωρίς ουσιαστικό νόημα) και τέλος την εφαρμογή του **SnowballStemmer** (δηλαδή του δίνω μια λέξη και παίρνω τον κορμό της λήξης π.χ. την λέξη morphological την μετατρέπει σε morpholog). Όπως παρατηρείτε και εδώ εφαρμόζουμε linguistic – γλωσσολογική προσέγγιση.

Μόλις η ερώτηση γίνει δήλωση εφαρμόζουμε **πανόμοια επεξεργασία και στις προτάσεις του άρθρου** και ψάχνουμε για το εάν **όλα τα tokens της δήλωσης είναι υποσύνολο των tokens κάποιας πρότασης του άρθρου** τότε υπάρχει απάντηση και είναι Ναι εάν όμως υπάρχω άρνηση (λέξη που ανήκει στο αρχείο 'Negative Yes No Words (15).txt' που φορτώσαμε στην αρχή) στα **tokens της πρότασης του άρθρου** τότε η απάντηση είναι Όχι. Τέλος εάν **χρειάζεται κάνω αντιστροφή απάντησης και επιστρέφω την απάντηση (Ναι ή Όχι) μαζί με ένα μήνυμα** (την δήλωση δηλαδή τροποποιημένη με Yes στην αρχή της ή No ανάλογα με την απάντηση).

8.3.2 Ερωτήσεις που έχουν να κάνουν με οντότητες (NERs)

Οι ερωτήσεις που έχουν να κάνουν με οντότητες (NERs) είναι αυτές που έχουν σαν απάντηση τους μια ημερομηνία, μια ώρα, έναν αριθμό, ένα πρόσωπο, μια τοποθεσία, μια πόλη ή μια χώρα είναι δηλαδή οι **ερωτήσεις που έχουν να κάνουν με τις ερωτήσεις που περιχέουν τις λέξεις when, how long, how many, how much, how old, who, whom, whose και where.**

Οι ερωτήσεις αυτές αρχικά φιλτράρονται στο αρχείο QnA_System.py και καλείτε η **συνάρτηση wh_familiar από το αρχείο question_answering.py.** Ανάλογα με την ερώτηση καλούμε και την συνάρτηση wh_familiar με τα **ανάλογα NER tags (ετικέτες) σαν παραμέτρους.**

Για της ερωτήσεις που έχουν να κάνουν με τις λέξεις **when, how long, how many, how much και how old** έχουμε τις παραμέτρους **DATE, TIME, NUMBER και ORDINAL**, με τις λέξεις **who, whom και whose** έχουμε την παράμετρο **PERSON** και για την λέξη **where** έχουμε τις παραμέτρους **LOCATION, CITY, STATE_OR_PROVINCE και COUNTRY**. Οι λίστες με τις ετικέτες που δίνονται σαν παράμετρος έχουν να κάνουν με το τι είδους οντότητα είναι η απάντηση της εκάστοτε ερώτηση.

Στη συνάρτηση **wh_familiar** πρώτα ψάχνω για ταυτίσεις - ταιριάσματα ανάμεσα στην **λίστα context_sentences_ners** (που έφτιαξα στην προ-επεξεργασία) και τα **NER tags** που ζητάει η ερώτηση και τα αποθηκεύω σε μια λίστα αυτή με τις **πιθανές απαντήσεις ανεξαρτήτου ερώτησης**. Δηλαδή εάν η ερώτηση μου έχει την λέξη who ψάχνω για οντότητες PERSON στην λίστα context_sentences_ners που έχουν σαν tag το PERSON και αποθηκεύω το/τα ζευγάρια **αριθμό πρότασης, λέξη/εις** (που αντιστοιχούν στην οντότητα που ψάχνω) .

Π.χ. Εάν ψάχνω για *NER tag = PERSON* μπορεί να έχω αυτά τα πιθανά αποτελέσματα: *[[4, (George, Bush)] , [10, (Kennedy)]]* δηλαδή στην 4^η πρόταση του άρθρου να έχω **George, Bush** & στην 10^η πρόταση του άρθρου την **Kennedy** σαν πιθανές απαντήσεις.

Οπότε τα σενάρια των ταυτίσεων – πιθανών απαντήσεων που επεξεργάζομαι είναι τρία (3): να υπάρχει μια (1) ταύτιση, από μια και πάνω (>1) ταυτίσεις και καμία δηλαδή μηδέν (0) ταυτίσεις.

Εάν η **ταύτιση είναι μονό μια (1)** τότε αυτή είναι και η απάντηση και **την επιστρέφω** ενώ εάν έχω **μηδέν (0) ταυτίσεις τότε επιστρέφω None** και έπειτα καλείτε από την QnA_System.py η συνάρτηση wh_questions του question_answering.py η οποία θα αναπτυχθεί στην επόμενη υποενότητα.

Εάν όμως έχω **από μια και πάνω (>1) ταυτίσεις** τότε πρέπει **επεξεργαστώ την ερώτηση** δηλαδή να εφαρμόσω την αφαίρεση του χαρακτήρα του ερωτηματικού, την εφαρμογή **tokenization** (δηλαδή σπάσιμο της πρότασης σε λίστα λέξεων), την **αφαίρεση των stopwords** (δηλαδή την αφαίρεση τετριμμένων λέξεων και λέξεων χωρίς ουσιαστικό νόημα) και τέλος την εφαρμογή του **SnowballStemmer** (δηλαδή του δίνω μια λέξη και παίρνω τον κορμό της λήξης π.χ. την λέξη morphological την μετατρέπει σε morpholog) και

έπειτα να εξετάσω τις προτάσεις των πιθανών απαντήσεων μια μέχρι να βρω το καθαρό υποσύνολο της ερωτήσεως σε μια από αυτές των πιθανών απαντήσεων – προτάσεων. Αυτή η διαδικασία **παίρνει το συντακτικό δένδρο της κάθε πιθανής πρότασης** και μέσω της συνάρτησης `tree_search` επιστρέφει τα **υποδένδρα** τύπου S δηλαδή τις υποπροτάσεις της προτάσεως, πολλές φορές μια πρόταση μπορεί να έχει πολλές υποπροτάσεις. Αφού γίνει αυτό στο επόμενο βήμα **γίνεται έλεγχος στα φύλλα** της υποπροτάσεως της κάθε πιθανής πρότασης ώστε να **βρεθεί το υποσύνολο της ερωτήσεως** σε μια από αυτές (γίνεται όμοια επεξεργασία με αυτή της ερωτώσας) και **επιστέφω το ζευγάρι οντότητα και την σχετική υποπρόταση εάν δεν βρεθεί επιστρέφω None**.

8.3.3 Όλες οι υπόλοιπες Ερωτήσεις

Όποια ερώτηση δεν ανήκει στις προηγούμενες περιπτώσεις ή εάν κάποια από τις προηγούμενες συναρτήσεις δεν βρει την απάντηση τότε **καλείτε από την `QnA_System.py` η συνάρτηση `wh_questions` του `question_answering.py`** .

Αρχικά βρίσκει **όλα τα ρήματα της ερώτησης και όλους τους πιθανούς χρόνους** που μπορεί να έχουν και γίνεται μια αναζήτηση αυτών στις προτάσεις του άρθρου. Ο λόγος που επιλέγω τα ρήματα είναι διότι τα ρήματα από γραμματικής άποψης περιγράφουν μια **δράση ή κατάσταση** και είναι το βασικό στοιχείο μιας πρότασης, Υποκείμενο - Ρήμα - Αντικείμενο - όχι πάντα με αυτή την σειρά αλλά πρέπει να υπάρχουν και τα τρία είδη μαζί. Εάν βρω **μια ταύτιση την επιστρέφω αλλιώς καλώ την συνάρτηση `find_best_answer`** για να βρω την καλύτερη πιθανή απάντηση.

Αρχικά βρίσκω τα ρήματα της **ερώτησης** ψάχνοντας στο συντακτικό της δένδρο για **φύλλα με ετικέτες VB, VBD, VBG, VBN, VBP ή VBZ** και κρατάω μόνο αυτά που δεν ανήκουν στο αρχείο 'Yes No Words (18).txt' το οποίο περιέχει τα βοηθητικά ρήματα που έχουν οι ερωτήσεις. Για κάθε ρήμα της ερώτημα, **ψάχνω για όλες τους διαθέσιμους χρόνους του από την συνάρτηση `find_all_tenses`** η οποία μέσω του `data_reading.py` και της συνάρτησης `read_verbs` διαβάζει το περιεχόμενο του αρχείου '`verbs (8567).txt`' το οποίο διαθέτει όλους τους χρόνους για 8567 ρήματα είναι **ένα είδος thesaurus** (δηλαδή λίστα με λέξεις με κοινά χαρακτηριστικά). Έπειτα, **ψάχνω** στις υποπροτάσεις των προτάσεων του άρθρου **για υποπροτάσεις που περιέχουν οποιαδήποτε ρήμα της ερώτησης και τις αποθηκεύω σε μια λίστα**.

Εάν η λίστα αυτή έχει **μέγεθος ένα (1)** τότε η υποπρόταση αυτή είναι και η **απάντηση** και επιστρέφεται αλλιώς εάν η λίστα έχει **μέγεθος πάνω από μια (>1)** τότε καλώ την **find_best_answer** με παραμέτρους την ερώτηση και τις **πιθανές υποπροτάσεις** ενώ εάν η λίστα έχει **μέγεθος μηδέν (0)** τότε καλώ την **find_best_answer** με παραμέτρους την ερώτηση και **όλες τις προτάσεις του άρθρου**.

Η συνάρτηση **find_best_answer** αυτό που κάνει είναι να **υπολογίζει το cosine similarity** (ομοιότητα συνημίτονων χωρίς κανονικοποίηση) Εικόνα 15 ανάμεσα στην διάνυσμα της ερώτησης και τα διανύσματα των παραμέτρων της είτε αυτά είναι πιθανές απαντήσεις είτε όλες τις προτάσεις του άρθρου ανάλογα με την κλήση της, η **πρόταση με το μεγαλύτερο score – βαθμολογία επιστρέφεται απάντηση** σαν.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

Εικόνα 15: Ο τύπος του Cosine Similarity.

Τέλος, η συνάρτηση **wh_questions** **επιστρέφει** είτε την μια και καλύτερη **απάντηση** είτε την απάντηση της **find_best_answer** **μαζί με ένα σχόλιο** το οποίο δεν εμφανίζεται τον χρήστη αλλά γράφεται μόνο στο αρχείο με της απαντήσεις (για λόγους μελλοντικής ανάπτυξης του προγράμματος) το οποία εάν έχω μια **πιθανή απάντηση** έχει το μήνυμα **‘Found in wh_questions’**, ένα **από μία και πάνω απαντήσεις** έχει το μήνυμα **‘Found in wh_questions -> Best possible’** και εάν δεν έχει καμία δηλαδή **μηδέν** τότε έχει ο μήνυμα **‘Found in wh_questions -> Possible Wrong’**.

8.4 Έξοδος του προγράμματος

Αφού ο χρήστης έχει επιλέξει ένα κείμενο στο δεξιό μέρος της οθόνης εμφανίζονται όλες οι ερωτήσεις και απαντήσεις και το ποσοστό επιτυχίας, αυτή είναι η **έξοδος που βλέπει ο χρήστης από το παραθυρικό περιβάλλον** του βλέπε Εικόνα 10.

Το πρόγραμμα όμως έχει σχεδιαστεί έτσι ώστε να αποθηκεύει τις **απαντήσεις** και το ποσοστό επιτυχίας – ακρίβειας **στον φάκελο** Output/[το όνομα του αντίστοιχου Dataset]/[το όνομα του αντίστοιχου Άρθρου] στα αρχεία **‘system_answers.txt’** και **‘system_answers_eval.txt’** αντίστοιχα για μελλοντική εξέλιξη του προγράμματος και αξιολόγηση του συστήματος. Στις Εικόνες 16 και 17 που ακολουθούν θα δείτε την δομή των αρχείων που προανέφερα για το άρθρο 10 του Dataset S10.

ArticleTitle	Question	Answer	Answer'sComment
Zebra	do the different species of zebras interbreed?	No	Not the different species of zebras interbreed.
Zebra	do zebras sleep standing up?	Yes	Zebras sleep standing up.
Zebra	have plains zebras been crossed with mountain zebras?	Yes	Plains zebras been crossed with mountain zebras.
Zebra	how many species of zebra are there?	There are three species of zebra: the plains zebra, grévy's zebra and the mountain zebra.	Found in
Zebra	what do zebras eat?	They feed mainly on grasses but will also eat shrubs , herbs , twigs , leaves and bark .	Found in wh_questions
Zebra	what are zebras hunted for?	Zebras were , and still are , hunted mainly for their skins .	Found in wh_questions -> Best possible
Zebra	what areas do the grevy's zebras inhabit?	Unlike the other zebra species, grevy's zebras do not have permanent social bonds.	Found in wh
Zebra	which species of zebra is known as the common zebra?	It , or particular subspecies of it , have also been known as the common zebra , the	
Zebra	at what age can a zebra breed?	Attempts to breed a grévy's zebra stallion to mountain zebra mares resulted in a high rate of miscarriage.	

Εικόνα 16: Αρχείο system_answers.txt για το άρθρο Zebra του Dataset S10.

```
For this article Zebra the score (correct/all_number) is: 66.67%
```

Εικόνα 17: Αρχείο system_answers.txt για το άρθρο Zebra του Dataset S10.

9 Πως χρησιμοποιώ το Σύστημα Απάντησης Ερωτήσεων (ΣΑΕ) με δεδομένα του χρήστη

Η χρήση του συστήματος από τον χρήστη και ειδικά με δικά του δεδομένα είναι βασική και για αν επιτευχθεί πρέπει να ακολουθηθούν οι παρακάτω οδηγίες:

- Τροποποίηση του αρχείου εισόδου από τον χρήστη
- Τρέξιμο του προγράμματος με δικά μου δεδομένα
- Γιατί επιλέγω την τροποποίηση του αρχείου εισόδου από την εισαγωγή μέσω της διεπαφής

Στις παρακάτω υποενότητες θα αναλυθεί διεξοδικά η απάντηση και εξήγηση στα παραπάνω ερωτήματα.

9.1 Τροποποίηση του αρχείου εισόδου από τον χρήστη

Ο χρήστης που επιθυμεί να τρέξει το πρόγραμμα με δικά του δεδομένα δεν έχει παρά να πάει στον φάκελο **Input/Question_Answer_Dataset_v1.2** να επιλέξει οποιαδήποτε από τους τρεις φακέλους (**S08, S09 & S10**) θέλει δεν παίζει ρόλο και να ανοίξει με έναν κειμενογράφο το αρχείο **‘question_answer_pairs.txt’** βλέπε Εικόνα 14 και να **προσθέσει στο τέλος ή σε όποιο σημείο θέλει** μια σειρά για κάθε ερώτηση του με βάση την κεφαλίδα που υπάρχει ήδη **‘ArticleTitle | Question | Answer | DifficultyFromQuestioner | DifficultyFromAnswerer | ArticleFile’** δηλαδή ένα όνομα που θέλει να δώσει ο χρήστης π.χ. **MyRun1** ακολουθούμενο από την **Ερώτηση** την **Απάντησης εάν την ξέρει** (ώστε να εφαρμόσει **benchmark**) **αλλιώς NULL** έπειτα να γράψει **NULL, NULL** και τέλος την θέση του κειμένου. Όλα πρέπει να **χωρίζονται με τον χαρακτήρα ‘|’** αυτό επαναλαμβάνεται για κάθε ερώτηση και επίσης **μπορεί να προστεθούν παραπάνω από ένα αρχεία κειμένου και ArticleTitle αντίστοιχα.**

9.2 Τρέξιμο του προγράμματος με δικά μου δεδομένα

Για να εκτελέσει και να δει τα αποτελέσματα του ο χρήστης πρέπει να εκτελέσει το πρόγραμμα να επιλέξει Dataset S08, S09 ή S10 ανάλογα με το αρχείο το οποίο τροποποίησε πιο πριν και να δει τα αποτελέσματα στο δεξιό μέρος της οθόνης και στα αρχεία που παράχθηκαν

9.3 Γιατί επιλέγω την τροποποίηση του αρχείου εισόδου από την εισαγωγή μέσω της διεπαφής

Η επιλογή του να τοποθετεί ο χρήστης το δικό του αρχείο κειμένου και να το τροποποιεί μέσω ενός αρχείου και μετά απλά να τρέχει το πρόγραμμα για εμένα είναι καλύτερη από μια ‘σύνθετη’ διεπαφή η οποία θα απαιτεί από τον χρήστη να εισάγει ένα αρχείο κειμένου και ένα αρχείο ερωτήσεων ή ένα αρχείο και με τα δύο μαζί .

Πιστεύω ότι είναι προτιμότερη η προσέγγιση του να **‘ρυθμίζεις’ ένα αρχείο το οποίο είναι η είσοδος σου μια φορά (άπαξ)** που θες να εισάγει ή να διαγράψεις κάτι. Παρά να επαναλαμβάνεις συνέχεια την ίδια ρουτίνα στην διεπαφή.

*π.χ. 20 εκτελέσεις σημαίνει ότι 20 φορές που θα ανοιχτεί το πρόγραμμα θα πρέπει να εισάγεις 20 φορές το αρχείο κειμένου και 20 φορές το αρχείο ερωτήσεων και εάν θες να κάνεις και 2ο έλεγχο με άλλο αρχείο κειμένου και άλλο αρχείο ερωτήσεων άλλες 20 φορές το ίδιο πράγμα δηλαδή 20 τρεξίματα * (2 φορτώματα αρχείων κειμένου + 2 φορτώματα αρχείων ερωτήσεων) = 80 φορτώσεις αρχείων, ενώ μπορείς να το αποφύγεις απλά τροποποιώντας ένα αρχείο και να πατάς μονό το πλήκτρο Run.*

Ενδεχομένως ένας μη εξοικειωμένος χρήστης να δυσκολευτεί λίγο αλλά για αυτό έχει δημιουργηθεί αυτή η ενότητα για την διευκόλυνση του και εάν κάποιος επιθυμεί να **κάνει πολλά και διαφορετικά τρεξίματα** μπορεί να αφιερώσει λίγο παραπάνω χρόνο για την τροποποίηση του αρχείου και έπειτα να τα εκτελέσει όλα μαζί σε μία εκτέλεση.

10

Μετρικές του Συστήματος Απάντησης Ερωτήσεων (ΣΑΕ)

Βασική είναι και η αξιολόγηση και η ανάλυση της αξιολόγησης του συστήματος που παρουσιάζεται στις παρακάτω υποενότητες:

- Ταυτότητα δεδομένων - στατιστικά
- Τα αποτελέσματα των μετρικών
- Ανάλυση κλίμακας των τιμών των μετρικών

Στις παρακάτω υποενότητες θα αναλυθεί διεξοδικά η απάντηση και εξήγηση στα παραπάνω ερωτήματα.

10.1 Ταυτότητα δεδομένων - στατιστικά

Το πρόγραμμα χρησιμοποιεί τρία Datasets τα S08, S09 και S10 τα οποία απαρτίζονται από 28, 27 και 44 άρθρα αντίστοιχα αυτό σημαίνει ότι το σύστημα έχει τρέξει σε **97 άρθρα**. Τα 97 άρθρα είχαν συνολικά **19.274 προτάσεις** δηλαδή κατά μέσο όρο 198,70 προτάσεις ανά άρθρο, νούμερο ρεαλιστικό εάν συμμεριστούμε ότι έχουμε να κάνουμε με **άρθρα τα οποία ανήκουν στη Wikipedia** και είναι επιστημονικά τα περισσότερα.

Όλες οι **ερωτήσεις που έγιναν στο σύστημα ήταν 1910** δηλαδή κατά μέσο όρο 19,69 ερωτήσεις ανά άρθρο, από τις 1910 ερωτήσεις **ξέρω την σωστή απάντηση μόνο για τις 1826 που έχουν απαντηθεί από τους αξιολογητές** οι υπόλοιπες 84 είχαν NULL σαν απάντηση.

Οι τιμές αυτές έχουν δημιουργηθεί από τον αρχείο `system_evaluation.py` στον φάκελο Evaluation ο οποίος διαβάζει όλα τα αρχεία `'system_answers_eval.txt'`. Αυτά τα στοιχεία αποτελούν την ταυτότητα των δεδομένων στα οποία έτρεξε το σύστημα.

10.2 Τα αποτελέσματα των μετρικών

Οι μετρικές που υπολόγισα Εικόνα 18 για το σύστημα απάντησης ερωτήσεων είναι η ακρίβεια, η μέση τιμή , ο μέσος όρος και η μέγιστη ακρίβεια που υπάρχει. Η **ακρίβεια** υπολογίζεται από τον λόγο του κλάσματος των ερωτήσεων που έχει απαντήσει σωστά το σύστημα προς το σύνολο των ερωτήσεων για τις οποίες ξέρω την απάντηση τους (1826). Τέλος, η **μέγιστη ακρίβεια** που έπιασε το σύστημα ήταν **88.89%**.

10.3 Ανάλυση κλίμακας των τιμών των μετρικών

Η **απόλυτη ταύτιση - ακρίβεια** είναι το ποσοστό που μετράει το ποσοστό των πραγματικά σωστών ερωτήσεων. Για την αξιολόγηση του συστήματος πρέπει να φτιάξουμε μια κλίμακα **από το 0% έως την ανθρώπινη απόδοση** στην απάντηση ερωτήσεων, από δοκιμές έχει προκύψει ότι η ανθρώπινη απόδοση είναι **της τάξεως του 82,30 %**.

11 Συμπεράσματα - Μελλοντικές κατευθύνσεις

11.1 Συμπεράσματα

Λάβουμε υπόψιν ότι το σύστημα μου προκύπτει από έναν **συνδυασμό γλωσσολογίας-γραμματικής και NLP τεχνολογιών**, δύναται να απαντήσει ερωτήσεις ανοιχτού τύπου ερωτήσεις, η οποίες δεν απαιτούν την συλλογή πρώτα ερωτήσεων και απαντήσεων (και μόνο για συγκεκριμένο domain), **χωρίς να χρησιμοποιηθεί ένα σύστημα που θα χρειάζεται εκπαίδευση** και με γρήγορη απάντηση (χωρίς την προεπεξεργασία) να **φθάνει στο 43%** μπορεί να θεωρηθεί καλό λόγο την οικουμενικότητας του και σίγουρα όχι τέλειο δηλαδή 83% οπότε είναι καλό για **μια γρήγορη απάντηση**, αξίζει να αναφερθεί ότι ο έλεγχος για τις απαντήσεις ήταν αυστηρός και υπολογίστηκε μόνο με τις 100% όμοιες απαντήσεις σαν σωστές (δεν υπήρξε καθόλου ελαστικότητα).

Έπειτα από δοκιμή της ‘εξαιρετικής’ **βιβλιοθήκης Keras** (Gulli & Pal , 2017) (<https://keras.io/>) η οποία ασχολείται με υψηλού επιπέδου Νευρωνικά Δίκτυα (Neural Networks – NN) και είναι γραμμένη σε Python και της **βιβλιοθήκης TensorFlow** (Abadi, et al., 2016) (<https://www.tensorflow.org>) μια πλατφόρμα ανοιχτού κώδικα για μηχανική μάθηση και την **χρήση RNN και LSTM** συστημάτων πάνω στο **bAbI Project** παρατήρησα ότι αν έκανα εκπαίδευση με το 70% του dataset και testing στο άλλο 30% να βλέπω **ποσοστά επιτυχίας 70% με 90%** και σε **άγνωστα δεδομένα να αγγίζει το 50%**. Αυτός ήταν και ο λόγος που δεν επέλεξα την κατεύθυνση των Νευρωνικών Δικτύων όπου θα κατέληγα σε ένα μεγάλο ποσοστό αλλά επι τις ουσίας πλασματικό – μη ρεαλιστικό.

11.2 Μελλοντικές κατευθύνσεις

Η αύξηση της διαθέσιμης μνήμης **RAM** που χρειάζεται το σύστημα, λόγο της έλλειψης της διαθέσιμης μνήμης RAM απέκλεισα κάποια άρθρα τα οποία θα ήθελα να συμπεριλάβω αλλά δεν μπορούσα διότι είχαν μεγάλες προτάσεις και το CoreNLP ήθελε πάνω από τα **6 GB** που είχα διαθέσιμα (βλέπε διακόπτη –mx6g Εικόνα 6) για να μην καταρρεύσει

(το CoreNLP) ακόμα λόγω αυτής της έλλειψης δεν υπήρχε η υποστήριξη των Dependency Parse tree στο σύστημα μου.

Η χρήση **Dependency Parse Tree & Coref** τα οποίο υποστηρίζονται από το CoreNLP, το οποίο **βρίσκει τις εξαρτήσεις** που υπάρχουν ανάμεσα στις προτάσεις του κειμένου μέσω των ρημάτων και θα διεύρυνε – **μεγάλωνε την ακρίβεια του συστήματος ειδικά στην κατηγορία όλες οι υπόλοιπες ερωτήσεις** (από την οποία έχω χάσει το μεγαλύτερο ποσοστό της ακρίβειας μου). Θα διόρθωνε το τωρινό σύστημα το οποίο βρίσκει την απάντηση της ερώτησης εάν η πρόταση που περιέχει την απάντηση περιέχει το ρήμα της ερώτησης πράγμα που δεν ισχύει πάντα.

Τέλος, η αγορά ενός domain και τι στήσιμο ενός server ο οποίος **θα έτρεχε το πρόγραμμα online** σε συνδυασμό με ένα interface φιλικό προς τον χρήστη και η **υποστήριξη της Ελληνικής γλώσσας** (όπως η speech engine MAIC της MLS) και γενικά η συλλογή feedback.

Βιβλιογραφία

Abadi, M. και συν., 2016. TensorFlow: A System for Large-Scale Machine Learning. Στο: *12th USENIX Symposium on Operating Systems Design and Implementation*. s.l.:USENIX, pp. 265-283.

Anon., 2018 . *Learn English British Council*. [Ηλεκτρονικό]
Available at: <https://learnenglish.britishcouncil.org/english-grammar-reference/questions-and-negatives>
[Πρόσβαση 15 January 2019].

Anon., 2018. *Your Dictionary*. [Ηλεκτρονικό]
Available at: <https://examples.yourdictionary.com/examples-of-open-ended-and-closed-ended-questions.html>
[Πρόσβαση 15 January 2019].

Barbieri, D. F. και συν., 2009. *C-SPARQL : SPARQL for continuous querying*. s.l., WWW.

Brill, E., Dumais, S. & Banko, M., 2002. An Analysis of the AskMSR Question-answering System. Στο: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, Pennsylvania: Association for Computational Linguistics, pp. 257-264.

Elworthy, D., 2001. Question Answering using a large NLP System. *TREC*, March .

Foard, C. F. & Kemler, D. G., 1984. Holistic and analytic modes of processing: The multiple determinants of perceptual analysis. *Journal of Experimental Psychology: General*, 113(1), pp. 94-111.

Green , C., 1969. Theorem-proving by resolution as a basis for question-answering systems, in. *Machine Intelligence*, B. Meltzer and D. Michie, eds, pp. 183-205.

Gulli , A. & Pal , S., 2017. *Deep Learning with Keras : Implementing deep learning models and neural networks with the power of Python*. 1st επιμ. Birmingham: Packt.

Haykin, S., 1994 . *Neural Networks: A Comprehensive Foundation*. 1st επιμ. New Jersey: Prentice Hall PTR.

Kalyanpur, A. και συν., 2012. Structured data and inference in DeepQA. *IBM Journal of Research and Development*, Τόμος 56, pp. 10:1-10:14.

LeCun, Y., Bengio, Y. & Hinton, G., 2015. Deep learning. *Nature*, Τόμος 521, pp. 436-444.

Loper, E. & Bird, S., 2002. NLTK: The Natural Language Toolkit. *CoRR*.

Manning, C. D. και συν., 2014. *The Stanford CoreNLP Natural Language Processing Toolkit*. Baltimore, Maryland, Association for Computational Linguistics.

Simmons, R. F., 1970. Natural language question-answering systems: 1969. Στο: *Communications of the ACM*. Volume 13 Issue 1 επιμ. New York: ACM, pp. 15-30.

Smith, N. A., Heilman, M. & Hwa, R., 2008. *Question Generation as a Competitive Undergraduate Course Project*. Arlington, In Proceedings of the NSF Workshop on the Question Generation Shared Task and Evaluation Challenge.

Voorhees, E. M., 2001. *Overview of the TREC-9 Question Answering Track*. s.l., s.n., pp. 71-80.

Παραρτήματα

Κατάλογος διαγραμμάτων

Εικόνα 1: Η αρχιτεκτονική του συστήματος DeepQA.....	22
Εικόνα 2: Οι διαφορές ανάμεσα στην Python 2 και Python 3.	26
Εικόνα 3: Παραδείγματα επισήμανσης του Stanford CoreNLP.....	30
Εικόνα 4: Εκκίνηση του Server μέσω του τερματικού.	31
Εικόνα 5: Στην σελίδα localhost:9000 του Browser μας τρέχει ο CoreNLP Server.....	32
Εικόνα 6: Βλέπουμε ότι ο Server εκτελεί Part of Speech για την πρόταση της προηγούμενης Εικόνας.	32
Εικόνα 7: Ένα απλό παράθυρο σχεδιασμένο με tKinter.	34
Εικόνα 8: Μήνυμα ειδοποίησης για την σωστή λειτουργία του προγράμματος.	39
Εικόνα 9: Η εκτέλεση του προγράμματος για το άρθρο Zebra (α).	40
Εικόνα 10: Η εκτέλεση του προγράμματος για το άρθρο Zebra (β).	41
Εικόνα 11: Το αρχείο a9.htm που αντιστοιχεί στο άρθρο Zebra.	43
Εικόνα 12: Το αρχείο a9.txt που αντιστοιχεί στο άρθρο Zebra.	43
Εικόνα 13: Η κεφαλίδα του αρχείου question_answer_pairs.txt	44
Εικόνα 14: Το περιεχόμενο του αρχείου question_answer_pairs.txt	44
Εικόνα 15: Ο τύπος του Cosine Similarity.....	50
Εικόνα 16: Αρχείο system_answers.txt για το άρθρο Zebra του Dataset S10.	51
Εικόνα 17: Αρχείο system_answers.txt για το άρθρο Zebra του Dataset S10.	51

Απόδοση όρων

Artificial Intelligence	Τεχνητή Νοημοσύνη
Cognitive computing	Σύστημα AI που προσομοιώνει την ανθρώπινη σκέψη
Computer vision	Σύστημα AI που προσομοιώνει την υπολογιστική όραση
Command line	Γραμμή εντολών
Deep Learning	Μέθοδος μάθησης βασισμένος σε τεχνητά νευρωνικά δίκτυα
Machine Learning	Μηχανική μάθηση
Natural Language Processing	Επεξεργασία φυσικής γλώσσας
Named Entity Recognition	Αναγνωρισμένη ονομασία οντοτήτων
Neural Network	Νευρικό δίκτυο
Speech engine	Μηχανή ομιλίας
Tag	Ετικέτα (π.χ. NER ή POS)
Wikipedia	Δωρεάν ηλεκτρονική εγκυκλοπαίδεια

Συντομογραφίες

ΣΑΕ	Σύστημα Απάντησης Ερωτήσεων
AI	Artificial Intelligence
BAbI	Project of Facebook AI Research
DARPA	Defense Advanced Research Projects Agency
IDE	Integrated Development Environment
IoT	Internet of Things
IR	Information Retrieval
LSTM	Long Sort Term Memory
NER	Named Entity Recognition
NLP	Natural Language Processing
NLTK	Natural Language ToolKit
NN	Neural Network
POS	Part Of Speech
QA	Question Answering
RAM	Random Access Memory
RNN	Recurrent Neural Network
SQL	Structured Query Language