

Ανάπτυξη Εφαρμογής για ταξιδιωτικούς προορισμούς

Ιωάννης-Παναγιώτης Κερασιώτης

26 Σεπτεμβρίου 2022



ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ



Αυτή η πτυχιακή έχει άδεια χρήσης Creative Commons CC0 1.0 Universal
(CC0 1.0) Public Domain Dedication.

Τριμελής Επιτροπή

Επιβλέπων, Δημήτρης Ρίγγας, Μέλος ΕΔΙΠ

Συνεπιβλέπων, Ελένη Χριστοπούλου, Μέλος ΕΔΙΠ

Συνεπιβλέπων, Μανώλης Μαραγκουδάκης, Καθηγητής

Η παρούσα πτυχιακή εργασία εκπονήθηκε στα πλαίσια του προπτυχιακού προγράμματος του Τμήματος Πληροφορικής του Ιονίου Πανεπιστημίου.

Στους γονείς μου

Περιεχόμενα

1	Εισαγωγή	7
2	Ανασκόπηση βιβλιογραφίας	9
2.1	Εισαγωγή στη Μηχανική Μάθηση	10
2.1.1	Supervised, Unsupervised Learning	12
2.1.2	Μηχανική μάθηση στα Νευρωνικά Δίκτυα	13
2.2	Εισαγωγή στο DeepLearning	14
2.3	Εισαγωγή στα Νευρωνικά Δίκτυα	16
2.3.1	Feed-Forward Νευρωνικά Δίκτυα	17
2.3.2	Reccurent Neural Networks	19
2.4	Sentiment Analysis	19
2.4.1	Ανασκόπηση έρευνας στο Sentiment Analysis	23
2.4.2	Μέθοδοι Sentiment Analysis	26
2.5	Transfer Learning	30
2.5.1	Τεχνικές Transfer Learning	32
2.6	Σύγκριση με παρόμοια προγράμματα	34
2.6.1	Άλλες ανάλογες εφαρμογές (Tripadvisor, Foursquare)	34
2.6.2	Σύγκριση άλλων μεθόδων Sentiment Analysis	37
2.6.3	Γιατι ένας χρήστης να χρειάζεται Sentiment Analysis	38
3	Υλοποίηση - Τεχνική επισκόπηση	39
3.1	Mobile Apps & Rest APIs	40
3.1.1	Κατηγορίες Mobile Εφαρμογών	41
3.1.2	Mobile Application Testing	45
3.2	Τεχνολογίες που χρησιμοποιήθηκαν	46
3.2.1	Back-end & Front-end Εφαρμογής	49
3.2.2	APIs που χρησιμοποιήθηκαν	50
4	SUS Evaluation	57
4.1	Σχολιασμός αποτελεσμάτων του SUS evaluation	58
4.2	Αξιολόγηση κριτικών από χρήστες	60
5	Συμπεράσματα - μελλοντικές επεκτάσεις	62
5.1	Αλλαγες στο Mobile App	62
5.2	Αλλαγές στο Sentiment Analysis	63
5.3	Προβλήματα που παρουσιάστηκαν κατά την υλοποίηση	63
6	Κομμάτια Κώδικα	65

6.1	Client-side	65
6.1.1	screens.js	65
6.1.2	googleReviews.js	66
6.1.3	resultsCities.js	69
6.2	webcamsDisplay.js	71
6.3	Server-side)	72
6.3.1	server.js	72
6.3.2	stores.js	74
6.3.3	worldcams.js	77
6.3.4	SentAnalysis.py	78

Κατάλογος Σχημάτων

1	Εικονική αναπαράσταση κλάδων της Τεχνητής Νοημοσύνης . . .	9
2	Γενικευμένη Διαδικασία Μηχανικής Μάθησης	12
3	Διαδικασία Μηχανικής Μάθησης	13
4	Διαγράμματα Μάθησης Με/Χωρίς Επίβλεψη	14
5	Ιεραρχία επιπέδων Deep Learning	15
6	Βιολογικός Νευρώνας	16
7	Αρχιτεκτονική Feed Forward Νευρωνικού Δικτύου	17
8	Γραμμική / Μη γραμμική απεικόνιση προβλήματος	18
9	Reccurent Neural Network Visualisation	20
10	Sentiment Analysis model	21
11	Sensitivity and specificity diagram	22
12	Sentiment Analysis Diagram	25
13	Traditional vs Transfer Learning	31
14	Inductive Learning, Inductive Transfer	33
15	TripAdvisor Screenshots	35
16	FourSquare Screenshots	37
17	Αρχιτεκτονική ολόκληρης της εφαρμογής	40
18	Αρχιτεκτονική του Cordova framework	43
19	Έρευνα σχετικά με τον αριθμό smartphone το 2017	44
20	App Screenshots	56
21	Αποτελέσματα ερωτήσεων του SUS Evaluation	58
22	Αποτελέσματα ερωτήσεων του SUS Evaluation (2)	59

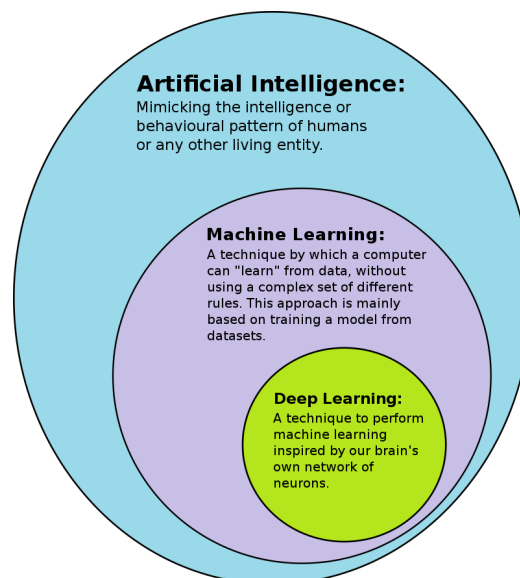
1 Εισαγωγή

Στη συγκεκριμένη πτυχιακή σκοπός είναι η δημιουργία μίας εφαρμογής για ταξιδιωτικούς προορισμούς, έχοντας ως στόχο την εύρεση καταστημάτων με τις χειρότερες κριτικές. Συγκεκριμένα ο χρήστης δίνει σαν παράμετρο έναν προορισμό, και του επιτρέφονται πληροφορίες όπως ο καιρός, οι τιμές που επικρατούν στην περιοχή σε σύγκριση με τις τιμές στη Θεσσαλονίκη, ζωντανό υλικό από κάμερες εφόσον υπάρχουν. Και τέλος και πιο σημαντικό, μαγαζιά της περιοχής, ταξινομημένα από το χειρότερο, με βάση κριτικές χρηστών, προς το καλύτερο. Μάλιστα συμπεριλαμβάνονται και τα σχόλια ένα-ένα μαζί με το βαθμό **sentiment analysis** στο κάθε σχόλιο. Ο καιρός και η αυτόματη συμπλήρωση στην αναζήτηση προορισμού, προέρχεται από το **WeatherApi**, η κατάσταση των τιμών από το **Numbeo**, και τα σχόλια και τα καταστήματα απ το **Google Places API**. Ο ρόλος της εφαρμογής είναι η εύρεση των καταστημάτων με τις χειρότερες κριτικές ώστε να ερευνηθούν οι λόγοι που υποβαθμίζουν την ποιότητα του εκάστοτε καταστήματος. Συμπληρωματικά μπορεί να χρησιμοποιηθεί και σαν οδηγός ώστε ο χρήστης να μπορέσει να αποφύγει αυτά τα καταστήματα. Τα σημαντικότερα εργαλεία που χρησιμοποιήθηκαν είναι η **JavaScript** καθώς το πρόγραμμα είναι εξ' ολοκλήρου γραμμένο σε αυτή τη γλώσσα, **front-end** και **back-end**. Και το **expo** το οποίο είναι το πρόγραμμα που δημιουργεί συμπληρωματικά αρχεία ώστε να μπορεί η εφαρμογή να κάνει **compile** σε **Android** και **iOS**. Αλλά και προσβάσιμη στην εκάστοτε συσκευή για κατ' ιδίαν δοκιμή. Η ανάλυση συναισθήματος παίζει κομβικό ρόλο καθώς είναι η πιο ουσιαστική προτοτυπία που υπάρχει, ο κώδικας για το **sentiment analysis** είναι γραμμένος σε **Python**, με τη μέθοδο **Transfer Learning**, εξάγει συμπέρασμα δίνοντας μία διαδυκνή τιμή (θετικό ή αρνητικό) για ένα σχόλιο. Στο τελικό στάδιο της εφαρμογής, βλέπουμε την κριτική που άφησε ο χρήστης μαζί με την βαθμολογία καθώς και τη βαθμολογία που παρήγαγε το **sentiment analysis**. Τέλος όλες αυτές οι υπηρεσίες αντλούνται από τα αντίστοιχα **API**, με εξαίρεση 1 περίπτωση, στο **API** με τις **Webcams**, όπου το χρειάστηκε να γίνει ένα προσωρινό **custom API**, λόγο του ότι στην προκείμενη στιγμή δεν είχε βρεθεί κάποιο εύκαιρο. Τα αποτελέσματα της εφαρμογής ήταν θετικά, καθώς ο αλγόριθμος αναγνωρίζει το συναίσθημα με μεγάλη ακρίβεια. Επίσης άνθρωποι έχουν δοκιμάσει ζωντανά την εφαρμογή και συμπλήρωσαν ένα ερωτηματολόγιο **SUS Evaluation**, και εκεί φαίνεται και η γνώμη αληθινών χρηστών για την εφαρμογή, αυτό το ερωτηματολόγιο εμπεριέχεται στην πτυχιακή.

2 Ανασκόπηση βιβλιογραφίας

Στη συγκεκριμένη πτυχιακή εφαρμόζεται μία μέθοδος μηχανικής μάθησης με χρήση νευρωνικού δικτύου, αυτή είναι γνωστή στην ακαδημαϊκή κοινότητα ως **sentiment analysis** που σημαίνει ανάλυση συναισθήματος. Συγκεκριμένα λαμβάνοντας υπόψην λέξεις του αγγλικού λεξιλογίου, αξιολογούμε κάποιες από αυτές ως είτε αρνητικές, θετικές ή ουδέτερες. Πριν την βαθύτερη επεξήγηση των τεχνολογιών που χρησιμοποιήθηκαν, γίνεται μία αναδρομή στο τι είναι μηχανική μάθηση, νευρωνικό δίκτυο, βαθιά μάθηση, και ποιες οι χρήσεις τους στη πτυχιακή.

Σημαντικό είναι να ομαδοποιήσουμε τη Τεχνητή Νοημοσύνη, τη Μηχανική Μάθηση και τη Βαθιά Μάθηση καθώς το ένα είναι υποσύνολο του άλλου. Πιο συγκεκριμένα όπως φαίνεται και στη παρακάτω εικόνα, όλα ανήκουν στην ομπρέλλα της τεχνητής νοημοσύνης η οποία μοντελοποιεί την ανθρώπινη συμπεριφορά. Ένα υποσύνολο της αποτελεί η μηχανική μάθηση, η οποία χρησιμοποιείται από αλγόριθμους που κάνουν τους υπολογιστές να 'μαθαίνουν' από δεδομένα. Και τέλος η βαθιά μάθηση που με τη πρωσομοίωση ανθρώπινων νευρώνων λεγόμενων ως νευρωνικών δικτύων, μπορούμε να ανάγουμε αυτή τη λειτουργία σε ένα τεχνητό σύστημα.



Σχήμα 1: Εικονική αναπαράσταση κλάδων της Τεχνητής Νοημοσύνης [1]

2.1 Εισαγωγή στη Μηχανική Μάθηση

Επίσημος ορισμός της μηχανικής μάθησης Υπάρχουν πολλοί επίσημοι ορισμοί της μηχανικής μάθησης, ένας εκ των οποίων ορίστηκε από τον **Tom Mitchell** ως εξής (ακολουθείται ελεύθερη μετάφραση) : ' Ένα πρόγραμμα λέγεται ότι μαθαίνει από μία εμπειρία E μέσω μίας διεργασίας T και μέτρο επίδοσης P , αν η επίδοση της διεργασίας T , μετρημένη από την P , βελτιώνεται με εμπειρία E . [2]

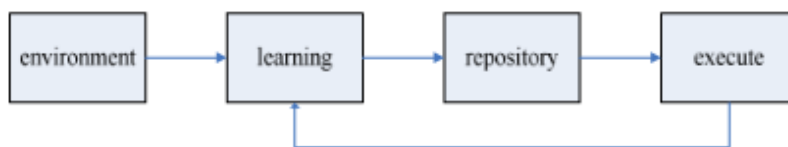
Η μηχανική μάθηση είναι ένας κλάδος που μελετά τον τρόπο χρήσης στην προσομοίωση των ανθρώπινων μαθησιακών δραστηριοτήτων και τη μελέτη μεθόδων αυτοβελτίωσης. Σε σύγκριση με την ανθρώπινη μάθηση, η μηχανική μάθηση μαθαίνει πιο γρήγορα, η συσσώρευση γνώσης διευκολύνει αποτελέσματα της μάθησης να διαδίδονται ευκολότερα. Με την πρόοδο της μηχανικής μάθησης και τον υπολογιστών εποφελείται και η ανθρώπινη κοινωνία. Υπάρχουν ποικίλλες χρήσεις της μηχανικής μάθησης σε διάφορους επιστημονικούς τομείς, όπως η τεχνητή νοημοσύνη, στην επιστήμη των υπολογιστών, τις πιθανότητες και τη ψυχολογία. Το μεγαλύτερο αναγνωριστικό αυτών των αλγορίθμων, είναι ο τρόπος να αντιλαμβάνονται τις μεταβλητές του εικονικού αυτού περιβάλλοντος, δίνοντας τους δεδομένα.

Πρώτα λαμβάνονται οι πληροφορίες του εξωτερικού περιβάλλοντος και στη συνέχεια μετατρέπονται σε γνώση και τοποθετούνται σε ένα αποθετήριο. Στο αποθετήριο αποθηκεύονται πολλές γενικές αρχές που καθοδηγούν ένα μέρος της υλοποίησης. Η ποιότητα των πληροφοριών επηρεάζει άμεσα την πραγματοποίηση μάθησης είτε τα δεδομένα είναι ταξινομημένα είτε όχι. Το αποθετήριο αποτελεί ένα ακόμη παράγοντα που επηρεάζει την αρχιτεκτονική του συστήματος. Η αποτύπωση των γνώσεων ποικίλλει, και μπορεί να έχει τη μορφή ιδιόσυστατος, λογικής πύλης (0 ή 1), σημασιολογικών δικτύων κ.α.. Κάθε τύπος είναι ιδιαίτερος και η καταλληλότητα του καθενός εξετάζεται ανά περίπτωση. Κατά τη διάρκεια του στάδιου της υλοποίησης η γνώση χρησιμοποιείται, όχι μόνο για την διεκπεραίωση της διαδικασίας, αλλά και για την ανατροφοδότηση, που βοηθά στην επιτάχυνση μεταγενέστερων διαδικασιών, επιτυγχάνοντας τη μάθηση από το σύστημα. Τέλος η τεχνολογία μηχανικής μάθησης έχει χρησιμοποιηθεί ευρέως στο **marketing**, τις τηλεπικοινωνίες, την ανάλυση δικτύων, αλλά και ευρύτερα σε διεργασίες προβλέψεων. Επιπλέον εφαρμόζεται στον τομέα της εξόρυξης δεδομένων σε συνδυασμό με άλλες εφαρμογές. Οι 2 κύριοι τύποι αλγορίθμων μηχανικής μάθησης είναι οι αλγόριθμοι με επίβλεψη, και αλγόριθμοι χωρίς επίβλεψη, στους αλγόριθμους με επίβλεψη τροφοδοτούμε το σύστημα με δεδομένα ώστε να παράξει κάποιο αποτέλεσμα, και στους αλ-

γόριθμους χωρίς επίβλεψη, προγραμματίζουμε τον αλγόριθμο να θέσει ο ίδιος τα χαρακτηριστικά και τις μετρήσεις. Αν και οι 2 διαδικασίες είναι εφικτές από τον ανθρώπινο νου, υπάρχουν διαδικασίες χρονοβόρες αλλά και διαδικασίες που να χρειάζονται ακρίβεια και λεπτομέρεια που είναι δύσκολο να επιτευχθούν από έναν άνθρωπο. Ένα παράδειγμα αποτελεί η επίβλεψη όγκων και οργάνων. Για παράδειγμα, πολλές φορές οι γιατροί εξάγουν συμπεράσματα βλέποντας και μόνο την πηγή του πρόβληματος, όπως ενός όγκου, ενός οργάνου ή σε δερματολογικά προβλήματα, το ίδιο το δέρμα. Επομένως ένας γιατρός μπορεί να αναπαρασταθεί ως έναν αλγόριθμο μηχανικής μάθησης, χωρίς επίβλεψη. Ενώ ένας γιατρός που έχει εκπαιδευμένη αντίληψη μπορεί να εντοπίσει πιο αποτελεσματικά ένα πρόβλημα, λόγο πείρας, ένας νέος γιατρός δεν κατέχει αυτό το πλεονέκτημα. Ένας αλγόριθμος όμως, μπορεί με την κατάλληλη εκπαίδευση να προσομοιώσει την αντίληψη ενός έμπειρου γιατρού, βοηθώντας τη δουλειά του εκάστωτε γιατρού να βρίσκει αίτια του προβλήματος με μικρότερο χρονικό διάστημα και με μεγαλύτερη ακρίβεια. Οι τύποι αυτών των αλγορίθμων αναλύονται παρακάτω μαζί με αντίστοιχα παραδείγματα. [3, 4]

Ιδιοδιανύσματα / Eigenvectors Ο προσδιορισμός των ιδιοδιανυσμάτων και των ιδιοτιμών ενός συστήματος είναι εξαιρετικά σημαντικός στη φυσική και τη μηχανική, όπου είναι ισοδύναμος με τη διαγωνίωση πίνακα και προκύπτει σε εφαρμογές όπως η ανάλυση ευστάθειας, στη φυσική περιστρεφόμενων σωμάτων και οι μικρές ταλαντώσεις των δονούμενων συστημάτων. Κάθε ιδιοδιάνυσμα αντιστοιχίζεται με μια ιδιοτιμή. Τα ιδιοδιανύσματα χωρίζονται σε δύο διαφορετικά είδη πρέπει να διακρίνονται: τα αριστερά ιδιοδιανύσματα και τα δεξιά ιδιοδιανύσματα. Ωστόσο, για πολλά προβλήματα στη φυσική και τη μηχανική, αρκεί να ληφθούν υπόψη μόνο σωστά ιδιοδιανύσματα. Ο όρος 'ιδιοδιάνυσμα' που χρησιμοποιείται χωρίς επιφύλαξη σε τέτοιες εφαρμογές μπορεί επομένως να κατανοηθεί ότι αναφέρεται σε ένα σωστό ιδιοδιάνυσμα.[5]

Σημασιολογικό Δίκτυο / Semantic Web Ένα σημασιολογικό δίκτυο ή δίκτυο ονομάζουμε μια δομή γραφήματος για την αναπαράσταση της γνώσης σε μοτίβα διασυνδεδεμένων κόμβων. Οι εφαρμογές σημασιολογικών δικτύων σε ηλεκτρονικούς υπολογιστές αναπτύχθηκαν πρώτη φορά για εφαρμογές τεχνητής νοημοσύνης και μηχανικής μετάφρασης, προηγούμενες εκδόσεις έχουν χρησιμοποιηθεί από καιρό στη φιλοσοφία, την ψυχολογία και τη γλωσσολογία.[6]



Σχήμα 2: Γενικευμένη Διαδικασία Μηχανικής Μάθησης

[3]

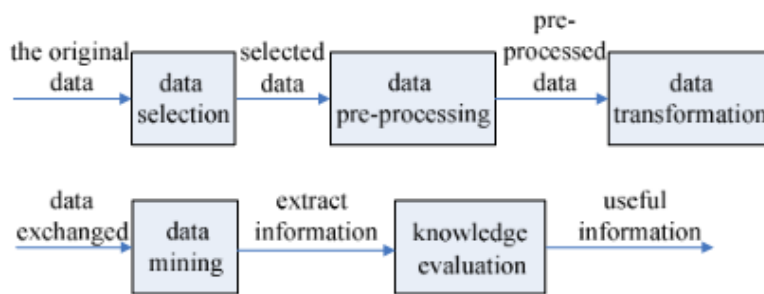
2.1.1 Supervised, Unsupervised Learning

Κάθε τύπος μηχανικής μάθησης είναι διαφορετικός και προορίζεται και διαφορετικά προβλήματα, συνεπώς δεν μπορούν να ταξινομηθούν διότι αποτελούν ξεχωριστές κατηγορίες.

Μάθηση με επίβλεψη (Supervised Learning) Στη μάθηση με επίβλεψη, η ανθρώπινη αντίληψη προσομοιώνεται σε έναν υπολογιστή. Πιο συγκεκριμένα, αναπαριστούμε κάθε αντικείμενο με τα αντίστοιχα χαρακτηριστικά του. Για παράδειγμα το αντικείμενο αυτοκίνητο αντιστοιχίζεται με τα επακόλουθα χαρακτηριστικά : 4 ρόδες, τουλάχιστον 2 πόρτες και 2 μπροστινούς προβολείς (φώτα). Έχοντας δώσει αυτά τα στοιχεία στον υπολογιστή, και έχοντάς τον τροφοδοτήσει με κατάλληλες φωτογραφίες από αυτοκίνητα, μπορούμε να δημιουργήσουμε ένα σύστημα που μπορεί να διακρίνει αυτοκίνητα που δεν έχει “ξαναδεί”. [3]

Μάθηση χωρίς επίβλεψη (Unsupervised Learning) Στη μάθηση χωρίς επίβλεψη, αντί να τροφοδοτούμε το σύστημα με δικές μας μεταβλητές, οι μεταβλητές δημιουργούνται απ’ το σύστημα καθεαυτό. Για παράδειγμα, θέλοντας να προγραμματίσουμε ένα σύστημα που θα πρέπει να στοχεύσει για να ρίξει μία μπάλα σε ένα καλάθι. Σε κάθε προσπάθεια να πετύχει το στόχο οι κινηματικοί βαθμοί του αλγορίθμου θα διαμορφώνονται έτσι ώστε κάθε προσπάθεια να γίνεται όλο και περισσότερο αποτελεσματική. Σε μία ιδανική περίπτωση το σύστημα θα μπορεί να πετύχει το στόχο ασχέτος που βρίσκεται, έχοντας βέβαια προπονηθεί σε πολλές, ποικίλλες των περιπτώσεων.[3]

semi-supervised learning Σε αυτή τη περίπτωση, δίνονται στον αλγόριθμο κάποια από τα χαρακτηριστικά του προβλήματος, ενώ κάποια θα πρέπει να τα δημιουργήσει από μόνο του. Ένα παράδειγμα θα μπορούσε να αποτελέσει, τα χαρακτηριστικά που δίνουμε εμείς να χρησιμοποιηθούν για βοήθεια στην εύρεση των χαρακτηριστικών που θα πρέπει να βρει μόνο του το σύστημα. Αυτός



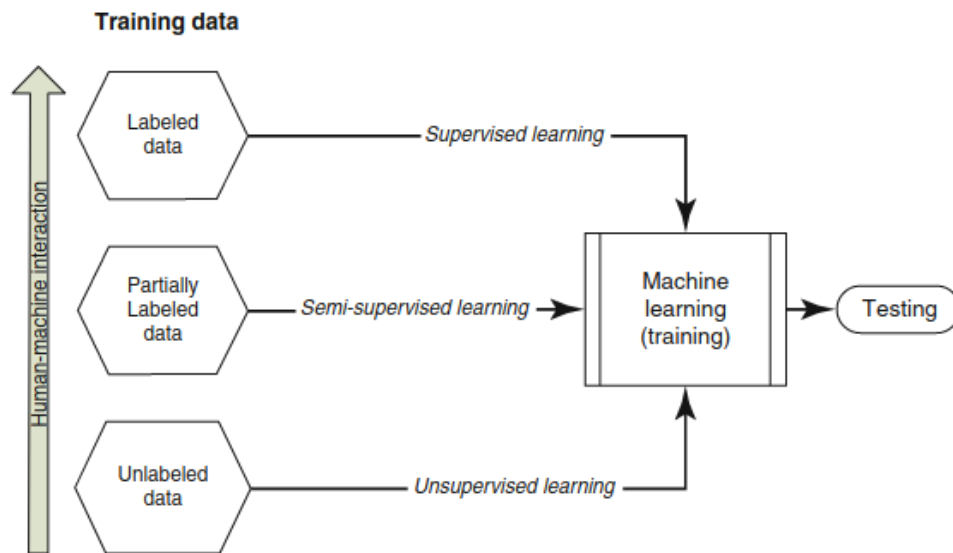
Σχήμα 3: Διαδικασία Μηχανικής Μάθησης
[4]

ο τύπος αλγορίθμων είναι συγκλίνει περισσότερο στον τρόπο με τον οποίο οι άνθρωποι μαθαίνουμε και αναπτυσσόμαστε.[3]

2.1.2 Μηχανική μάθηση στα Νευρωνικά Δίκτυα

Η φύση του νευρωνικού δικτύου εξαρτάται από δύο βασικούς παράγοντες: τη δομή της τοπολογίας του δικτύου, τις σωστές τιμές και τους κανόνες εργασίας του δικτύου. Ο συνδυασμός των δύο μπορεί να σχηματίσει τους κύριους χαρακτήρες ενός δικτύου. Το μαθησιακό πρόβλημα του νευρωνικού δικτύου είναι το πρόβλημα της προσαρμογής των τιμών του δικτύου. Ο πιο συνηθισμένος τρόπος υπολογισμού της τιμής του νευρωνικού είναι ο **back-propagation**. Η ανακάλυψη γνώσης του αποθετηρίου είναι μια ανώτερη διαχειριζόμενη διαδικασία για τον εντοπισμό αποτελεσματικού, χρήσιμου και κατανοητού μοντέλου από μεγάλες ποσότητες δεδομένων. Η διαδικασία ανακάλυψης της γνώσης φαίνεται στο παρακάτω σχήμα.[4]

Back-propagation Το **back-propagation** είναι ένας αλγόριθμος που χρησιμοποιείται κατα κόρον στο τομέα της μηχανικής μάθησης, για τη βελτίωση ακρίβειας των τελικών τιμών, ή πιο συγκεκριμένα για τον υπολογισμό παραγώγων. Πιο αναλυτικά συγκρίνουμε τις τελικές τιμές του αλγορίθμου με τιμές που θα θέλαμε εμείς να έχουμε. Στη συνέχεια οι τιμές στα βάρη χρησιμοποιούνται για να συγκλίνει η τιμή του δικτύου όσο τον δυνατόν περισσότερο στη προβλεπόμενη.[7]



Σχήμα 4: Διαγράμματα Μάθησης Με/Χωρίς Επίβλεψη [3]

2.2 Εισαγωγή στο **DeepLearning**

Η τεχνολογία μηχανικής μάθησης είναι ολοένα και πιο πανταχού παρούσα σε καταναλωτικά είδη, όπως στις κάμερες και τα **smartphone**, και τροφοδοτεί πολλά μέρη της σύγχρονης ζωής, από αναζητήσεις στον ιστό έως φιλτράρισμα περιεχομένου στα κοινωνικά δίκτυα, προτάσεις σε ιστότοπους ηλεκτρονικού εμπορίου. Τα συστήματα μηχανικής μάθησης χρησιμοποιούνται για την αναγνώριση αντικειμένων στις φωτογραφίες, τη μετατροπή ομιλίας σε κείμενο, την αντιστοίχιση ειδήσεων, αναρτήσεων ή προϊόντων με τα ενδιαφέροντα των χρηστών και την επιστροφή σχετικών σελίδων κατά την αναζήτηση. Μέχρι πρότινος, ο σχεδιασμός ενός εργαλείου εξαγωγής χαρακτηριστικών που μετέτρεπε τα ακατέργαστα δεδομένα (όπως τιμές **pixel**) σε σύστημα αναγνώρισης προτύπων ή μηχανικής μάθησης, χρειαζόταν σχολαστική μηχανική και εκτενή εμπειρία στο θέμα.

Το **Deeplearning** έχει επίσης εισχωρήσει σε άλλα παρακλάδια της πληροφορικής το **comptuer vision**, της αναγνώρισης ομιλίας και πιο πρόσφατα, της επεξεργασίας φυσικής γλώσσας. Πολλές αιτίες έχουν συμβάλει στην αναβίωση των νευρωνικών δικτύων. Η διαθεσιμότητα ισχύος του υπολογιστή λόγω βελτιώσεων στο υλικό (π.χ. **GPU**), η διαθεσιμότητα τεράστιων ποσοτήτων δεδομένων εκπαίδευσης και η ισχύς και ευελιξία της εκμάθησης ενδιάμεσων



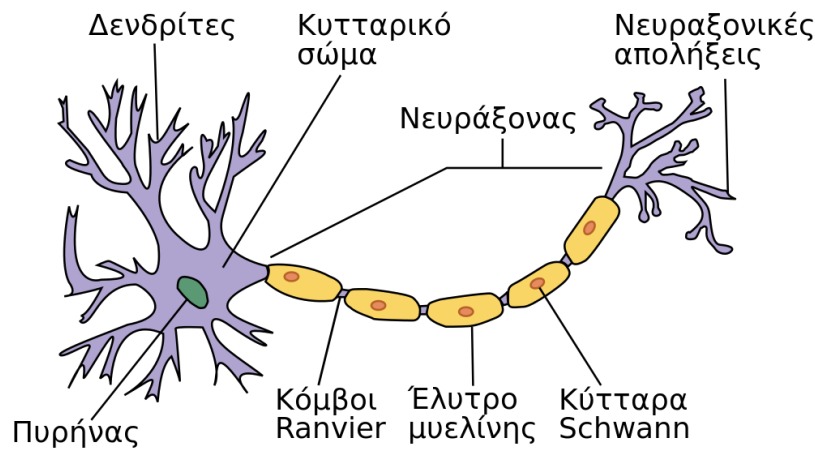
Σχήμα 5: Ιεραρχία επιπέδων Deep Learning
[8]

αναπαραστάσεων είναι από τα πιο απαραίτητα. Με μια λέξη, η βαθιά μάθηση εξάγει και μετασχηματίζει χαρακτηριστικά χρησιμοποιώντας έναν καταρράκτη πολυάριθμων επιπέδων μη γραμμικών μονάδων επεξεργασίας. Τα χαμηλότερα επίπεδα μαθαίνουν απλά χαρακτηριστικά κοντά στην είσοδο δεδομένων, αλλά τα υψηλότερα επίπεδα μαθαίνουν πιο εξελιγμένα χαρακτηριστικά που προέρχονται από χαρακτηριστικά χαμηλότερου επιπέδου.

Το παρακάτω σχήμα απεικονίζει την ιεραρχία χαρακτηριστικών που μαθαίνεται από τη βαθιά μάθηση στην ταξινόμηση εικόνας προσώπου από τα αριστερά (κάτω στρώμα) προς τα δεξιά (υψηλότερο στρώμα). Από σταγόνες/άκρες μέχρι τις μύτες/μάτια/μάγουλα μέχρι τα πρόσωπα, μπορούμε να δούμε πώς τα χαρακτηριστικά της εικόνας επεκτείνονται σε πολυπλοκότητα. Τα μοντέλα Deep Learning που έχουν χρησιμοποιηθεί ευρέως στον τομέα του NLP τα τελευταία χρόνια και έχουν δείξει σημαντικές υποσχέσεις. Οι κύριες αρχιτεκτονικές βαθιάς μάθησης και οι σχετικές προσεγγίσεις που έχουν χρησιμοποιηθεί σε προβλήματα NLP περιγράφονται συνοπτικά στις ακόλουθες ενότητες.[8]

Word Embedding Το word embedding είναι μία τεχνική στο NLP που χρησιμοποιείται στη μοντελοποίηση γλώσσας και στη μάθηση χαρακτηριστικών. Η τεχνική αυτή αναλυτικά, μετατρέπει τις λέξεις σε διανύσματα συνεχών πραγματικών αριθμών, για παράδειγμα, η λέξη 'hat' θα μετατραπεί σε (... , 0,15,..., 0,23,..., 0,41,...). Το embedding γίνεται από έναν αραιό διανυσματικό χώρο υψηλών διαστάσεων, σε έναν πυκνό, χαμηλότερων διαστάσεων. Όπου κάθε διάσταση του διανύσματος αναπαριστά μορφολογικές ή σημασιολογικές πληροφορίες μίας λέξης.

Το word embedding σε ένα κείμενο, εκτός από νευρωνικά δίκτυα μπορεί να επιτευχθεί και με παραγοντοποίηση πινάκων. Μία σύννητες χρησιμοποιούμενη εφαρμογή είναι το isWord2Vec το οποίο χρησιμοποιεί νευρωνικό δίκτυο για



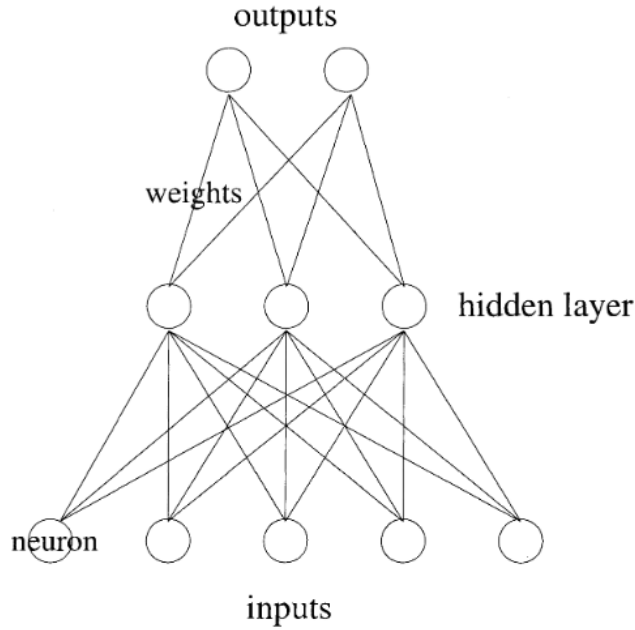
Σχήμα 6: Βιολογικός Νευρώνας
[9]

να μάθει τα **word embeddings** απ' το κείμενο, και 2 μοντέλα, το **Continuous bag-of-words (CBow)** και το **Skipgram(SG)**. Ακολουθεί επεξηγηματική αναφορά του κάθε μοντέλου.

- **CBow** : Λειτουργεί όπως το γνωστό αυτο-σorreστ, μαντεύοντας τη λέξη που θέλουμε να γράψουμε, βασισμένο σε ότι γράφτηκε πιο πριν. Για παράδειγμα στη πρόταση 'Το αυτοκίνητο _ γρήγορα', θα προέβλεπετε τη λέξη τρέχει. Καθώς παίρνει σαν **input** όλη τη πρόταση, χρησιμοποιείται σε μικρά σύνολα δεδομένων / **datasets**.
- **SG** : Έχει την αντρίστροφη λειτουργία, προβλέποντας το κείμενο δίνοντάς του τη λέξη. Καθώς χρειάζεται 1 λέξη για **input**, μπορεί να δουλέψει με μεγάλα σύνολα δεδομένων.

2.3 Εισαγωγή στα Νευρωνικά Δίκτυα

Τα νευρωνικά δίκτυα αποτελούν ένα εργαλείο που λύνει προβλήματα σε διάφορους τομείς όπως η Τεχνητή Νοημοσύνη, η Μηχανική Μάθηση και το **Deep Learning**. Η ιδέα πίσω απ' τα νευρωνικά δίκτυα προήλθε από τους βιολογικούς νευρώνες που βρίσκουμε στον ανθρώπινο εγκέφαλο. Μία απλοποιημένη απεικόνιση νευρώνα φαίνεται στο παραπάνω σχήμα, στο οποίο διακρίνουμε τους δενδρίτες, που παράγουν σήματα και τα στέλνουν μέσω του νευάξονα στο νευρώνα. Κατ' επέκταση κάθε νευρώνας επικοινωνεί με χιλιάδες άλλους μέσω των κόμβων ή αλλιώς συνάψεων. Αν και κάθε νευρώνας έχει χαμηλή επεξεργαστική ισχύ, πετυχαίνει αυξημένη ταχύτητα, και μάλιστα μεγαλύτερη από έναν υπερυ-



Σχήμα 7: Αρχιτεκτονική **Feed Forward** Νευρωνικού Δικτύου [10]

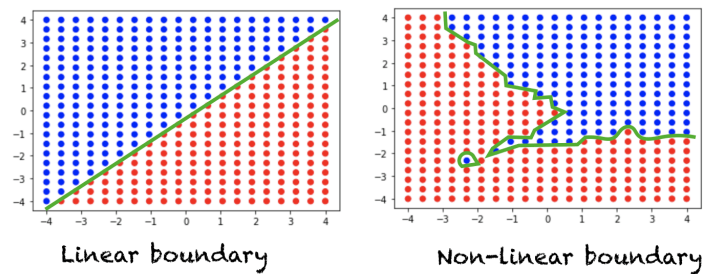
πολογιστή, με τη βοήθεια της ταυτόχρονης παραλληλοποίησης διαδικασιών σε όλες τις συνάψεις.[10]

2.3.1 **Feed-Forward** Νευρωνικά Δίκτυα

Τα τεχνητά λοιπόν νευρωνικά δίκτυα, αποτελούνται από 1 στρώμα εισαγωγής πληροφορίας (**input layer**), το οποίο αποτελείται πολλούς νευρώνες, ένα ή περισσότερα κρυφά στρώματα, και ένα τελικό στρώμα εξαγωγής πληροφοριών. Το $\sigma()$ ονομάζεται συνάρτηση μεταφοράς, N ο αριθμός των νευρώνων στο στρώμα εισόδου, $V_{i,j}$ τα βάρη, x_j οι είσοδοι στους νευρώνες του στρώματος εισόδου και T_i^{hid} το κατώφλι τιμών στους κρυφούς νευρώνες.

$$h_i = \sigma \left(\sum_{j=1}^N V_{ij} x_j + T_i^{hid} \right)$$

Συνάρτηση Ενεργοποίησης Η συνάρτηση ενεργοποίησης βρίσκεται σε ένα στρώμα του νευρωνικού δικτύου, παίρνει σαν παράμετρο πραγματικούς αριθμούς και επιστρέφει 0 ή 1. Ο σκοπός της συνάρτησης ενεργοποίησης είναι να μπορεί το μοντέλο να λύνει μη γραμμικά προβλήματα. Πιο συγκεκριμένα η διαφορά ενός γραμμικού με ενός μη-γραμμικού προβλήματος, όπως φαίνεται και στη παρα-



Σχήμα 8: Γραμμική / Μη γραμμική απεικόνιση προβλήματος
[11]

κάτω εικόνα, είναι η εξής : Έστω ένα πρόβλημα ταξινόμησης όπου θα πρέπει να χωρίσουμε μπλε τελείες απ' τις κόκκινες. Στην αριστερή εικόνα το πρόβλημα είναι γραμμικό ενώ στην αριστερή μη-γραμμικό, διαισθητικά καταλαβαίνουμε ότι το πρόβλημα θεωρείται γραμμικό επειδή μπορούμε να ξεχωρίσουμε τα δεδομένα τραβώντας μια ευθεία γραμμή, ενώ στη δεξιά εικόνα αυτό δεν ισχύει, οπότε το πρόβλημα της δεξιάς ονομάζουμε μη-γραμμικό.[11]

Σκοπός εν γέννη του νευρωνικού δικτύου είναι να μας δίνει στοχευμένα αποτελέσματα (που δίνονται απ' το χρήστη), δίνοντας τυχαία δεδομένα που αναπαριστούνται, ενδεχομένως, από το φυσικό μας περιβάλλον, τυχαιότητα στο περιβάλλον υπάρχει στον καιρό, στο μήκος των φύλλων ενός δέντρου, στο μέγεθος των σταγόνων της βροχής κ.α. Συνεπώς οι είσοδοι και οι έξοδοι είναι δυνατό να έχουν αυθαίρετη τιμή αλλά και μορφή. Όπως δυαδική (0/1), είτε συμβολική (κύκλος, τετράγωνο). Επομένως, καλύπτεται ένα ευρύ φάσμα εφαρμογής νευρωνικών δικτύων. Όπως για παράδειγμα, ένα νευρωνικό δίκτυο που θα υπολογίζει που θα πρέπει να στοχεύσουμε τη μπάλα για να μπει σε ένα καλάθι, δεδομένου ότι υπάρχει κάποια αντίσταση ανέμου. Ένα σύστημα που θα προβλέπει τον καιρό ή συστημα πρόβλεψης μελλοντικής τιμής μετοχών, και ου το καθεξής.

Το σύνολο πρέπει επομένως να είναι αρκετά μεγάλο ώστε το νευρωνικό δίκτυο να απομνημονεύει τα χαρακτηριστικά/τάσεις που είναι ενσωματωμένα στο σετ εκπαίδευσης. Από την άλλη πλευρά, εάν περιέχονται πάρα πολλές ασήμαντες λεπτομέρειες στο σετ εκπαίδευσης, το νευρωνικό δίκτυο μπορεί να σπαταλήσει τα βάρη του ανάλογα με το θόρυβο. Επομένως, μια συνετή επιλογή και αναπαράσταση των δεδομένων είναι κρίσιμης σημασίας για την επιτυχή υλοποίηση των νευρωνικών δικτύων. Σημειώστε ότι οι ορισμοί της επικύρωσης και του συνόλου δοκιμής αντιστρέφονται μεταξύ των συγγραφέων διαφορετικών πεδίων.[10]

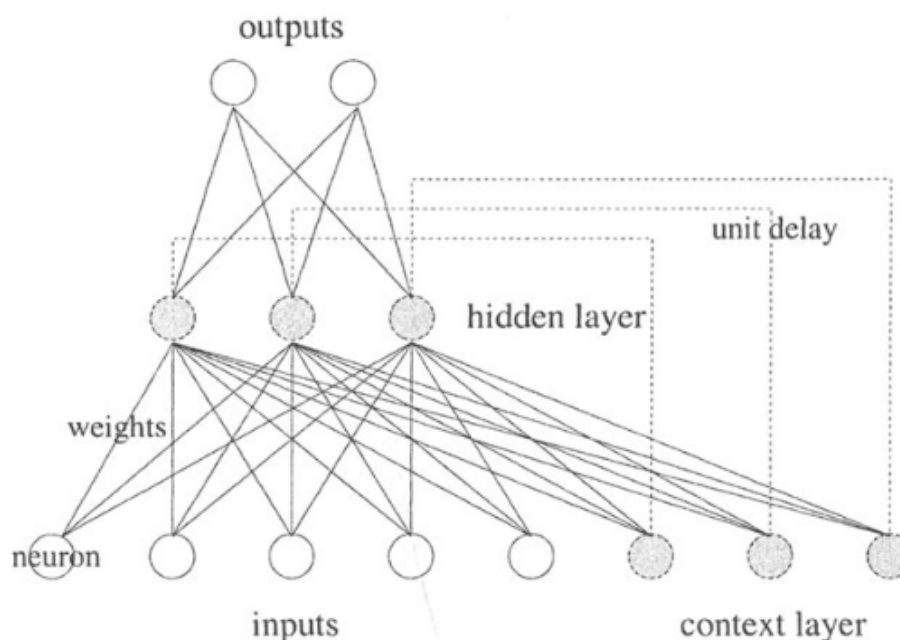
2.3.2 Recurrent Neural Networks

Τα νευρωνικά δίκτυα ανατροφοδосίας αν και δεν χρησιμοποιούνται για **Sentiment Analysis**) όπως το προκείμενο θέμα της πτυχιακής. Είναι όμως μία αρκετά ενδιαφέρουσα εκδοχή νευρωνικού δικτύου, με πολλές εφαρμογές σε σημερινά προβλήματα, όπως η αναγνώριση ομιλίας ή η μετάφραση μηχανής. Αναγνώριση ομιλίας είναι να ο τρόπος η μηχανή να ταυτίζει ένα ηχητικό δεδομένο με τη φωνή ενός ανθρώπου. Μετάφραση μηχανής είναι η μετάφραση ομιλίας ή γραπτού κειμένου από μία γλώσσα σε μία άλλη χρησιμοποιώντας ως μεσάζοντα τον H/Y.

Το **Recurrent Neural Network (RNN)**, έχει μνήμη στην οποία αποθηκεύει κάθε **output** για να το χρησιμοποιήσει σε επόμενες επαναλήψεις, παίρνει συνολικά 2 **inputs**, δεδομένα απ' το παρόν και απ' το κοντινό παρελθόν, και με ένα **loop** οι κόμβοι επεξεργάζονται ν-φορές τα δεδομένα. Αυτός ο αλγόριθμος είναι κατάλληλος για προβλήματα τεχνητής νοημοσύνης με αλληλοσυσχετιζόμενα δεδομένα (**sequential data**). Ένα παράδειγμα διαφοράς μεταξύ ανατροφοδοτούμενου και **Feed-Forward** νευρωνικού δικτύου είναι το εξής : Έστω ότι πρέπει να αναλύσουμε τη λέξη 'ανάλυση' με ένα νευρωνικό δίκτυο. Ένα **feed-forward** ξεκινώντας την ανάλυση απ' το γράμμα 'α' όταν φτάσει στο 'υ' θα το έχει ξεχάσει, κάτι που το καθιστά αδύνατο να μπορεί να προβλέψει το επόμενο γράμμα. Τέλος ενώ τα **feed-forward** αντιστοιχίζουν 1 **input** σε 1 **output (one to one)**, τα **RNN** υποστηρίζουν **one to many**, όπως αυτόματη συμπλήρωση κειμένου, **many to many** όπως μετάφραση. Συνεπώς, η διαφορά ανάμεσα στα **Feed-Forward** και τα **Recurrent** είναι ότι η πληροφορία στα πρώτα περνά 1 φορά από κάθε νευρώνα, ενώ στα τελευταία ανατροφοδοτείται.[12]

2.4 Sentiment Analysis

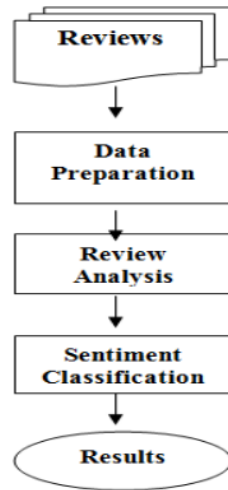
Το συναίσθημα που εκφράζεται στα σχόλια και τις κριτικές μπορεί να χρησιμοποιηθεί για διάφορους σκοπούς. Αυτά τα συναισθήματα μπορούν να χωριστούν σε κατηγορίες: θετικά, αρνητικά και ουδέτερα ή σε μια κλίμακα πέντε βαθμών, όπως πολύ καλό, καλό, ικανοποιητικό, κακό και πολύ κακό. Το **Sentiment Analysis** χρησιμοποιείται κυρίως από εταιρείες για να έχουν μία αντικειμενική εικόνα των προϊόντων τους ή για πολιτικούς για να βγάλουν συμπεράσματα για την κοινή γνώμη. Ο πιο αποδοτικός τρόπος για να ταξινομήσουμε ένα σχόλιο είναι να εξετάζουμε το σχόλιο σε κάθε ένα ταξινομητή διαδοχικά. Πιο απλά σε περίπτωση που ο ταξινομητής A δεν μπορέσει να ταξινομήσει την εγγραφή, αυτή να περνά στον B, το Γ και ου το καθεξής. Η εγγραφή που δεν μπορεί να ταξινομηθεί, παραλείπεται. Η δυσκολία της εξαγωγής συμπεράσματος



Σχήμα 9: Reccurent Neural Network Visualisation

έγγυται στην ανγνώριση συνακτικών και σημασιολογικών διαφοροποιήσεων, καθώς προτάσεις με διαφορετικές λέξεις και σύνταξη είναι σύνηθες να αντιστοιχούν στο ίδιο συναίσθημα. Σημαντικό είναι το σύστημα να λαμβάνει υπόψη τα συναισθήματα όλων των προτάσεων αντί για το μεγαλύτερο ποσοστό, αποκλείοντας τη μειοψηφία των σχόλιων. Σε αλγόριθμους ανάκτησης κειμένου προτάσεις με συντακτικά λάθη δεν συμπεριλαμβάνονται, και αντιμετωπίζονται σαν **True falses**, παρακάτω ακολουθεί εξήγηση για τα **True Falses**. Μη δημοφιλή γνώμες συχνά περιλαμβάνουν χρήσιμες πληροφορίες, οπότε καλό θα ήταν να συνυπολογίζονται με τις υπόλοιπες. Εκτός από την ανάλυση συναισθήματος με μηχανική μάθηση, υπάρχει και ένας εναλλακτικός τρόπος, που ονομάζεται **Lexicon-based approach**, μεταφράζεται ως λεξικολογική εκδοχή. Χωρίζεται σε 2 κατηγορίες **Dictionary-based** και **Corpus-based** που εξετάζονται παρακάτω, βοηθητικό είναι και το σχήμα 10, που παρουσιάζει ένα διάγραμμα με όλες τις τεχνικές **Sentiment Analysis**. Υπάρχει μία συγκεκριμένη διαδικασία που ακολουθείται στην ανάλυση συναισθήματος, ή οποία περιλαμβάνει :

1. Την προετημασία της δομής των κειμένων : Σε αυτή τη διαδικασία τα δεδομένα παίρνουν μία συγκεκριμένη φόρμα ώστε να μπορούν να επεξεργαστούν. Ενδιάμεσα στα κείμενα μπορεί να υπάρχουν και αχρείαστες πληροφορίες όπως **HTML tags**, πληροφορίες για το **URL** κ.α. Τα οποία



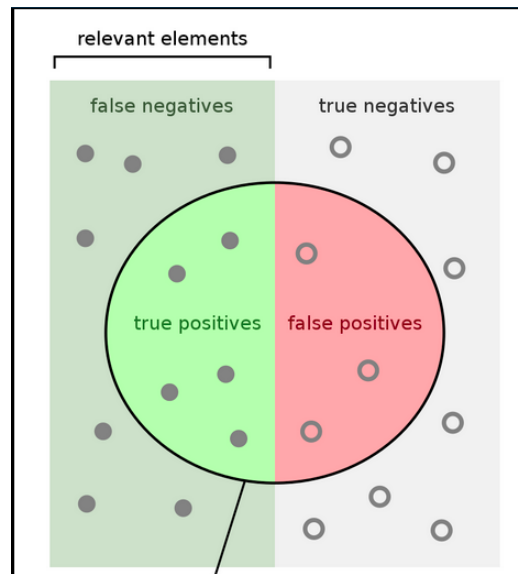
Σχήμα 10: Sentiment Analysis model
[15]

πρέπει να διαγραφούν.

2. Ανάλυση δεδομένων : Σε αυτό το βήμα γίνεται εξάγεται η χρήσιμη πληροφορία του κειμένου που είναι το συναίσθημα του συγγραφέα. Για αυτό το αποτέλεσμα προηγούνται διάφορες υπολογιστικές διαδικασίες, συνήθως αυτές είναι το **POS tagging** και το **Negation tagging**.
3. Κατηγοριοποίηση συναισθημάτων : Υπάρχουν 2 επιλογές για αυτή τη διαδικασία, η πρώτη ονομάζεται **Sentiment orientation approach** και η δεύτερη **Machine learning approach**. Σε πρώτη φάση εξάγουμε συμπεράσματα για την άποψη του συγγραφέα για όλο το κείμενο, ενώ σε δεύτερη έρχεται η άποψη του συγγραφέα για κάθε πρόταση.

[13, 14, 15]

Sensitivity and specificity Αυτοί οι όροι χρησιμοποιούνται για να ταξινομήσουμε δεδομένα ως θετικά ή αρνητικά. Στην συγκεκριμένη πτυχιακή χρησιμοποιείται αλγόριθμος που δείχνει θετικά και αρνητικά συναισθήματα. Επομένως για αυτά τα αποτελέσματα υπάρχουν 4 ενδεχόμενα. Κάποιο θετικό συναίσθημα να το χαρακτηρίσει σωστά (**True positive**), κάποιο θετικό συναίσθημα να το χαρακτηρίσει λανθασμένα (**False positive**), το αρνητικό να το χαρακτηρίσει σωστά (**True negative**), και το αρνητικό να το χαρακτηρίσει λανθασμένα (**False negative**). Στο παρακάτω σχήμα φαίνεται ξεκάθαρα η διαφορά των



Σχήμα 11: Sensitivity and specificity diagram
[16]

ενδεχομένων.

Αυτόματη ταξινόμηση κειμένων Αυτόματη ταξινόμηση είναι η διαδικασία όπου το πρόγραμμα καθεαυτό αποφασίζει σε ποια κλάση ανήκει το κάθε κείμενο. Ως αποτέλεσμα, κάθε κείμενο d ανήκει σε μία κλάση c , και έτσι σχηματίζονται σύνολα κλάσεων τα οποία περιέχουν σύνολα από κείμενα. Έστω το σύνολο το κειμένων $D = \{d_1, d_2, d_3, \dots, d_n\}$ και το σύνολο των κλάσεων $C = \{c_1, c_2, c_3, \dots, c_n\}$, κάθε ζευγάρι κειμένου με την αντίστοιχη κλάση ανήκουν στα υποσύνολα τους, $c_i, d_i \in D \times C$. Στη διαδικασία της ταξινόμησης, ο αλγόριθμος λαμβάνει ως παραμέτρους δεδομένα για δοκιμές **testing set** και δεδομένα που βοηθούν στην μάθηση, **training set**. Με αυτή τη μέθοδο μπορούμε να δημιουργήσουμε πρόγραμμα που ταξινομεί δεδομένα όπως και ένας άνθρωπος, απ' την άλλη πλευρά υπάρχουν και κάποια μειονεκτήματα. Χρειάζεται αξιόλογος χρόνος για να ορίσει κάποιος τα χαρακτηριστικά που θα πρέπει να ανιχνεύει ένα πρόγραμμα, να ορίσει τις κλάσεις και φυσικά να εκπαιδευτεί. Μάλιστα σε περίπτωση που ο όγκος των δεδομένων είναι μεγάλος, το να ορίσει κάποιος τους κανόνες και τα χαρακτηριστικά είναι μία κουραστική διαδικασία στην οποία υπάρχει αυξημένη πιθανότητα να γίνει λάθος. Τέλος η ακρίβεια των αποτελεσμάτων ποικίλλει ανάλογα με το σύνολο δεδομένων, δημιουργώντας αμφιβολίες για το ποιο σύνολο είναι πιο ποιοτικό και αντιπροσωπεύει την πραγματικότητα. [13]

2.4.1 Ανασκόπηση έρευνας στο **Sentiment Analysis**

Εκτός από την ταξινόμηση κειμένων σε κλάσεις βάση συναισθημάτων, έχουν γίνει και άλλες συναφή προσπάθειες ανάλυσης συναισθημάτων σε χαμηλότερο επίπεδο, με στόχο την παραγωγή πιο αντιπροσωπευτικών αποτελεσμάτων. Κάποιες από αυτές είναι η εύρεση συναισθήματος λέξεων, εκφράσεων, προτάσεων και θεμάτων. Υπάρχουν διάφοροι αλγόριθμοι για την επιτυχή ανάλυση, όπως ο **Naive Bayes**, **Maximum Entropy**, **Support Vector Machine** και η μάθηση χωρίς επίβλεψη.

Εύρεση συναισθήματος λέξεων Στη δημοσίευση των Χατζιβασιλόγλου και **McKeown**, πραγματοποιείται **sentiment analysis** εξετάζοντας τις λέξεις ξεχωριστά, και συγκεκριμένα τα επίθετα. Αναφέρεται πως 2 συνώνυμες λέξεις είναι εφικτό να αποδίδουν διαφορετικό συναίσθημα, όπως οι λέξεις απλός και απλουστευμένος, η πρώτη δηλώνει επιθυμία ενώ η δεύτερη όχι. Σε αντίθεση με τα αντώνυμα, όπως οι λέξεις ζεστός/κρύος, ο συναισθηματικός προσανατολισμός τους είναι διαφορετικός και μπορεί να γίνει εύκολα αντιληπτός χωρίς την υπόλοιπη πρόταση. Αναγνωρίζοντας τον σημασιολογικό προσανατολισμό των λέξεων, μπορούμε να δημιουργούμε σύστημα, που να αυτοματοποιούν αυτή τη διαδικασία. Για παράδειγμα οι λέξεις δίκαιο και αυθεντικό και κακό και κατεστραμμένο και βίαιο, έχουν διαφορετική σημασιολογία από τις, δίκαιο και βίαιο και κατεστραμμένο και αυθεντικό, οι λέξεις καθεαυτές δεν μας αποδίδουν τη σημασιολογική έννοια της πρότασης, και κατ' επέκταση το συναίσθημα του συγγραφέα. Στην περίπτωση που η συνδετική μας λέξη είναι το αλλά και όχι το και, το νόημα αντιστρέφεται. Το σύστημα έχει ακρίβεια > 90%, ενώ τελικός στόχος είναι να μπορεί ο αλγόριθμος να ξεχωρίζει τα αντόνυμα και συνώνυμα λέξεις. Εν κατακλείδι, σκοπός είναι η δημιουργία ενός συστήματος που να εξάγει συμπεράσματα για το ύφος του συγγραφέα χρησιμοποιώντας πληροφορίες που εξάγονται έμμεσα. [17]

Τα βήματα του αλγορίθμου που χρησιμοποιήθηκε είναι τα εξής :

1. Επίθετα και μορφολογικές σχέσεις, εξάγονται από το κείμενο
2. Πληροφορίες από διάφορες διασυνδέσεις συγκρίνονται, για καταλάβουμε αν οι προτάσεις με τα επίθετα τους, έχουν όλες το ίδιο ή διαφορετικό συναίσθημα.
3. Ταξινομούμε τα επίθετα σε ομάδες ανάλογα με τη σημασιολογία τους.
4. Συγκρίνουμε την μέση συχνότητα των ομάδων, και η ομάδα με τη μεγαλύτερη συχνότητα, χαρακτηρίζεται ως θετική.

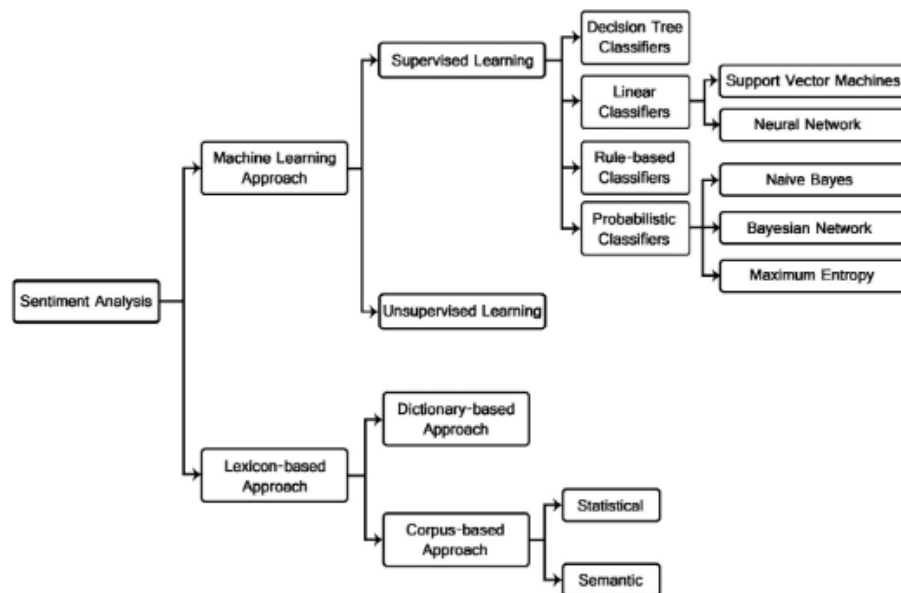
Εύρεση συναισθήματος μονάδων Για την περιγραφή του αλγόριθμου αυτού θα πρέπει πρώτα να διευκρινιστεί τι σημαίνει συναισθηματική μονάδα. Μία συναισθηματική μονάδα αποτελείται από : το συναίσθημα, το κατηγορούμενο, και τα επίθετα στο κατηγορούμενο. Στον αλγόριθμο που συνοδεύεται από τη δημοσίευση, χρησιμοποιούνται αυτές οι μονάδες για την εξαγωγή αποτελεσμάτων. Για παράδειγμα μία κριτική μπορεί να έχει 3 συναισθηματικές μονάδες όπως η παρακάτω :

Η κάμερα έχει εξαιρετικούς φακούς, αλλά η τιμή είναι υψηλή. Πιστεύω πως δεν υπάρχει πρόβλημα με την ποιότητα του φορτιστή.

1. Θετικό → εξαιρετικοί φακοί, μη προβληματικός φορτιστής
2. Αρνητικό → τιμή

Όπως παρατηρούμε στις προτάσεις, μία μηχανή θα μπορούσε κάλλιστα να παρερμηνεύσει το νόημα του κειμένου βγάζοντας το αντίθετο συμπέρασμα. Στην 1η πρόταση 'αλλά η τιμή είναι υψηλή' το υψηλή ταξινομείται ως αρνητικό χαρακτηριστικό, αλλά θα μπορούσε να παρερμηνευτεί από μία μηχανή ως θετικό, με τη λογική ότι η λέξη υψηλή θα χαρακτήριζε την ποιότητα των φακών. Στη συγκεκριμένη δημοσίευση χρησιμοποιήθηκε η τενχική **transfer learning** όπως και στην πτυχιακή. Συγκεκριμένα χρησιμοποιήθηκε ένα υπάρχον σύστημα μετάφρασης, που οι αλγόριθμοι του αντικαταστάθηκαν με διαφορετικούς, που περιλαμβάνουν μετάφραση συναισθήματος. Το ζορπυς που χρησιμοποιήθηκε και οι κριτικές των πειραμάτων ήταν γραμμένα στην Ιαπωνική γλώσσα.[14]

Τεχνικές ταξινόμησης συναισθημάτων Οι προσεγγίσεις μηχανικής μάθησης, βασισμένες στο λεξικό και υβριδικές προσεγγίσεις ταξινόμησης συναισθημάτων μπορούν να χωριστούν χαλαρά σε τρεις κατηγορίες. Η Μηχανική Προσέγγιση Εκμάθησης (ML) χρησιμοποιεί γνωστές μεθόδους μηχανικής εκμάθησης καθώς και γλωσσικά χαρακτηριστικά. Ένα λεξικό συναισθημάτων ή μια συλλογή γνωστών και προμεταγλωττισμένων όρων συναισθήματος, χρησιμοποιείται στην προσέγγιση που βασίζεται στο λεξικό. Διαχωρίζεται σε δύο τύπους προσεγγίσεων: προσεγγίσεις που βασίζονται σε λεξικό και προσεγγίσεις που βασίζονται σε σώματα, οι οποίες χρησιμοποιούν στατιστικές ή σημασιολογικές μεθόδους για τον προσδιορισμό της πολικότητας των συναισθημάτων. Η υβριδική προσέγγιση είναι αρκετά διαδεδομένη, με τα λεξικά συναισθημάτων να παίζουν ζωτικό ρόλο στην πλειονότητα των λύσεων. Όπως αναφέρθηκε προηγουμένως, οι διάφορες τεχνικές και οι πιο δημοφιλείς αλγόριθμοι SC απεικονίζονται στο παρακάτω σχήμα. [13, 17]



Σχήμα 12: Sentiment Analysis Diagram

Προσεγγίση μέσω Μηχανικής Μάθησης Το SA (Sentiment Analysis) αντιμετωπίζεται ως τυπικό πρόβλημα ταξινόμησης κειμένου με συντακτικά ή/και γλωσσικά χαρακτηριστικά και η τεχνική μηχανικής μάθησης χρησιμοποιεί γνωστούς αλγόριθμους για την επίλυσή του. Το Πρόβλημα ταξινόμησης κειμένου ορίζεται ως εξής: Έχουμε ένα σύνολο εγγραφών εκπαίδευσης $D = X_1, X_2, \dots, X_n$, καθένα από τα οποία επισημαίνεται με μια κλάση. Το μοντέλο ταξινόμησης συνδέεται με μία από τις ετικέτες κλάσης βάσει χαρακτηριστικών στην υποκείμενη εγγραφή. Στη συνέχεια, το μοντέλο χρησιμοποιείται για να προβλέψει μια ετικέτα κλάσης για μια δεδομένη παρουσία άγνωστης κλάσης. Όταν εκχωρείται μόνο μία ετικέτα σε μια εγγραφή, γίνεται μια χρονοβόρα κατηγοριοποίηση. Όταν μια πιθανολογική τιμή ετικετών καταχωρείται σε μία εγγραφή, αυτό το ονομάζουμε **soft classification**. [13]

Απαιτείται η ύπαρξη επισημασμένων αρχείων κατάρτισης για το **supervised learning**. Στη βιβλιογραφία, υπάρχουν αρκετοί διαφορετικοί τύποι ταξινομητών. Παρακάτω παραθέτονται μερικοί από τους πιο δημοφιλείς ταξινομητές ανάλυσης συναισθήματος.

Οι πιθανοτικοί ταξινομητές είναι ένας τύπος ταξινομητή που βασίζεται στην πιθανότητα. Τα μοντέλα μιγμάτων χρησιμοποιούνται από πιθανολογικούς ταξινο-

μητές για την ταξινόμηση δεδομένων. Κάθε κατηγορία θεωρείται ως συστατικό του μείγματος στο μοντέλο του μείγματος. Κάθε συστατικό μείγματος είναι ένα παραγωγικό μοντέλο που υπολογίζει την πιθανότητα δειγματοληψίας μιας συγκεκριμένης φράσης. Οι γενεσιουργοί ταξινομητές είναι ένα άλλο όνομα για αυτούς τους τύπους ταξινομητών. Οι ακόλουθες ενότητες παρουσιάζουν τρεις από τους πιο γνωστούς πιθανολογικούς ταξινομητές. [17]

2.4.2 Μέθοδοι **Sentiment Analysis**

lexicon-based Σε πολλές εργασίες ταξινόμησης συναισθημάτων, όπως και στη συγκεκριμένη εργασία χρησιμοποιούνται λέξεις γνώμης. Οι λέξεις θετικής γνώμης εκφράζουν ορισμένες επιθυμητές καταστάσεις, ενώ οι λέξεις αρνητικής γνώμης εκφράζουν ορισμένες ανεπιθύμητες καταστάσεις. Το λεξικό γνώμης αναφέρεται στη συλλογή φράσεων απόψεων και ιδιωματισμών. Για τη δημιουργία ή τη συλλογή της λίστας λέξεων γνώμης, υπάρχουν τρεις κύριες τεχνικές. Η χειροκίνητη προσέγγιση διαρκεί πολύ και σπάνια χρησιμοποιείται μόνη της. Χρησιμοποιείται συχνά σε συνδυασμό με τις άλλες δύο αυτοματοποιημένες διαδικασίες ως τελικός έλεγχος για την αποφυγή σφαλμάτων που προκαλούνται από αυτοματοποιημένες μεθόδους. Οι δύο αυτοματοποιημένες προσεγγίσεις συζητούνται στις παρακάτω παραγράφους. [18]

Dictionary-based Στο dictionary-based, ένα μικρό σύνολο λέξεων γνώμης με γνωστούς προσανατολισμούς συλλέγεται χειροκίνητα. Στη συνέχεια, η λίστα επεκτείνεται αναζητώντας συνώνυμα και αντώνυμα σε γνωστά σώματα όπως τα **WordNet**, **thesaurus**. Οι λέξεις που ανακαλύφθηκαν πρόσφατα προστίθενται στη λίστα σποράς και η διαδικασία επανάληψης συνεχίζεται. Όταν δεν υπάρχουν νέες λέξεις, η επαναληπτική διαδικασία σταματά, η μη αυτόματη επιθεώρηση μπορεί να πραγματοποιηθεί μετά την ολοκλήρωση της διαδικασίας για την αφαίρεση ή την επιδιόρθωση προβλημάτων. Η τεχνική που βασίζεται σε λεξικό έχει ένα σημαντικό μειονέκτημα: είναι αδύνατο να βρεθούν λέξεις γνώμης με συγκεκριμένους προσανατολισμούς, τομέα και πλαίσιο χρησιμοποιώντας αυτήν τη μέθοδο. Για να προσδιορίσουν συναισθηματικές προτάσεις στη διαφήμιση με βάση τα συμφραζόμενα, ανέπτυχθηκε μια τεχνική που βασίζεται σε λεξικό. Μια στρατηγική διαφήμισης για τη βελτίωση της συνάφειας των διαφημίσεων και της εμπειρίας του χρήστη. Χρησιμοποιήθηκε συντακτική ανάλυση και ένα λεξικό συναισθημάτων για να αντιμετωπίσει η εξαγωγή λέξεων θέματος και τον προσδιορισμό της στάσης του πελάτη στην εξαγωγή λέξεων-κλειδιών διαφήμισης και πρόσφεραν μια λύση βασισμένη σε κανόνες. Τα ευρήματά τους έδειξαν ότι η προτεινόμενη μέθοδος για την εξαγωγή λέξεων-κλειδιών διαφήμισης και την

επιλογή διαφημίσεων είναι αποτελεσματική. [18]

Corpus-based Η μέθοδος που βασίζεται σε σώματα βοηθά στην αναζήτηση λέξεων γνώμης με προσανατολισμούς που σχετίζονται με το πλαίσιο. Οι προσεγγίσεις του βασίζονται σε γραμματικά μοτίβα ή μοτίβα που εμφανίζονται μαζί με μια αρχική λίστα όρων γνώμης σε ένα μεγάλο σώμα για να εντοπίσουν άλλες λέξεις γνώμης. Οι Χατζηβασιλόγλου και **McKeown** αντιπροσώπευαν μία από αυτές τις στρατηγικές. Ξεκίνησαν με μια λίστα επιθέτων απόψεων και τα χρησιμοποίησαν σε συνδυασμό με ένα σύνολο γλωσσικών περιορισμών για να βρουν περισσότερους όρους επίθετων γνώμης και τους προσανατολισμούς τους. Οι περιορισμοί ισχύουν για συνδέσεις όπως **AND, OR, BUT, EITHER-OR**, και ούτω καθεξής. Για παράδειγμα, ο σύνδεσμος **AND** δηλώνει ότι τα συνενωμένα επίθετα έχουν συνήθως τον ίδιο προσανατολισμό. Αυτή η έννοια είναι γνωστή ως σταθερότητα συναισθήματος, αλλά δεν ισχύει πάντα στην πράξη. Ωστόσο, υπάρχουν αντικρουόμενοι όροι όπως αλλά, ωστόσο, που υποδηλώνουν ότι η γνώμη κάποιου έχει αλλάξει. Η μάθηση χρησιμοποιείται σε ένα τεράστιο σώμα για να ανιχνεύσει εάν δύο ενωμένα επίθετα έχουν τον ίδιο ή διαφορετικούς προσανατολισμούς. Οι σχέσεις μεταξύ των επιθέτων σχηματίζουν στη συνέχεια ένα γράφημα, το οποίο στη συνέχεια ομαδοποιείται για να δώσει δύο σύνολα λέξεων: θετική και αρνητική. [18]

Επειδή είναι δύσκολο να προετοιμαστεί ένα μεγάλο σώμα για να καλύψει όλες τις αγγλικές λέξεις, η χρήση της προσέγγισης που βασίζεται σε σώματα από μόνη της δεν είναι τόσο αποτελεσματική όσο η χρήση της προσέγγισης που βασίζεται σε λεξικό, αλλά αυτή η προσέγγιση έχει ένα σημαντικό πλεονέκτημα στο ότι μπορεί να βοηθήσει στην εύρεση του τομέα και του πλαισίου συγκεκριμένες λέξεις γνώμης και τους προσανατολισμούς τους χρησιμοποιώντας ένα σώμα τομέα.

Εύρεση συναισθήματος Επειδή υπάρχει κάποια ασάφεια σχετικά με τη διαφορά μεταξύ γνώμης, συναισθήματος και συναισθήματος, όρισαν τη γνώμη ως μια μεταβατική έννοια που αντανακλά τη στάση απέναντι σε μια οντότητα. Η ανάλυση συναισθήματος θεωρείται μερικές φορές μια εργασία **NLP** για την ανακάλυψη απόψεων σχετικά με μια οντότητα. Και επειδή υπάρχει κάποια ασάφεια σχετικά με τη διαφορά μεταξύ γνώμης, συναισθήματος και συναισθήματος, όρισαν τη γνώμη ως μια μεταβατική έννοια που αντανακλά τη στάση απέναντι σε μια οντότητα. Αυτό το συναίσθημα εκφράζει ένα συναίσθημα ή ένα συναίσθημα, ενώ το συναίσθημα εκφράζει μια στάση. Ο **Plutchik** υποστήριξε ότι υπάρχουν οκτώ βασικά και πρωτότυπα συναισθήματα: χαρά, λύπη, οργή, φόβος, εμπιστοσύνη,

αηδία, έκπληξη και προσδοκία. Η ανίχνευση συναισθημάτων (ED) είναι μια εργασία που μπορεί να ταξινομηθεί ως εργασία SA. Ενώ η SA ασχολείται κυρίως με την έκφραση θετικών ή αρνητικών ιδεών, η ED ασχολείται με την αναγνώριση διαφόρων συναισθημάτων από το κείμενο. Το ED μπορεί να υλοποιηθεί ως εργασία Ανάλυσης Συναισθήματος χρησιμοποιώντας είτε μια προσέγγιση ML είτε μια μέθοδο που βασίζεται στο λεξικό, ωστόσο η προσέγγιση που βασίζεται στο λεξικό χρησιμοποιείται πιο συχνά. [18]

Οι Lu και Lin υποστήριζαν την ED σε επίπεδο πρότασης. Πρότειναν μια μέθοδο εξόρυξης κειμένου για την ανίχνευση συναισθημάτων σε αγγλικές προτάσεις που είναι ενσωματωμένες σε μεμονωμένα γεγονότα. Η μέθοδός τους βασίστηκε στην κατανομή πιθανοτήτων κοινών αμοιβαίων ενεργειών μεταξύ του υποκειμένου και του αντικειμένου του συμβάντος. Για να αναγνωρίσουν ένα σύστημα αναγνώρισης συναισθημάτων γεγονότων, συνδύασαν τεχνικές εξόρυξης κειμένου και σημασιολογικής επισήμανσης ρόλων με μια σειρά από ζεύγη οντοτήτων αναφοράς και χειροποίητους αλγόριθμους δημιουργίας συναισθημάτων. Δεν χρησιμοποιήθηκαν λεξιλογικές πηγές ή βάσεις γνώσεων μεγάλης κλίμακας. Έδειξαν ότι η μέθοδός τους ήταν αποτελεσματική στον εντοπισμό χαρούμενων, αρνητικών και ουδέτερων συναισθημάτων. Έδειξαν ότι το ζήτημα ανίχνευσης συναισθήματος εξαρτάται από το πλαίσιο. [18]

Ο Balahur παρουσίασε μια υβριδική στρατηγική που συνδύαζε τη μηχανική μάθηση και μια προσέγγιση βασισμένη στο λεξικό. Πιο συγκεκριμένα, μια προσέγγιση που βασίζεται σε πληροφορίες κοινής λογικής που περιέχονται στη βάση γνώσεων του σώματος συναισθημάτων (EmotiNet). Ισχυρίστηκαν ότι τα συναισθήματα συχνά απεικονίζονται όχι με τη χρήση λέξεων που έχουν συναισθηματική σημασία, όπως το χαρούμενο, αλλά με την περιγραφή εμπειριών από την πραγματική ζωή που οι αναγνώστες αναγνωρίζουν ότι σχετίζονται με ένα συγκεκριμένο συναίσθημα. Για να πετύχουν τον στόχο τους, χρησιμοποίησαν τους αλγόριθμους SVM και SVM-SO. Έδειξαν ότι η τεχνική που βασίζεται στο EmotiNet είναι η καλύτερη για τον εντοπισμό συναισθημάτων σε περιπτώσεις όπου δεν υπάρχουν λέξεις που να σχετίζονται με το συναίσθημα. Έδειξαν ότι η αναγνώριση συναισθημάτων από κείμενα όπως αυτά στο σώμα συναισθημάτων ISEAR (τα οποία έχουν λίγες ή καθόλου λεξιλογικές ενδείξεις για να επηρεάσουν) μπορεί να αντιμετωπιστεί αποτελεσματικά με αλγόριθμους που βασίζονται στην κοινή λογική. Η χρήση του EmotiNet παράγαγε καλύτερα αποτελέσματα από τις προσεγγίσεις που χρησιμοποιούν εποπτευόμενη μάθηση σε μεγαλύτερο σύνολο εκπαίδευσης ή λεξιλογικές γνώσεις. [18]

Affect Analysis Η Ανάλυση Επιρροής (AA) είναι μία διαδικασία αναγνώρισης συναισθημάτων που προκαλούνται από μια συγκεκριμένη σημειωτική μέθοδο. Ένα μοντέλο ανάλυσης επιρροής έχει προταθεί από τη **Neviarous & kaya**. Το συμβολικό σύνθημα, η συντακτική δομή, η ανάλυση σε επίπεδο λέξης, σε επίπεδο φράσης και σε επίπεδο πρότασης είναι τα πέντε στάδια του **AAM** τους. Πολλές από τις εφαρμογές που παρουσιάζονται στο χρησιμοποίησαν αυτό το **AAM**. Επίσης δημιούργησαν ένα σύστημα ανάλυσης στάσης βασισμένο στην αρχή της σύνθεσης και μια νέα προσέγγιση για την αντιμετώπιση της σημασιολογίας των ρημάτων. Μέσα από ένα δίκτυο προσωπικών ιστοριών, οι άνθρωποι μοιράζονται προσωπικές εμπειρίες, σκέψεις, απόψεις, συναισθήματα, πάθη και εξομολογήσεις. Στο έργο της ανάλυσης της στάσης κειμένου, το σύστημά τους παρήγαγε αξιόπιστα ευρήματα, σύμφωνα με την αξιολόγησή τους. [18]

Χρησιμοποιώντας μια τεχνική βασισμένη στο λεξικό, όπως προτείνεται από τους **Kesht-kar & Inkpen**, θα μπορούσαν να χρησιμοποιηθούν λέξεις επηρεασμού συναισθημάτων. Ανέπτυξαν μια προσέγγιση βοοστραπινγκ που βασίζεται σε συναφείς και λεξιλογικούς παράγοντες για την εύρεση παραφράσεων σε μη παράλληλους κορμούς και την εξαγωγή τους για όρους συναισθημάτων. Ξεκίνησαν με λίγη ποσότητα σπόρων (**WordNetAffect emotion words**). Η μέθοδός τους ανακάλυψε μοτίβα εξαγωγής για έξι διαφορετικούς τύπους συναισθημάτων. Εξήγαγαν παραφράσεις από σχολιασμένα ιστολόγια και άλλα σύνολα δεδομένων ως κείμενα. Χρησιμοποίησαν δεδομένα από ζωντανά περιοδικά, ιστολόγια, κείμενο, παραμύθια και σχολιασμένα ιστολόγια για να δημιουργήσουν την έρευνά τους. Έδειξαν ότι η μέθοδός τους απέδωσε καλά στο σύνολο δεδομένων τους. [18]

Η βασισμένη σε κείμενο ανάλυση επηρεασμού (AA) των ιαπωνικών αφηγήσεων από τον **Aozora Bunko** έχει μελετηθεί από τους **Ptaszynski**. Εξέτασαν το ζήτημα της αναγνώρισης συναισθημάτων που σχετίζονται με το πρόσωπο/χαρακτήρα σε παραμύθια στη μελέτη τους. Πρώτα ανέσυραν το θέμα των συναισθημάτων από μια πρόταση χρησιμοποιώντας αναφορική ανάλυση έκφρασης και στη συνέχεια χρησιμοποίησαν τη διαδικασία ανάλυσης συναισθήματος για να προσδιορίσουν σε τι είδους συναισθηματική κατάσταση βρισκόταν κάθε χαρακτήρας σε κάθε σημείο της ιστορίας. Ο **Mohammad** πρωτοστάτησε στη μελέτη της AA μέσω **e-mail** και δημοσιεύσεων. Ερεύνησε το σώμα ηλεκτρονικού ταχυδρομείου της Ενρον και ανακάλυψε σημαντικές διαφορές στον τρόπο με τον οποίο άνδρες και γυναίκες χρησιμοποιούν τους όρους συναισθημάτων στα **email** στο χώρο εργασίας. Με το **crowdsourcing**, κατασκεύασε ένα λεξικό που περιέχει χειροκίνητους σχολιασμούς των συσχετισμών μιας λέξης με θετική/αρνητική πολικότητα, καθώς και τα οκτώ βασικά συναισθήματα. Το

χρησιμοποίησε για να παρακολουθεί και να αξιολογεί την κατανομή των όρων συναισθημάτων σε βιβλία και email. Εξετάζοντας βιβλία και παραμύθια, καθιέρωσε την έννοια της πυκνότητας των λέξεων του συναισθήματος. Απέδειξε ότι τα παραμύθια είχαν σημαντικά μεγαλύτερο εύρος συναισθηματικής πυκνότητας λέξεων από τα μυθιστορήματα. [18]

2.5 Transfer Learning

Ένα από τα κύρια προβλήματα που αντιμετωπίζουμε στην εκπαίδευση μοντέλων, είναι η βελτίωση ενός μοντέλου. Τα δεδομένα που χρειάζονται για την εκπαίδευση και το **testing** ενός μοντέλου πρέπει να είναι αντλημένα από την ίδια συλλογή δεδομένων. Συγκεκριμένα κάθε φορά που θέλουμε να προσθέσουμε δεδομένα, χρειάζεται να εκπαιδεύσουμε το μοντέλο απ' την αρχή. Σε μικρή κλίμακα το πρόβλημα δεν είναι σημαντικό, καθώς ένα μοντέλο **sentiment analysis** για μερικές χιλιάδες προτάσεις, χρειάζεται λιγότερο από 5 λεπτά για να δημιουργηθεί. Σε μεγάλη κλίμακα όμως το πρόβλημα είναι ανέφικτο ή πολύ κοστοβόρο. Μία περίπτωση χρήσης του **transfer learning**, όπως χρησιμοποιείται και στην παρούσα πτυχιακή, είναι η αυτόματη ταξινόμηση κριτικών σε θετικές και αρνητικές. Για να γίνει αυτό, θα πρέπει πρώτα να ταξινομήσουμε χειροκίνητα κάποια δεδομένα και σε κάποια να βάλουμε **labels** καθώς δεν υπάρχουν διάφορες κατηγορίες ίδιων προϊόντων. Όπως πχ κάμερες, ώστε να μπορεί ο ταξινομητής να αυτοματοποιήσει τη διαδικασία. Με τη βοήθεια του **transfer learning**, μπορούμε να πετύχουμε κάτι τέτοιο σε λιγότερο χρόνο με μικρότερο κόστος, χωρίς να χρειαστεί οποιαδήποτε χειροκίνητη διαδικασία. [19]

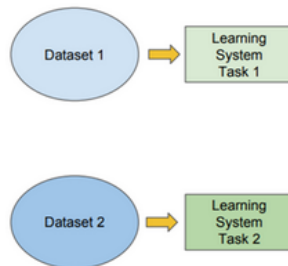
Οι παραδοσιακοί αλγόριθμοι μηχανικής μάθησης λειτουργούν απομονωμένα και λαμβάνουν χώρα αποκλειστικά με βάση συγκεκριμένες διεργασίες, σύνολα δεδομένων. Το **transfer learning** μας επιτρέπει να χρησιμοποιούμε τις πληροφορίες μοντέλων που έχουν εκπαιδευτεί προηγουμένως (χαρακτηριστικά, βάρη, κ.λπ.) για να εκπαιδεύσουμε νεότερα μοντέλα. Δεν υπάρχει γνώση που να μπορεί να μεταφερθεί από το ένα μοντέλο στο άλλο, κύριος στόχος είναι να μεταφέρουμε γνώση από προηγούμενες δραστηριότητες σε νεότερες. Εάν έχουμε πολύ περισσότερα δεδομένα για την εργασία **T1**, μπορούμε να τα χρησιμοποιήσουμε για να μάθουμε από αυτήν και να εφαρμόσουμε αυτά που μάθαμε στην εργασία **T2** (η οποία έχει λιγότερα δεδομένα). Ορισμένες ιδιότητες χαμηλού επιπέδου, όπως άκρες, φόρμες, γωνίες και ένταση, μπορούν να κοινοποιηθούν σε εργασίες στον τομέα του **computer vision**, επιτρέποντας τη μεταφορά γνώσης μεταξύ εργασιών. Επομένως όπως φαίνεται στο σχήμα, οι πληροφορίες από μια προηγούμενη εργασία χρησιμεύουν ως πρόσθετη είσοδο στην εκμάθηση νέας εργασίας. [19]

Traditional ML

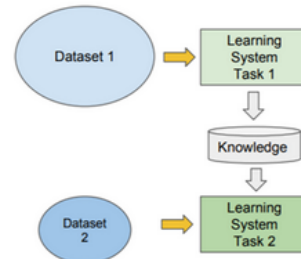
vs

Transfer Learning

- Isolated, single task learning:
 - Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in other tasks



- Learning of a new tasks relies on the previous learned tasks:
 - Learning process can be faster, more accurate and/or need less training data



Traditional Learning vs Transfer Learning

Σχήμα 13: Traditional vs Transfer Learning

Πριν ξεκινήσουμε τη διαδικασία αυτή, θα πρέπει να λάβουμε υπόψη :

- Τι πρέπει να μεταφερθεί : Αυτό είναι το πρώτο και πιο κρίσιμο βήμα σε όλη τη διαδικασία. Προκειμένου να αυξήσουμε την απόδοση της εργασίας, προσπαθούμε να καταλάβουμε ποια μέρη της γνώσης μπορούν να μεταφερθούν από το αρχικό μοντέλο στο τελικό. Όταν προσπαθούμε να απαντήσουμε σε αυτό το ζήτημα, προσπαθούμε να προσδιορίσουμε ποιες πτυχές της γνώσης είναι μοναδικές.
- Πότε να μεταφέρουμε γνώση : Υπάρχουν φορές που η μεταφορά γνώσης μόνο για χάρη της μπορεί να κάνει τα πράγματα χειρότερα παρά καλύτερα (γνωστή και ως αρνητική μεταφορά). Στόχος του αλγορίθμου είναι η βελτίωση αντί για την επιδείνωση της απόδοσης/αποτελεσμάτων. Σε ό,τι αφορά τη μεταφορά και όταν πρόκειται για τη μη μεταφορά, πρέπει να είμαστε προσεκτικοί.
- Τρόπος μεταφοράς : Μόλις απαντηθούν οι ερωτήσεις για το τι και πότε, μπορούμε να προχωρήσουμε στον τρόπο μεταφοράς γνώσης μεταξύ τωμάτων/εργασιών. Αυτό συνεπάγεται την τροποποίηση των τρεχόντων αλγορίθμων και τη χρήση άλλων στρατηγικών, τις οποίες αναλύονται με περισσότερη λεπτομέρεια σε επόμενες ενότητες.[19]

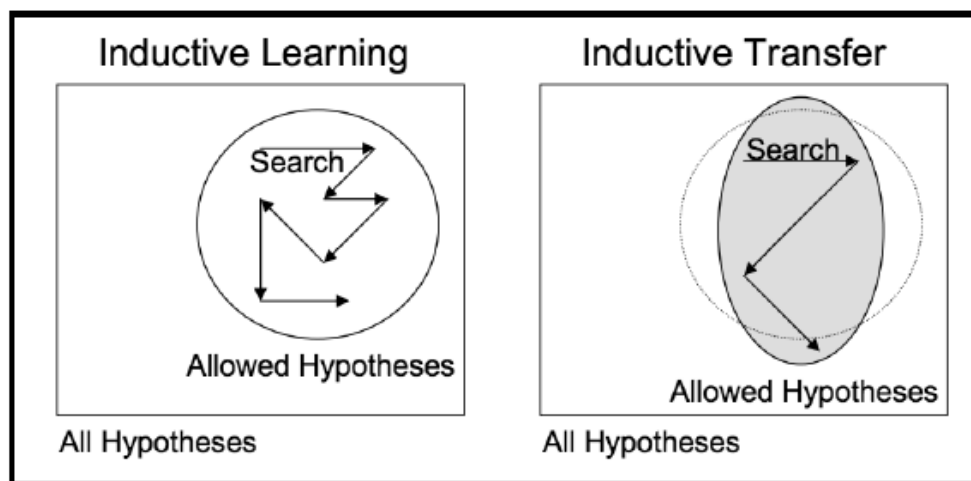
2.5.1 Τεχνικές **Transfer Learning**

Όπως φαίνεται στην προηγούμενη εικόνα, οι μέθοδοι εκμάθησης μεταφοράς μπορούν να ταξινομηθούν ανάλογα με τον τύπο των κλασικών αλγορίθμων μηχανικής μάθησης που χρησιμοποιούνται. Αυτοί οι τύποι είναι γενικοί και μπορούν να εφαρμοστούν σε τεχνικές μηχανικής μάθησης. Αυτοί οι τύποι είναι οι παρακάτω:

- **Inductive Transfer learning** : Παρόλο που οι τομείς προέλευσης και στόχος είναι οι ίδιοι σε αυτήν την περίπτωση, οι εργασίες προέλευσης και στόχος δεν είναι. Οι αλγόριθμοι προσπαθούν να βελτιώσουν την εργασία-στόχο χρησιμοποιώντας τις επαγωγικές προκαταλήψεις του τομέα προέλευσης. Αυτό μπορεί να χωριστεί περαιτέρω σε δύο ομάδες, συγκρίσιμες με την εκμάθηση πολλαπλών εργασιών και την αυτοδιδασκόμενη μάθηση, ανάλογα με το αν ο τομέας προέλευσης περιέχει δεδομένα με ετικέτα ή όχι.
- **Unsupervised Transfer Learning** Η εκμάθηση χωρίς επίβλεψη μεταφοράς είναι παρόμοια με την επαγωγική μεταφορά, αλλά εστιάζει σε εργασίες χωρίς επίβλεψη στον τομέα προορισμού. Τα καθήκοντα είναι διαφορετικά, αλλά οι τομείς προέλευσης και προορισμού είναι παρόμοιοι. Τα δεδομένα με ετικέτα δεν είναι διαθέσιμα σε κανέναν τομέα σε αυτήν την περίπτωση.
- **Transductive Transfer Learning** : Οι εργασίες πηγής και προορισμού είναι συγκρίσιμες σε αυτό το σενάριο, αλλά οι αντίστοιχοι τομείς είναι διακριτοί. Ο τομέας προέλευσης έχει πολλά δεδομένα με ετικέτα σε αυτήν την περίπτωση, αλλά ο τομέας προορισμού δεν έχει κανένα. Αυτό μπορεί να χωριστεί περαιτέρω σε υποκατηγορίες, οι οποίες αναφέρονται σε καταστάσεις στις οποίες οι χώροι χαρακτηριστικών ή οι οριακές πιθανότητες είναι διαφορετικοί.[19]

Inductive transfer techniques Οι τεχνικές επαγωγικής μεταφοράς χρησιμοποιούν τις επαγωγικές της εργασίας πηγής για να βοηθήσουν την τελική εργασία. Αυτό μπορεί να γίνει με διάφορους τρόπους, όπως προσαρμόζοντας την επαγωγική της τελικής εργασίας, περιορίζοντας το χώρο του μοντέλου, το χώρο της υπόθεσης ή κάνοντας προσαρμογές στην ίδια τη διαδικασία αναζήτησης με τη βοήθεια της γνώσης από την εργασία προέλευσης. Αυτή η διαδικασία απεικονίζεται οπτικά στο παρακάτω σχήμα.

Εκτός από την επαγωγική μεταφορά, οι αλγόριθμοι επαγωγικής μάθησης χρη-



Σχήμα 14: Inductive Learning, Inductive Transfer

σιμοποιούν επίσης τεχνικές **Bayesian** και Ιεραρχικής μεταφοράς για να βοηθήσουν στη βελτίωση της εκμάθησης και της απόδοσης της καταλκτικής εργασίας.[19]

DistilBERT Το DistilBERT είναι το πιο γνωστό και ευραίως διαδομένο μοντέλο για **sentiment analysis**. Είναι προεξπαιδευμένο μοντέλο βασισμένο στο BERT, είναι 40% μικρότερο, 60% ταχύτερο, διατηρώντας τις γλωσσικές του δυνατότητες κατά 97%. Η ύπαρξη του δείχνει πώς ένα μοντέλο με λιγότερα δεδομένα, μπορεί να έχει τις ίδιες δυνατότητες με ένα μεγαλύτερο, και παράλληλα να διατηρεί την υψηλή του ταχύτητα. Αυτό το μοντέλο έχει εκπαιδευτεί για να συμπεριφέρεται όπως το μοντέλο δάσκαλος, το BERT, οπότε χρειάστηκαν λιγότερα χρήματα και χρόνος για να εκπαιδευτεί. Το DistilBERT μπορεί να χαρακτηριστεί ως μαθητής, καθώς βασίζεται πάνω σε άλλο μοντέλο το BERT, το οποίο με τη σειρά του, ακολουθεί τον όρο του καθοδηγητή, τα στρώματα **embeddings** και **pooler** δεν συμπεριλαμβάνονται σε αυτό. Το μοντέλο αυτό δεν χρειάζεται σύστημα με υψηλά στανταρ, αφού θα μπορούσε να τρέξει και σε μία κινητή συσκευή. Το Distil στο όνομα του προέρχεται από μία βασική λειτουργία του για να μειώσει το βάρος του μοντέλου που ονομάζεται **Knowledge distillation**, μετάφραση αυτού θα μπορούσε να αποτελέσει το απόσταξη γνώσης. Είναι μία διαδικασία συμπίεσης, ώστε να εκπαιδευτεί ένα μοντέλο, με στόχο να προσομοιώσει τη συμπεριφορά ενός μεγαλύτερου, όπως το BERT.

Άλλα παρόμοια μοντέλα : Άλλα παρομοια και εξίσου δημοφιλοί συστήματα είναι τα εξής :

- **Twitter-roberta-base-sentiment** : διατηρείται από το facebook, χρησιμοποιεί το μοντέλο roBERTa το οποίο είναι επίσης βασισμένο στο BERT, έχει εκπαιδευτεί με 58 εκατομμύρια tweets, σε αντίθεση με το DistilBERT, αυτό χρησιμοποιεί την τεχνική φινε-τυνινγκ, στην οποία φορτώνεται ο αλγόριθμος με συγκεκριμένα δεδομένα, με σκοπό να συμπεριφερθεί όπως ο DistilBERT
- **Bert-base-multilingual-uncased-sentiment** : αυτό το μοντέλο λειτουργεί όπως το πάνω με fine-tuning, υποστηρίζοντας όμως περισσότερες γλώσσες, όπως Γερμανικά, Ολλανδικά, Γαλλικά, Ισπανικά και Ιταλικά.
- **Distilbert-base-uncased-emotion** : Τρίτο και αρκετά ενδιαφέρον, είναι ένα σύστημα που επίσης χρησιμοποιεί το fine-tuning, άλλα είναι σχεδιασμένο για να καταλαβαίνει πιο συγκεκριμένα συναισθήματα, όπως θυμός, λύπη, χαρά, αγάπη κ.α.

[20]

2.6 Σύγκριση με παρόμοια προγράμματα

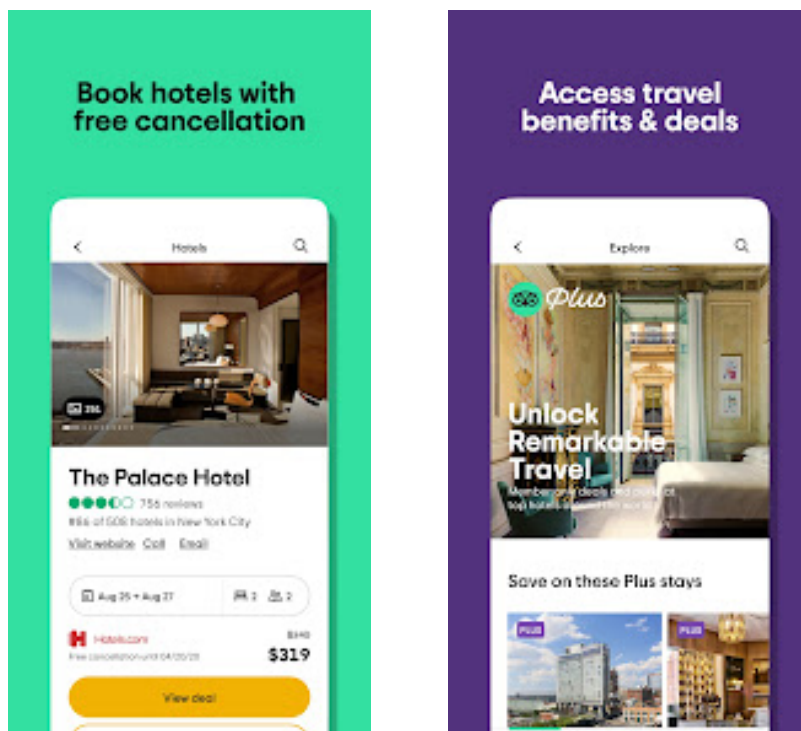
Σαφώς υπάρχουν άλλες εφαρμογές για ταξιδιωτικούς προορισμούς που παρέχουν δελεαστικές υπηρεσίες που βοηθούν τους ταξιδιώτες να κάνουν έχουν μία πιο οργανωμένη και προγραμματισμένη εμπειρία, αλλά και στο sentiment analysis έχει γίνει μεγάλη έρευνα, καθώς υπάρχουν ποικίλοι τρόποι να δοκιμάσει κανείς να αναλύσει το συναισθήμα του συγγραφέα σε ένα κείμενο. Σε αυτήν την ενότητα παρουσιάζονται 2 κορυφαίες εφαρμογές για ταξιδιωτικούς προορισμούς αλλά και 2 επιστημονικά άρθρα σχετικά με την ανάλυση συναισθήματος. Τέλος ακολουθεί ένα κείμενο παραθέτει κάποια επιχειρήματα, στο γιατί η ανάλυση συναισθήματος αξίζει να προστεθεί σε μία εφαρμογή για ταξιδιωτικούς προορισμούς.

2.6.1 Άλλες ανάλογες εφαρμογές (**Tripadvisor**, **Foursquare**)

TripAdvisor Το Tripadvisor είναι ταξιδιωτική ιστοσελίδα με περιεχόμενο που δημιουργείται από χρήστες και ένα site σύγκρισης τιμών. Προσφέρει υπηρεσίες online κρατήσεων ξενοδοχείων και κρατήσεις για μετακίνηση, διαμονή, ταξιδιωτικές εμπειρίες και εστιατόρια.

Για την αναζήτηση ξενοδοχείου στο TripAdvisor, στο κύριο μενού αναζήτησης και επιλέγουμε 'Ξενοδοχεία' και ακολουθούμε τα παρακάτω βήματα.[21]

- Επιλέγουμε τοποθεσία.



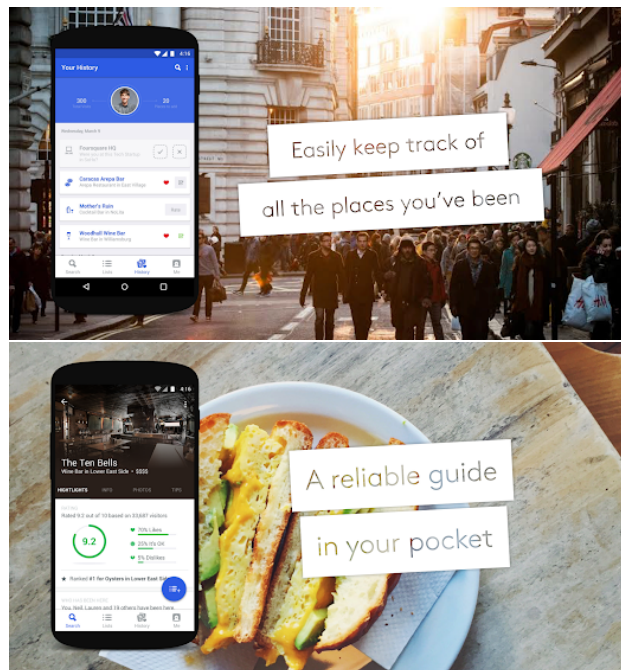
Σχήμα 15: TripAdvisor Screenshots

- Εισαγάγουμε τις ημερομηνίες του ταξιδιού και τον αριθμό των ατόμων που ταξιδεύουν για να μάθουν για συγκεκριμένες τιμές ξενοδοχείων σε αυτήν την τοποθεσία. Δεν χρειάζονται πληροφορίες εάν μας ενδιαφέρουν μόνο οι κριτικές και οι φωτογραφίες.
- Οι επιλογές φιλτραρίσματος βρίσκονται στην αριστερή πλευρά της οθόνης.
- Αυτά περιλαμβάνουν έναν χάρτη που επιτρέπει να αναζήτηση ανά τοποθεσία.
- Κάθε αποτέλεσμα αναζήτησης παρέχει πληροφορίες για το ξενοδοχείο καθώς και για το κόστος.
- Σημαντικό είναι ότι η πιο ακριβή τιμή που εμφανίζεται δεν είναι πάντα η καλύτερη τιμή.

Foursquare Το Foursquare, είναι μια εφαρμογή καθοδήγησης που παρέχει εξατομικευμένες προτάσεις για μέρη κοντά στην τρέχουσα τοποθεσία, το ιστορικό περιήγησης ή/και check-in του χρήστη. [22]

- Θετικά

- Αποφύγη παραπλανητικής περιγραφής. Με τις κριτικές και τις συμβουλές του **Foursquare**, μπορούμε να μάθουμε όλα όσα πρέπει να γνωρίζουμε για μια επιχείρηση ή μια περιοχή πριν πάμε. Αφού γνήσια άτομα βρέθηκαν εκεί, το αξιολόγησαν και είπαν τι ακριβώς έλαβαν.
 - Μαθαίνει τις προτιμήσεις καθώς χρησιμοποιείται - Όσο περισσότερο χρησιμοποιείται το **Foursquare**, τόσο πιο ακριβείς θα είναι οι συστάσεις του. Με άλλα λόγια, βοηθά να στην ανακαλύψω με νέα μέρη για επίσκεψη με βάση τις περιοχές που επισκεπτόμαστε! Αυτό είναι ιδιαίτερα ωφέλιμο όταν επισκεπτόμαστε μια νέα πόλη, καθώς μπορεί να προτείνει τοποθεσίες για επίσκεψη βάση προτιμήσεων.
- Αρνητικά
 - Το **Foursquare** δεν έχει τόσες κριτικές όπως άλλοι ιστότοποι. Αν και είναι εξαιρετικό για τον εντοπισμό μαγαζιών και για χώρους διασκέδασης, δεν έχει τόσους πολλούς χρήστες όσο έχουν γνωστοί ιστότοποι όπως το **TripAdvisor**. Δεν υπάρχουν τόσες πολλές επιχειρήσεις στη συλλογή τους και (κατά συνέπεια) δεν υπάρχουν τόσες πολλές αξιολογήσεις.
 - Τα ανεπιθύμητα μηνύματα είναι κοινά. Μερικοί χρήστες χρησιμοποιούν τις συμβουλές και την ενότητα κριτικών του ιστότοπου για να προωθήσουν τις δικές τους επιχειρήσεις, ενώ αυτό είναι απίθανο να σχετίζεται με αυτό την αναζήτηση που κάνουμε. Επίσης, οι ιδιοκτήτες ενδέχεται να δημιουργήσουν ψευδείς αξιολογήσεις για την επιχείρησή τους, να αλλοιώνεται η αντικειμενικότητα των κριτικών.



Σχήμα 16: FourSquare Screenshots

2.6.2 Σύγκριση άλλων μεθόδων **Sentiment Analysis**

Σύγκριση άλλων αλγορίθμων για Sentiment Analysis					
Συγγραφείς	Μέθοδος	Μοντέλο	Πηγή Δεδο- μένων	Μέγεθος Δεδο- μένων	Ακρίβεια
Χατζιβασιλόγλου & McKeown (1997)	Κατηγοριοποίηση αντικειμένων	Μη- ιεραρχική κατηγοριο- ποίηση	Σχόλια χρηστών του WSJ corpus	657(+) (Αντικε- ίμενα)	82%
Turney (2002)	Κατηγοριοποίηση κειμένου	PMI-IR	Σχόλια χρηστών από ταινίες, τράπεζες, ταξ. προορισμούς κ.α.	700(-) 240(+)	65.8-84

Χατζιβασιλόγλου & McKeown Στη συγκεκριμένη δημοσίευση ο αλγόριθμος που προτείνεται βρίσκει αντικείμενα που έχουν είτε θετικό είτε αρνητικό συναίσθημα μαζί με οτιδήποτε μπορεί να αποδόσει συναίσθημα. Με ένα **log-linear regression** μοντέλο και τη βοήθεια αυτών των επιθέτων επιστρέφει ένα βαθμό που αντιπροσωπεύει το πόσο θετικά είναι τα συναισθήματα. Στη συνέχεια ένας

άλλος αλγόριθμος χωρίζει τα επίθετα στις 2 κατηγορίες θετικά/αρνητικά, και τέλος οι μέσες συχνότητες κάθε ομάδας συγκρίνονται με την θετική ομάδα με τη μεγαλύτερη συχνότητα, η πρόβλεψη έχει ακρίβεια 82%.

Turney Στη δημοσίευση του κ. Turney χρησιμοποιείται ένας αλγόριθμος μηχανικής μάθησης χωρίς επίβλεψη, όπου η ταξινόμηση γίνεται με βάση το συναισθηματικό προσανατολισμό φράσεων, βάση των επιθέτων ή των επιρρημάτων. Συγκεκριμένα, κάθε λέξη παίρνει ένα βαθμό απ το ένα ως το δέκα, και συγκρίνεται με τις λέξεις εξαιρετικό και φτωχό, αν το αποτέλεσμα βγεί θετικό τότε η αξιολόγηση είναι θετική. Ο αλγόριθμος πετυχαίνει, ανάλογα με τα δεδομένα, ακρίβεια 74% με 84%.

2.6.3 Γιατι ένας χρήστης να χρειάζεται **Sentiment Analysis**

Είναι ένας ενδιαφέρον και αντικειμενικός τρόπος να βαθμολογεί κάποιος, εκφράζοντας απλά τις εμπειρίες του με λόγια, ένα σύστημα μπορεί να βάλει μία αντικειμενική βαθμολογία βάση τα λεγόμενά του. Λόγο συγκυριών μπορεί η κρίση κάποιου χρήστη να θολωθεί, μην έχοντας αντικειμενικό γνώμονα για την κριτική που πάει να δημοσιεύσει. Επίσης, πιθανό είναι κάποιοι κριτές, να μην μπορούν να αποτυπώσουν αντικειμενικά την άποψη τους σε μία κλίμακα βαθμολογίας. Το **sentiment analysis** έρχεται να λύσει τις όποιες τυχόν παρεκτροπές δίνοντας στη κοινότητα μία αντικειμενική βαθμολογία σε μέρη ψυχαγωγίας και αναψυχής με τη βοήθεια μοντέλων μηχανικής μάθησης.

Παραδείγματα χρηστών Οι ίδιοι χρήστες του **TripAdvisor** και του **FourSquare** θα μπορούσαν να μεταφερθούν και σε αυτή την εφαρμογή, σε ένα υποθετικό συνάριο που η εφαρμογή θα ήταν εμποροποιημένη. Θα παρείχε δηλαδή εξίσου αξιόπιστες υπηρεσίες. Η ανάλυση συναισθήματος ωστόσο προσδίδει στην εφαρμογή ένα πιο έντονο τόνο εμπιστευτικότητας κάνοντας ανθρώπους που ενδιαφέρονται περισσότερο ή αποκλειστικά για τα σχόλια χρωστών, κάποια σενάρια ανθρώπων που θα χρησιμοποιούσαν την εφαρμογή είναι τα εξής :

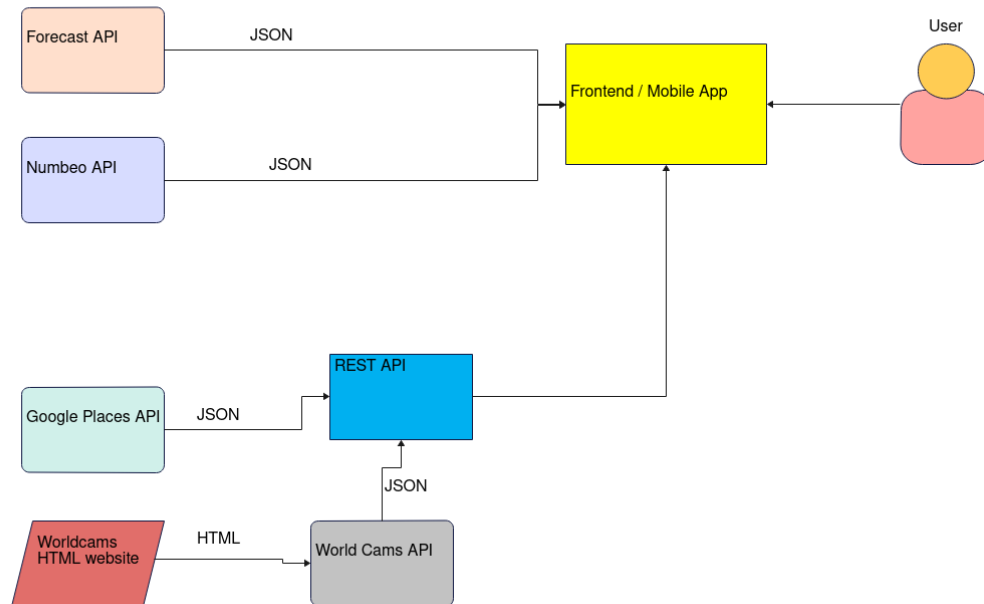
1. Ταξιδιώτες : Όλες οι λειτουργίες της εφαρμογής είναι φτιαγμένες για να εξυπηρετούν ταξιδιώτες που οργανώνουν το ταξίδι τους, οπότε πιο προφανές σενάριο χρήσης είναι από ταξιδιώτες.
2. Κριτικούς : Κριτικοί εστιατορείων/μπαρ ή για οποιοδήποτε άλλο μέρος θα μπορούσαν να χρησιμοποιήσουν την εφαρμογή για να τους βοηθήσει να ασκήσουν πιο αντικειμενική κριτική. Έχοντας την κριτική του συστήματος μπορούν να επαληθεύσουν αν το σχόλιό τους συνάδει με τη

βαθμολογία που έδωσαν ή με την εμπειρία που είχαν στο κατάστημα.

3. **Ιδιοκτήτες Καταστημάτων :** Ένας πιο άμεσος τρόπος να δουν τι πραγματικά μένει στους πελάτες, αφού φύγουν από την επιχειρησή τους. Ο μέσος πελάτης που θα βαθμολογήσει το κατάστημα σε μία ιστοσελίδα, ίσως να μη βαθμολογεί με αυστηρά κριτήρια, είτε να μην δείχνει από την συμπεριφορά του ότι κάτι τον άφησε ανικανοποίητο. Ένα αντικειμενικό σχόλιο, παρουσιάζοντας μόνο πληροφορίες, μπορεί να βοηθήσει τον καταστηματούχο να καταλάβει πιο άμεσα το πόσο αρεστές είναι οι υπηρεσίες του, χωρίς να χρειάζεται να διαβάσει όλα τα σχόλια των χρηστών.
4. **Δημοσιογράφους :** Ίσως να μην είναι τόσο δημοφιλή, αλλά υπάρχουν μία κατηγορία δημοσιογράφων που δεν ασχολούνται με την πολιτική ή τον αθλητισμό, αλλά με την κουζίνα, τη ψυχαγωγία ή την αισθητική. Αυτή η εφαρμογή ίσως φανεί ενδιαφέρον στο μάτι ενός δημοσιογράφου, αφού θα έχει μία άμεση εικόνα για την εικόνα που έχει η κοινότητα για συγκεκριμένα μέρη και μαγαζιά.
5. **Τηλεπερσόνες στα Μέσα Κοινωνικής Δικτύωσης :** Είναι αρκετά δημοφιλείς στους νέους και λιγότερο στις μεγαλύτερες ηλικίες. Άνθρωποι που με τη ελευθερία που τους προσφέρει το διαδίκτυο έχουν καταφέρει να αποκτήσουν κοινό και να ασκήσουν επιρροή σε ένα ευρύ φάσμα ανθρώπων σε όλα τα μέρη της γης. Είναι συχνό φαινόμενο να δημιουργούνται βίντεο κριτικής που να περιγράφουν την εμπειρία τους σε ένα μέρος/κατάστημα. Επίσης ενδιαφέρον, είναι να επισκέπτονται μέρη που έχουν ήδη βαθμολογηθεί από άλλους χρήστες για να επαληθεύσουν την αξιοπιστία αυτών των βαθμολογιών. Συνήθως για να είναι πιο ελκυστικό το περιεχόμενο τους δεν επισκέπτονται κοινά εστιατόρια, αλλά εστιατόρια που είτε είναι πολύ δημοφιλή είτε καθόλου. Μπορεί στα πολύ δημοφιλή εστιατόρια να έχουν εξαίρετες υπηρεσίες αλλά επίσης δελεαστικό, είναι να δει κανείς τι μπορεί να κάνει τόσους ανθρώπους να δώσουν κακές κριτικές.

3 Υλοποίηση - Τεχνική επισκόπηση

Η εφαρμογή αποτελείται από το **Front-end**, που είναι γραμμένο σε **React Native** και το **Back-end** που είναι γραμμένο σε **NodeJs**, όπου και τα 2 είναι **frameworks** της **JavaScript**. **Framework** είναι ένας όρος που χρησιμοποιούμε όταν θέλουμε να αναφερθούμε σε ένα εργαλείο που είναι χτισμένο πάνω σε ένα άλλο, στην προκειμένη περίπτωση, τα **frameworks Nodejs & React-Native** είναι χτισμένα πάνω στη **JavaScript**. Επίσης σαν λοιπά εργαλεία



Σχήμα 17: Αρχιτεκτονική ολόκληρης της εφαρμογής

χρησιμοποιήθηκαν τα **Expo io, Visual Studio Code (IDE), Git/Github, Heroku**. Παρατηρώντας το σχεδιάγραμμα της αρχιτεκτονικής, καταλαβαίνουμε πως τα δεδομένα

3.1 Mobile Apps & Rest APIs

REST API API είναι το πρόγραμμα που μεσολαβεί ανάμεσα στον χρήστη και τον διακομιστή (server) και είναι υπεύθυνο να επιστρέψει ή να ζητήσει δεδομένα. Το REST υποδηλώνει ότι το συγκεκριμένο API πληρεί κάποιες προϋποθέσεις, όπως : Να υποστηρίζονται HTTP κλήσεις, κάθε κλήση είναι ξεχωριστή / δεν εξαρτάται από τις προηγούμενες, δεν αποθηκεύονται δεδομένα από τα requests, τα δεδομένα είναι αποθηκευμένα στην cache, που σημαίνει ότι η πρόσβαση στα δεδομένα είναι ταχύ και διαγράφονται αυτόματα κ.α. Η cache είναι μνήμη που η προσπέλαση είναι γρήγορη, είναι μικρή σε χωρητικότητα και χρησιμοποιείται συνήθως από browsers και προγράμματα που τα δεδομένα διαγράφονται σε κάθε session(συνεδρία). Υπάρχουν διάφορα είδη APIs, με τα πιο ευρέως διαδεδομένα να είναι τα REST API και GraphQL.

[23][24]

Mobile Apps Ένας σύντομος ορισμός για το τι είναι ένα **mobile app**, είναι : Μία εφαρμογή που κάνει ζομπιλε πάνω σε κινητές συσκευές και συνήθως συνεργάζεται με απομακρυσμένους **servers** και συγκεκριμένα **REST APIs** από τα οποία λαμβάνει πληροφορίες που χρειάζεται ο χρήστης. Τα κυρίαρχα λογισμικά είναι το **iOS** της **Apple** το οποίο τρέχει μόνο σε κινητά της ίδιας της εταιρείας, και το **Android** που παρέχεται απ' τη **Google**. Το **Android** χρησιμοποιείται από πληθώρα κατασκευαστών καθώς είναι δωρεάν η εγκατάσταση του μαζί με τις διάφορες υπηρεσίες της **Google** όπως το **Google Chrome** και το **YouTube** τα οποία είναι υποχρεωμένοι οι κατασκευαστές να έχουν στις συσκευές τους. Η κατασκευή εφαρμογών γίνεται από το εκάστοτε **Software Development Kit** που στην περίπτωση του **iOS** γίνεται μόνο από **Apple** υπολογιστή, καθώς το **emulator** είναι διαθέσιμο μόνο εκεί. **Emulator** η αλλιώς προσομοιωτή ονομάζουμε το πρόγραμμα που τρέχει σε υπολογιστή αλλά προσομοιώνει μία κινητή η **tablet** συσκευή ώστε να μας δώσει μία καθαρότερη εικόνα για το πως θα φαίνεται η εφαρμογή που υλοποιούμε σε άλλο περιβάλλον. Για το **Android** υπάρχουν πολλά **SDK**, το πιο ευραίως χρησιμοποιούμενο είναι το **Android Studio**, το οποίο παρέχεται δωρεάν απ' την **Google** και είναι διαθέσιμο σε **Linux** και σε **Windows**, αλλά και διάφορα άλλα όπως το **Xamarin**, **Expo** με το τελευταίο να είναι **cross-platform** που επίσης χρησιμοποιήθηκε στην εφαρμογή της πτυχιακής.[24]

3.1.1 Κατηγορίες **Mobile** Εφαρμογών

Υπάρχουν διάφορες κατηγορίες κινητών εφαρμογών με κάθε μία να έχει της δικές της ιδιαιτερότητες, κάθε εφαρμογή ανταποκρίνεται και σε διαφορετικές ανάγκες και κάποια από τα κρητήρια των προγραμματιστών μπορεί να είναι : τα τεχνικά χαρακτηριστικά της συσκευής αλλά και του **server**, την αλληλεπίδραση που θέλουνε να έχει με το χρήστη, τα διαθέσιμα χρήματα για την υλοποίηση αλλά και για τη συντήρησή της κ.α.

- **Native Mobile Applications** Είναι εφαρμογές που είναι γραμμένες στη γλώσσα και στο **framework** που προτείνεται απ' τους αντίστοιχους κατασκευαστές του λογισμικού. Είναι πολύ γρήγορες, έχουν άμεση πρόσβαση στα **API** της συσκευής, όμως είναι κοστοβόρες και χρειάζονται διαφορετικοί κώδικες για κάθε πλατφόρμα.
- **Cross-Platform Mobile Applications** Αυτές οι εφαρμογές παρέχουν μεγαλύτερη ευελιξία στον προγραμματιστή, καθώς είναι γραμμένες σε οποιαδήποτε γλώσσα, αλλά γίνονται **compile** στη **native** γλώσσα του λογισμικού. Είναι μία θετική επέκταση της παραπάνω κατηγορίας, καθώς

ο ίδιος κώδικας μπορεί να τρέξει σε πολλαπλές πλατφόρμες, όμως είναι πιο αργό καθώς πρέπει να μεταγλωττίζεται σε διαφορετική γλώσσα.

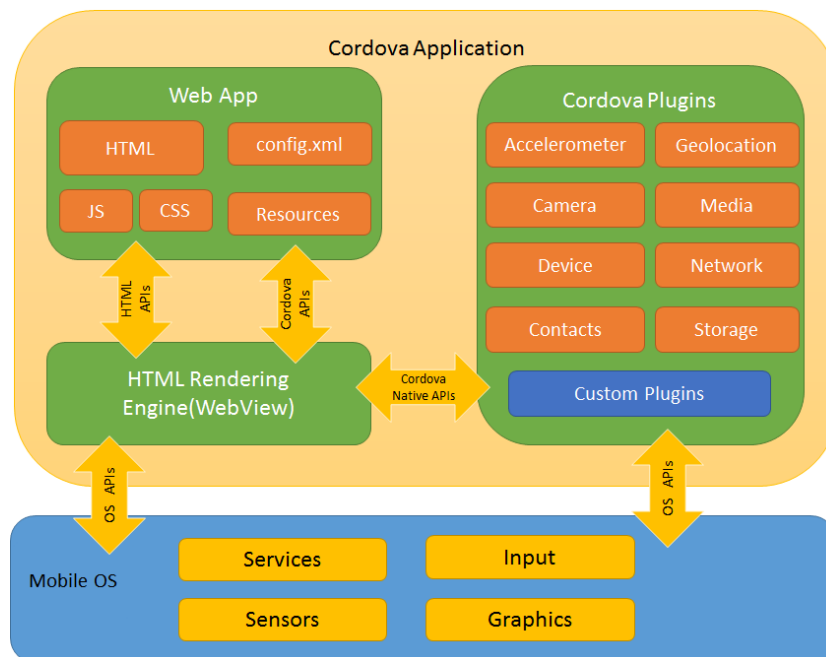
- **Hybrid Mobile Applications** Είναι οι πιο οικονομικές εφαρμογές, καθώς είναι γραμμένες με εργαλεία που χρησιμοποιούνται για τη κατασκευή ιστοσελίδων όπως **Javascript**, **HTML**, **CSS**, και ο **bundler** που κάνει τον κώδικα αυτό να γίνεται **compile** από **smartphone** ονομάζεται **Apache Cordova**. Όπως είναι φανερό, παρέχεται ανακυκλώσιμος κώδικας, καθώς ο κώδικας της ιστοσελίδας μπορεί αν χρησιμοποιηθεί και στην εφαρμογή, με τα αρνητικά να είναι η ταχύτητα και η υποστήριξη που υπάρχει από βιβλιοθήκες του κάθε λογισμικού.
- **Progressive Mobile Applications** Αυτή είναι μία ιδιαίτερη κατηγορία εφαρμογών, καθώς οι εφαρμογές αυτές δεν χρειάζονται εγκατάσταση από κάποιο **store** εφαρμογών. Γίνονται εγκατάσταση και **compile** μέσω του **browser**, και φυσικά τρέχουν μέσω αυτού. Στην πραγματικότητα, η ιστοσελίδα αποθηκεύεται στο κινητό και δίνει την ψευδαίσθηση εφαρμογής. Παρολλ' αυτά είναι πολύ γρήγορες, εύκολες στην εγκατάσταση, αλλά οι δυνατότητες αυτών των εφαρμογών είναι πολύ περιορισμένες.

[24]

Γιατί επιλέχτηκε αυτός ο τύπος εφαρμογής· Λόγο του ότι ήθελα να δημιουργήσω μία εφαρμογή που να μπορεί να χρησιμοποιηθεί από πληθώρα συσκευών αλλά και να είναι γρήγορη κατά την πλοήγηση, μιάς και οι κλήσεις σε **API** θα ελάττωναν την ταχύτητα, επέλεξα αυτόν τον τύπο εφαρμογής. Σε διαφορετική περίπτωση θα έρπεπε να γραφούν δύο κώδικες, ένας για το **Android**, γραμμένος σε **Kotlin** και **Java**, και ο άλλος για το **iOS** γραμμένος σε **Swift** και **Objective-C**. Φυσικά και αυτές οι εφαρμογές θα έτρεχαν γρηγορότερα, όμως κάτι τέτοιο θα σπαταλούσαν πολύτιμο χρόνο που θα μπορούσε να χρησιμοποιηθεί στην υλοποίηση του κυρίως κομματιού που είναι το **Sentiment Analysis**.

[24]

Πως το **Front-end** συνδέεται με το **Back-end** Ως **front-end** ορίζουμε το γραφικό κομμάτι που αλληλεπιδρά με το χρήστη, περιέχει τα κουμπιά, τις εικόνες, τις λίστες κ.α. Το **front** παίρνει πληροφορίες από το βασκ-ενδ το οποίο συνήθως ανήκει σε κάποιον απομακρυσμένο σερβερ. Δεν είναι σπάνιο φαινόμενο, κάποια εφαρμογή να πληρώνει μία υπηρεσία για πληροφορίες, όπως και στη περίπτωση της συγκεκριμένης πτυχιακής. Για να επικοινωνήσει το **front** με το **back**, θα πρέπει να έχει αρκετές και κρίσιμες πληροφορίες όπως την ηλεκτρονική



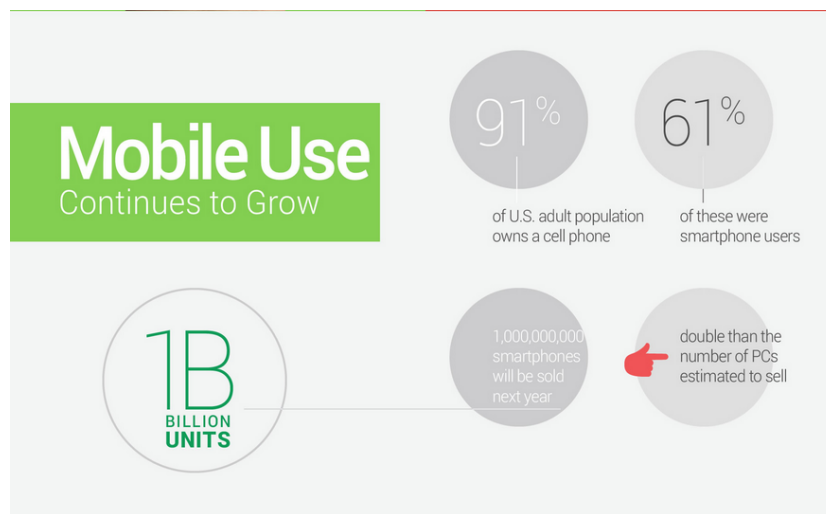
Σχήμα 18: Αρχιτεκτονική του Cordova framework [25]

διεύθυνση του **server**, το πρωτόκολλο αυθεντικοποίησης αλλά και τα ενδποιντ που μπορεί να χτυπήσει ώστε να πάρει τις κατάλληλες πληροφορίες. Ένα παράδειγμα μίας τέτοιας υπηρεσίας είναι το **Google Places API**, θα ήταν μάταιο αν δεν αναφέραμε πως επίσης εφικτό είναι ο ίδιος ο χρήστης να δημιουργήσει το **API** που θα χρειαστεί για να επικοινωνήσει με τον **server**. Το **GraphQL** χρησιμοποιεί **SQL queries** και ένα σχήμα δεδομένων ώστε να μπορεί ο χρήστης να κάνει κλήση στο συγκεκριμένο **API**.

[24]

SQL Είναι μία γλώσσα που χρησιμοποιείται για την επικοινωνία βασκ-ενδ κώδικα με μία βάση. Αυτός ο κώδικας περιλαμβάνει εντολές που δημιουργούν, διαβάζουν, ανανεώνουν και διαγράφουν εγγραφές. Αυτές οι εντολές ονομάζονται εναλλακτικά **CRUD operations**, από τα αρχικά **CREATE**, **READ**, **UPDATE** και **DELETE**.

Cloud Back-end Υπηρεσίες Όταν αναφερόμαστε σε ζλουδ υπηρεσίες, εννοούμε υπηρεσίες διαδικτύου, οι οποίες χρησιμοποιούνται από ένα απομαχρυσμένο **server**, εύκολα προσβάσιμο σε εμάς μέσω διαδικτύου. Υπάρχουν υπηρεσίες που



Σχήμα 19: Έρευνα σχετικά με τον αριθμό **smartphone** το 2017
[26]

προσφέρουν δεδομένα όπως : πρόβλεψη καιρού, τιμές προϊόντων, υλικό από **live** κάμερες που φέρουν ως αντάλλαγμα ένα συμφωνημένο χρηματικό ποσό. Αυτό το ποσό συνήθως κοστολογείται ανάλογα με : την υπηρεσία, το σύνολο των κλήσεων που κάνει η εφαρμογή στην υπηρεσία, το μέγεθος των δεδομένων που αντλείται αλλά και το χρονικό διάστημα που η υπηρεσία θα είναι διαθέσιμη στην εφαρμογή.

[24]

Γιατί κινητή εφαρμογή και όχι ιστοσελίδα: Σύμφωνα με πρωτογενή έρευνα, 91% αμερικανοί ενήλικες χρησιμοποιούν κινητά, εκ των οποίων τα 61% είναι **smartphone**, το 2017 ο αριθμός των **smartphone** που θα πωλούνταν άγγιζε το 1 δισεκατομμύριο, αριθμός διπλάσιος από τον αριθμό υπολογιστών που πωλούνταν εκείνη την περίοδο. Η αναλογία ανθρώπων και **smartphone** αντιστοιχεί στο 1.4, που με άλλα λόγια, κάθε άνθρωπος στον πλανήτη έχει κατά μέσο όρο περισσότερα από 1 **smartphone**. Επίσης γνωρίζοντας και από προσωπικά ότι το κινητό είναι πιο διαδεδομένο από τον υπολογιστή, θεώρησα χρησιμό να γίνει κινητή εφαρμογή με την προοπτική ότι θα δοκιμαστεί αλλά και να χρησιμοποιηθεί από περισσότερους ανθρώπους.

[26]

3.1.2 Mobile Application Testing

Με τον όρο αυτό ορίζουμε το **testing** κινητών εφαρμογών αλλά και συσκευών, το οποίο γίνεται με τη βοήθεια σχετικού λογισμικού. Σε αυτή τη διαδικασία εξετάζεται, όσον αφορά την εφαρμογή : η ανταπόκρισή της, η ποιότητα των λειτουργιών και των υπηρεσιών, καθώς και η απόδοσή της. Επίσης εξίσου σημαντικά είναι και η ασφάλεια, η διαλειτουργικότητά και η χρηστικότητα της. Ως διαλειτουργικότητα αναφερόμαστε στο να υπάρχει ίδια ανταπόκριση της εφαρμογής χωρίς να την επηρεάζει το μέγεθος της οθόνης της συσκευής (δηλαδή όλα τα σχήματα που εμφανίζονται στην οθόνη να είναι ανάλογα του μεγέθους της οθόνης), τα τεχνικά χαρακτηριστικά και τη χωρητικότητα της μπαταρίας. Χρηστικότητα της εφαρμογής είναι το πόσο εύκολη είναι η πλοήγηση, πόσο εύκολα μπορεί κάποιος να τη μάθει κ.α. Σχετικές ερωτήσεις που κρίνουν τη χρηστικότητα είναι σε παρακάτω ενότητα (**SUS Evaluation**) που έχει και τις ερωτήσεις ώστε να γίνει μία τέτοια αξιολόγηση και στην προκειμένη εφαρμογή. Οι κατηγορίες **testing** είναι οι εξής :

- **Test** λειτουργικότητας : Πιστοποιούν ότι οι δυνατότητες στο λογισμικό του κινητού είναι λειτουργικές.
- **QoS Test** : Αφορούν τις τεχνικές επίδοσεις του κινητού.
- **Test** διαλειτουργικότητας : Σε αυτό το τεστ δοκιμάζεται η επίδοση της εφαρμογής σε διαφορετικές συσκευές και δίκτυα.
- **Test** χρηστικότητας : Εδώ δοκιμάζονται οι λειτουργίες του κινητού που ανιχνεύουν ανθρώπινη κίνηση. Αυτός είναι ένας παράγοντας που μπορεί να αφορά την εφαρμογή ή/και την συσκευή.
- **Test** ασφάλειας : Αφορά τα πρωτόκολλα ασφαλείας που υποστηρίζει το κινητό όταν στέλνει δεδομένα, καθώς και το σύστημα αυθεντικοποίησης χρήστη.

Τεχνικές **Testing**

- **Emulation-based testing** : Αυτό το τεστ λαμβάνει μέρος σε ένα **Android/iOS emulator**, το οποίο είναι ένα πρόγραμμα που προσομοιώνει μία συσκευή που συνήθως λειτουργεί με ένα απ' τα 2 λογισμικά. Το θετικό είναι ότι είναι χαμηλού κόστους, αφού από 1 υπολογιστή μπορούμε να προσομοιώσουμε πληθώρα συσκευών (πλην **iOS** που γίνεται μόνο σε **mac**), με το μειονέκτημα να υπάρχουν περιορισμοί στα **gestures**. **Gestures** ονομάζουμε τρόπους που ένας άνθρωπος επικοινωνεί με το κινητό,

πέρα από το να πατάει κουμπιά, αυτό θα μπορούσε να είναι, να κάνει **zoom** με 2 δάχτυλα, να περιστρέφει τη συσκευή, να χρησιμοποιεί την κάμερα, το **fingerprint** κ.α.

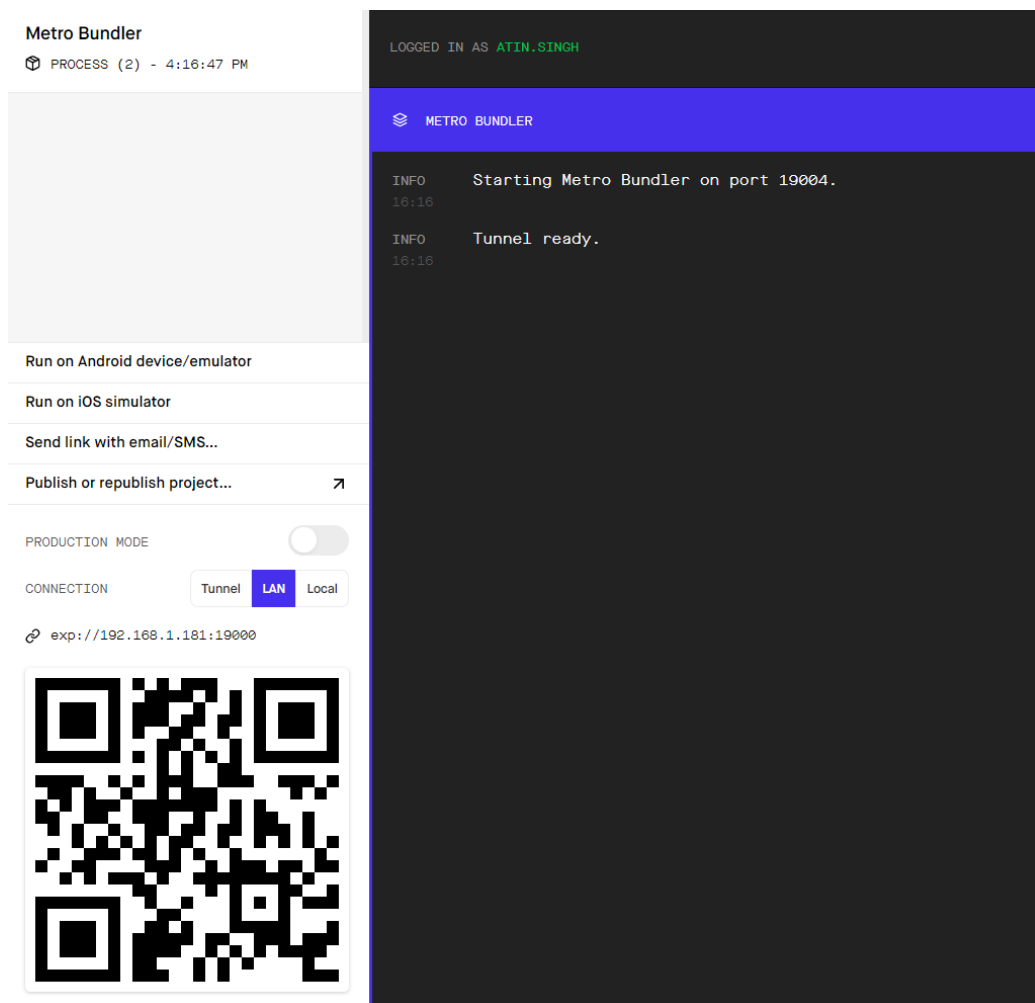
- **Device-based testing** : Σε αντίθεση με την πρώτη περίπτωση, εδώ έχουμε αληθινές συσκευές στις οποίες εξετάζουμε τη συμπεριφορά της εφαρμογής. Έχοντας την δυνατότητα να προσομοιώσουμε στο έπακρο την εμπειρία του χρήστη, μπορούμε να αντιληφθούμε προβλήματα που ίσως υπάρχουν στα **gestures**. Το μειονέκτημα είναι τα χρήματα που χρειάζεται για να καλύφθει μεγάλο εύρος συσκευών, ώστε επαληθευτεί η αξιοπιστία της εφαρμογής όσο το δυνατόν περισσότερο.
- **Cloud testing** : Είναι μία κατηγορία πανομοιότυπη της **Emulation-based**, που αντί να τρέχουμε τον **emulator** τοπικά, υπάρχει διαδικτυακή πλατφόρμα που το τρέχει σε δικό της σύστημα και μας δείχνει τα αποτελέσματα σε πραγματικό χρόνο. Φυσικά κάτι τέτοιο προβλέπει χρέωση, που συνήθως συνάδει με το χρόνο που χρησιμοποιείται.
- **Crowd-based testing** : Ίσως η πιο ενδιαφέρον κατηγορία, στην οποία **testers** ή **test engineers** αναλαμβάνουν τις δοκιμές μέσα από μία διαδικτυακή πλατφόρμα που μπορεί να υποστηρίξει πολλούς χρήστες ταυτόχρονα. Η αξιοπιστία της έγγυται στην κρίση και δυνατότητα των ανθρώπων που την τεστάρουν, αλλά δεν γλυτώνεται το κόστος απ' το να αγοράστούν πραγματικές συσκευές. Εκτός από **testers** που μπορούν να προσληφθούν από την εταιρεία, υπάρχουν και μεσάζοντες υπηρεσίες με **testers** που εξετάζουν την λειτουργικότητα της εφαρμογής για σένα.

[27]

3.2 Τεχνολογίες που χρησιμοποιήθηκαν

Expo io Το **expo** είναι και το πρόγραμμα με το μεγαλύτερο ρόλο στην υλοποίηση της εφαρμογής. Συνοπτικά είναι ένα πρόγραμμα που δημιουργεί αυτόματα όσα αρχεία χρειάζεται για να τρέξει μία εφαρμογή, αφήνοντας το χρήστη να γράψει μόνο τα απολύτως απαραίτητα για την εφαρμογή, που είναι η **React Native**. Μαλιστα παρέχει και άλλες λειτουργίες όπως η προβολή του **application** μέσω **browser** αλλά και σε άλλη συσκευή εντός του δικτύου. Τρέχοντας την εντολή **expo start**, τρέχει ο **server** που ονομάζεται **Metro Bundler**, δημιουργείτε ένα **endpoint localhost:7001**, που από εκεί όλες οι συσκευές του δικτύου μπορούν να δουν την εφαρμογή. Με το παράθεμα **-web** στην παραπάνω εντολή, τρέχει την **mobile** εφαρμογή σε παράθυρο του **browser** διαμορφωμένο

στις διαστάσεις ενός **smartphone**. Όλοι οι **browser** έχουν από **default** αποθηκευμένες διαστάσεις από κάποια τηλέφωνα όπως **iPhone 12**, **Samsung Galaxy S7**, **Google Pixel** κ.α. Επίσης σε περίπτωση που θέλουμε να ανοίξουμε την εφαρμογή στη δική μας φυσική κινητή συσκευή, υπάρχει μία εφαρμογή (στο **PlayStore** και στο αντίστοιχο της **Apple AppStore**) που ονομάζεται **expo** η οποία αυτόματα ανιχνεύει το δίκτυο στο οποίο είμαι συνδεδεμένοι για τον **Metro Bundler**, και μπορούμε να ανοίξουμε την εφαρμογή με ένα κουμπί. Κάτι ξεχωριστό και ιδιαίτερα χρήσιμο, είναι η επιλογή **-web** που ανοίγει το **application** στο **browser**. Σε διαφορετικά προγράμματα όπως το **expo**, χρειάζεται και η εγκατάσταση ενός **Android/iOS emulator** που κάνει την όλη διαδικασία πιο περίπλοκη. Τέλος με την εντολή **expo build**, δημιουργούμε το **.apk** αρχείο που αποτελεί το ολοκληρωμένο **client application** και με το **expo publish** μπορούμε να ανεβάσουμε την εφαρμογή στο **cloud** του **Expo** ώστε να η εφαρμογή να είναι διαθέσιμη για εγκατάσταση και σε άλλους. Διαφορετικά αν θέλουμε να την ανεβάσουμε σε κάποιο **store**, γράφουμε την εντολή **eas submit -p android** ή **iOS** αντίστοιχα.



Visual Studio Code (IDE) Το VsCode αν και δεν έπαιξε σημαντικό ρόλο στην υλοποίηση, βοήθησε στη συγγραφή καθαρού κώδικα, στο **version control** με τη σύνδεση με το **git**, καθώς και το **integrated terminal** ώστε μπορώ να γράφω κώδικα και να τον τεστάρω χωρίς να αλλάζω σελίδα, αλλά και να τρέχω 2 τερματικά ταυτόχρονα.

Git/Github Το git/github σαν **version control** σύστημα, αν και είναι χρήσιμο κυρίως σε ομαδικές εργασίες με βοήθησε να αντιμετωπίσω προβλήματα επιστρέφοντας σε παλαιότερα **commits** κάθε φορά που αντιμετώπιζα κάποιο πρόβλημα.

Heroku Το Heroku είναι μία **cloud computing** εφαρμογή στην οποία έγινε **deploy** ο **server**. Για να μη χρειάζεται να τρέχει στο ίδιο σύστημα **server**

και **client**, έγινε **deploy** στο **Heroku** ώστε να είναι προσβάσιμος από κάθε εφαρμογή ανά πάσα στιγμή.



3.2.1 **Back-end & Front-end** Εφαρμογής

Το **Back-end** της εφαρμογής, αποτελεί τον δίαυλο επικοινωνίας της εφαρμογής του χρήστη με τα δεδομένα. Στον **server** υπάρχουν κατάλληλα διαμορφωμένα **ports** ώστε να φέρνουν δεδομένα όπως το υλικό απ' τις κάμερες, και τα σχόλια απ' τα μαγαζιά. Το **site** με τις κάμερες δεν προσφέρει κάποιο **API**, οπότε τις έκανα μόνος μου **scraping** δημιουργώντας ένα δικό μου **API**. Για τα σχόλια από τα μαγαζιά, αν και δεν χρειάστηκε **scraping**, χρειάζεται μία αλυσιδωτή κλήση σε **APIs** για να λάβει κάποιος τα δεδομένα του. Πιο συγκεκριμένα η πόλη που είναι σε αλφαριθμητική μορφή μετατρέπεται σε συντεταγμένες, από τις συντεταγμένες παίρνουμε τα μαγαζιά σε ακτίνα 30χλμ, και από τα μαγαζιά τα 5 χειρότερα σχόλια με βάση τη βαθμολογία που έβαλε ο κάθε χρήστης στο σχόλιό του.

Το **front-end** της εφαρμογής είναι γραμμένο σε **React-Native**, το συντακτικό της οποίας είναι πολύ απλό, όλα τα ζομπονεντς της εφαρμογής είναι συναρτήσεις οι οποίες βρίσκονται στο φακελο **screens**. Για την αναζήτηση της πόλης χρησιμοποιείται το **autocorrect** απ' το **API** της **Weather API**, όπως φαίνεται και στο αρχείο **resultsCities.js**. Αφού πάρουμε τη σωστά γραμμένη πόλη, κάθε φορά που μπαίνουμε σε μία σελίδα, κάνει μία κλήση στην αντίστοιχη υπηρεσία.



Οι υπόλοιπες υπηρεσίες δεν παρέχονται απ' τον **server** που υλοποίησα άλλα απο άλλα απομακρυσμένα **api**. Αυτά είναι το **Weather API** για το οποίο η μόνη προϋπόθεση για να αποκτήσει κάποιος πρόσβαση, είναι να δημιουργήσει

ένα προσωπικό λογαριασμό στο **site**, έπειτα το κλειδί δίνεται δωρεάν. Το **Numbeo**, είναι το **api** στο οποίο γίνεται κλήση ώστε να φέρει η εφαρμογή κάποιες από τις τιμές που θα συναντήσει κάποιος στην πολή που αναζήτησε, βέβαια υπάρχει και **site** που παρέχει αυτές τις πληροφορίες δωρεάν. Για να λάβει κάποιος πρόσβαση στο **API** θα πρέπει να στείλει ένα **email**, για το πως θα χρησιμοποιηθεί το κλειδί. Το **Google Places API**, έχει ένα δωρεάν όριο στα **requests** και έπειτα από αυτό υπάρχει κοστολόγιο. Για να λάβει όμως κάποιος το κλειδί του θα πρέπει να δημιουργήσει λογαριασμό στη **Google** και να δώσει τα στοιχεία της πιστωτικής ή χρεωστικής του κάρτας.



weather
api



NUMBEO



Google Places API

Express Server Το **express** είναι ένα **library** της **JavaScript** που μας βοηθά να τρέξουμε το **server** μας. Έχει πολύ απλό συντακτικό και με λίγες γραμμές κώδικα μπορούμε να δημιουργήσουμε **endpoints** και **middleware functions** για την όποια λειτουργία θέλουμε να έχει. **Middleware functions** είναι συναρτήσεις που εκτελούνται όταν δέχεται ένα **request**. Συγκεκριμένα η συνάρτηση **app.get** είναι η συνάρτηση που εκτελείται όταν ζητάμε αυτό το **endpoint** και η **middleware function** εκτελείται στο ενδιάμεσο. Το **next()** δηλώνει ότι μετά το πέρας της συνάρτησης ακολουθεί κάποια άλλη.

3.2.2 **APIs** που χρησιμοποιήθηκαν

Υπήρξαν πολλές εναλλακτικές στην επιλογή **API**, υπάρχουν αντίστοιχα **API** με εξαιρετικές υπηρεσίες που δεν χρησιμοποιήθηκαν λόγω κόστους, καθώς ένα από τα κριτήρια που επιλέχθηκαν οι συγκεκριμένες υπηρεσίες ήταν το να είναι δωρεάν. Κάτω από κάθε υπηρεσία υπάρχει και ένα ενδεικτικό **request** με το **endpoint** αλλά και την απάντηση που επιστρέφει. Το **api key** αντιπροσωπεύει

το κλειδί που δίνουμε για να συνδεθούμε στα API ενώ το **country** και **city** την πόλη και την χώρα που βρίσκεται αυτή η πόλη που αναζητούμε.

Weather API Το WeatherAPI.com παρέχει πρόσβαση σε δεδομένα καιρού και γεωγραφικά δεδομένα μέσω ενός **JSON Rest API**. Επιτρέπει τη δημιουργία εφαρμογών χρησιμοποιώντας αυτά τα δεδομένα με μεγάλη ευκολία. Το **days** αντιπροσωπεύει τις μέρες πρόγνωσης που θέλουμε να πάρουμε και το **aqi** δέχεται τις τιμές **yes** ή **no** (που είναι και το **default**) αν θέλουμε στις πληροφορίες να συμπεριληφθεί και η ποιότητα του αέρα.

Αυτά τα δεδομένα είναι:

- Καιρός σε πραγματικό χρόνο.
- Πρόγνωση καιρού 3 ημερών.
- API αναζήτησης ή αυτόματης συμπλήρωσης.
- Στοιχεία ποιότητας αέρα.

Παραδειγμα κλήσης στο **API**

- Request → `https://api.weatherapi.com/v1/forecast.json?key={api key}&q=Thessaloniki& days=10&aqi=yes`

Result →

```
1      {
2        "location": {
3          "name": "Thessaloniki",
4          "region": "Central Macedonia",
5          "country": "Greece",
6          ...
7        },
8        ...
9        "forecast": {
10         "forecastday": [
11           {
12             "date": "2022-08-19",
13             "date_epoch": 1660867200,
14             "day": {
15               "maxtemp_c": 41.1,
```

```

16      ...
17      }
18    }
19  }
20 }

```

[28]

Numbeo API Τα δεδομένα απ' το Numbeo είναι σε μορφή JSON.

- Υποστηρίζει ερωτήματα με βάση την ονομασία διαφορετικής πόλης/χώρας καθώς διαθέτει μια υπηρεσία διασύνδεσης γεω-κανονισμού.
- Υποστηρίζει ερωτήματα κατά γεωγραφικό πλάτος/μήκος
- Πρόσβαση σε καταχωρήσεις ακατέργαστων δεδομένων με χρονικές σημάνσεις

Παραδειγμα κλήσης στο **API**

- Request → https://www.numbeo.com/api/city_prices?api_key={api key}&city=Thessaloniki&country=Greece

Result →

```

1  {
2    "name": "Thessaloniki, Greece",
3    "currency": "EUR",
4    ...
5    "prices": [
6      {
7        "data_points": 34,
8        "item_id": 1,
9        "lowest_price": 6,
10       "average_price": 10.5,
11       "highest_price": 20,
12       "item_name": "Meal, Inexpensive Restaurant,
13         Restaurants"
14     },
15     ...
16   ]
17 }

```

Google Places API Το Places API είναι μια υπηρεσία που παρέχει πληροφορίες σχετικά με χώρους ψυχαγωγίας/εστίασης κ.α. Αν και το συγκεκριμένο API παρέχει ποικιλία πληροφοριών, αυτές που χρησιμοποιούμε είναι ο κατάλογος με τα εστιατόρια από κάθε πόλη, μαζί με κριτικές συνοδευόμενες από μία βαθμολογία. Πιο συγκεκριμένα οι 2 υπηρεσίες που χρησιμοποιούμε είναι τα :

- Το **Place Search** επιστρέφει μια λίστα με μέρη με βάση την τοποθεσία ενός χρήστη ή τη συμβολοσειρά αναζήτησης.
- Το **Place Details** επιστρέφει πιο λεπτομερείς πληροφορίες για ένα συγκεκριμένο μέρος, συμπεριλαμβανομένων των κριτικών χρηστών.

Μπορεί η διαδικασία να φαίνεται απλή, καθώς δίνουμε σαν παράμετρο μία πόλη και παίρνουμε → εστιατόρια και κριτικές. Αλλά ενδιάμεσα ακολουθεί μία αλυσίδα κλήσεων σε API τα οποία αυτοματοποιούν αυτή τη διαδικασία. Κάθε πληροφορία πρέπει να διαχωρίζεται με "%2C", για παράδειγμα, αν θέλαμε στο **findplacefromtext** να πέρναμε σαν αποτέλεσμα εκτός από τις γεωγραφικές συνταταγμένες και την διεύθυνση, έπρεπε να γράψουμε **fields=geometry%2Cformatted_address**, και κάθε μεταβλήτη που του περνάμε με '&' όπως άλλωστε διακρίνουμε και παρακάτω.

Αναλυτικά :

1. Request → [https://maps.googleapis.com/maps/api/place/findplacefromtext/json?input=Thessaloniki&inputtype=textquery&fields=geometry&key=\\${api key}](https://maps.googleapis.com/maps/api/place/findplacefromtext/json?input=Thessaloniki&inputtype=textquery&fields=geometry&key=${api key})

Result →

```

1  {
2    "candidates": [
3      {
4        "geometry": {
5          "location": {
6            "lat": 40.6400629,
7            "lng": 22.9444191
8          },
9          ...

```

Η συγκεκριμένη λειτουργία ονομάζεται **findplacefromtext**, βρίσκει την πόλη δίνοντας ως παράμετρο το όνομα της. Το **json** σημαίνει ότι το θέλουμε το αποτέλεσμα που θα μας επιστρέψει να είναι σε μορφή **JSON**, έχουμε και την επιλογή να λάβουμε και αποτελέσματα σε μορφή **XML**. Απ' το **inputtype** καταλαβαίνει ότι η παράμετρος που λαμβάνει είναι σε μορφή **text**, σε περίπτωση που θέλαμε να ψάξουμε με βάση τον αριθμό τηλεφώνου, μίας και αυτό αποτελεί μία υποστηριζόμενη λειτουργία, θα βάζαμε **phonenummer**. Και στο **fields** γράφουμε τα πεδία που θέλουμε να μας επιστρέψει, στο συγκεκριμένο παράδειγμα θέλουμε να μας επιστρέψει τις γεωγραφικές συντεταγμένες, κάτι που θα μας βοηθήσει μετ' έπειτα.

2. Request → <https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=40.6400629,22.9444191&radius=30000&type=restaurant&key=api key>

Result →

```
1  "results": [  
2    {  
3      "name": "Astoria Hotel Thessaloniki",  
4      "place_id": "ChIJh_rW_Ak5qBQRTw2Jbntcuog",  
5      "rating": 4,  
6      "types": [  
7        "lodging",  
8        "restaurant",  
9        "food",  
10       "point_of_interest",  
11       "establishment"  
12     ]...  
]
```

Το **nearbysearch** μας επιστρέφει τα **ID** από τα εστιατόρια στην περιοχή. Φυσικά όπως και όλα τα **endpoint** μας επιστρέφουν περισσότερα δεδομένα, το μόνο που χρειαζόμαστε εμείς όμως από το συγκεκριμένο, είναι τα **ID** από τα εστιατόρια στην Θεσσαλονίκη, σε ακτίνα 30 χλμ. από το κέντρο (κέντρο θεωρείται η πλατεία Αριστοτέλους). Αναλυτικά το **URL**, στο **location** βάζουμε ως παράμετρο τις συντεταγμένες του μέρους συνοδευόμενες από ένα κόμμα ' , ' ενδιάμεσα. Έπειτα το ραδιος είναι η ακτίνα από αυτό το μέρος που θέλουμε να πάρουμε από το χάρτη, μονάδα μέτρησης είναι το μέτρο, οπότε αυτό μεταφράζεται στα 30000 μέτρα, τέλος

μία λέξη κλειδί για το μέρος που ψάχνουμε, στη συγκεκριμένη περίπτωση βάλαμε **restaurant**.

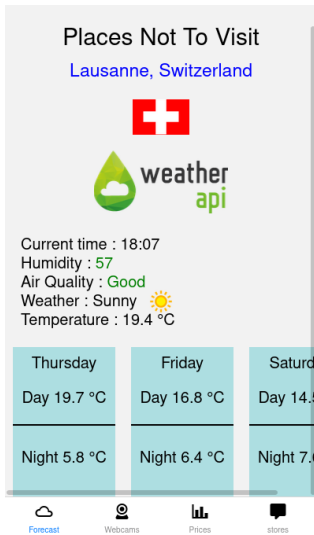
3. Request → https://maps.googleapis.com/maps/api/place/details/json?place_id=ChIJh_rW_Ak5qBQRTw2Jbntcuog&fields=name,reviews&key={api key}

Result →

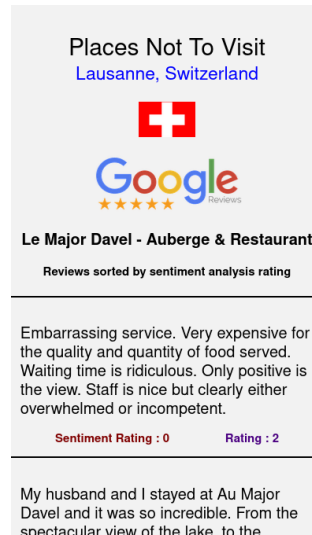
```
1 "result": {
2   "name": "Hotel Thessaloniki",
3   "reviews": [
4     {
5       "rating": 5,
6       "text": "Very nice hotel in the heart of the
              city center.",
7     },
8     {
9       "rating": 4,
10      "text": "The best place to stay in Thessaloniki
              ! Very close to the sea and local
              entertainment area with awesome food!",
11    },
12    ...
```

Τελευταίο και πιο σημαντικό call, γίνεται στο **endpoint** που ονομάζεται **place details**, και μας δίνει πληροφορίες για το κατάστημα, και συγκεκριμένα, σχόλια χρηστών μαζί με βαθμολογία που εκείνοι έβαλαν. Στη συνέχεια αυτή τη βαθμολογία τη συγκρίνουμε με τη βαθμολογία που έβγαλε ο **sentiment analysis** αλγόριθμος για να επαληθεύσουμε την ακρίβειά του.

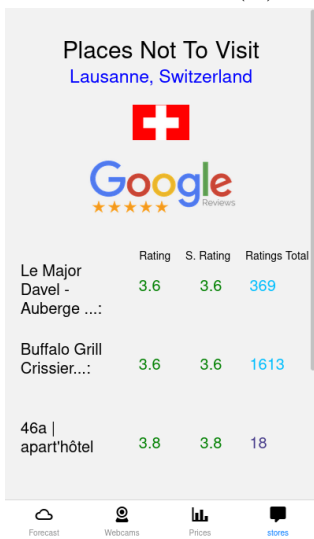
World Cams API (Unofficial) Το World Cams API, δεν είναι επίσημο API της σελίδας World Cams, αλλά ένα ψευδο-API που έφτιαξα εγώ για την υλοποίηση της εφαρμογής. Αυτό το API έχει 2 **endpoints** από όπου δέχεται 2 **GET Requests**, καθώς και ένα κλειδί για λόγους ασφαλείας. Το ένα **endpoint** επιστρέφει τα ονόματα απ τα μέρη απ την κάθε κάμερα καθώς και **thumbnails**. Το άλλο **endpoint** επιστρέφει την πηγή που κάνει **stream** αυτό το live. Για να πάρουμε τη σελίδα **html** χρησιμοποιήθηκε η βιβλιοθήκη **Axios** ενώ για το **scraping** το **Cheerio**.



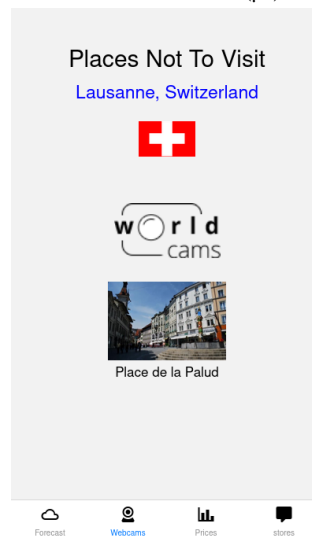
(α') Weather



(β') Reviews



(γ') Stores



(δ') Cameras

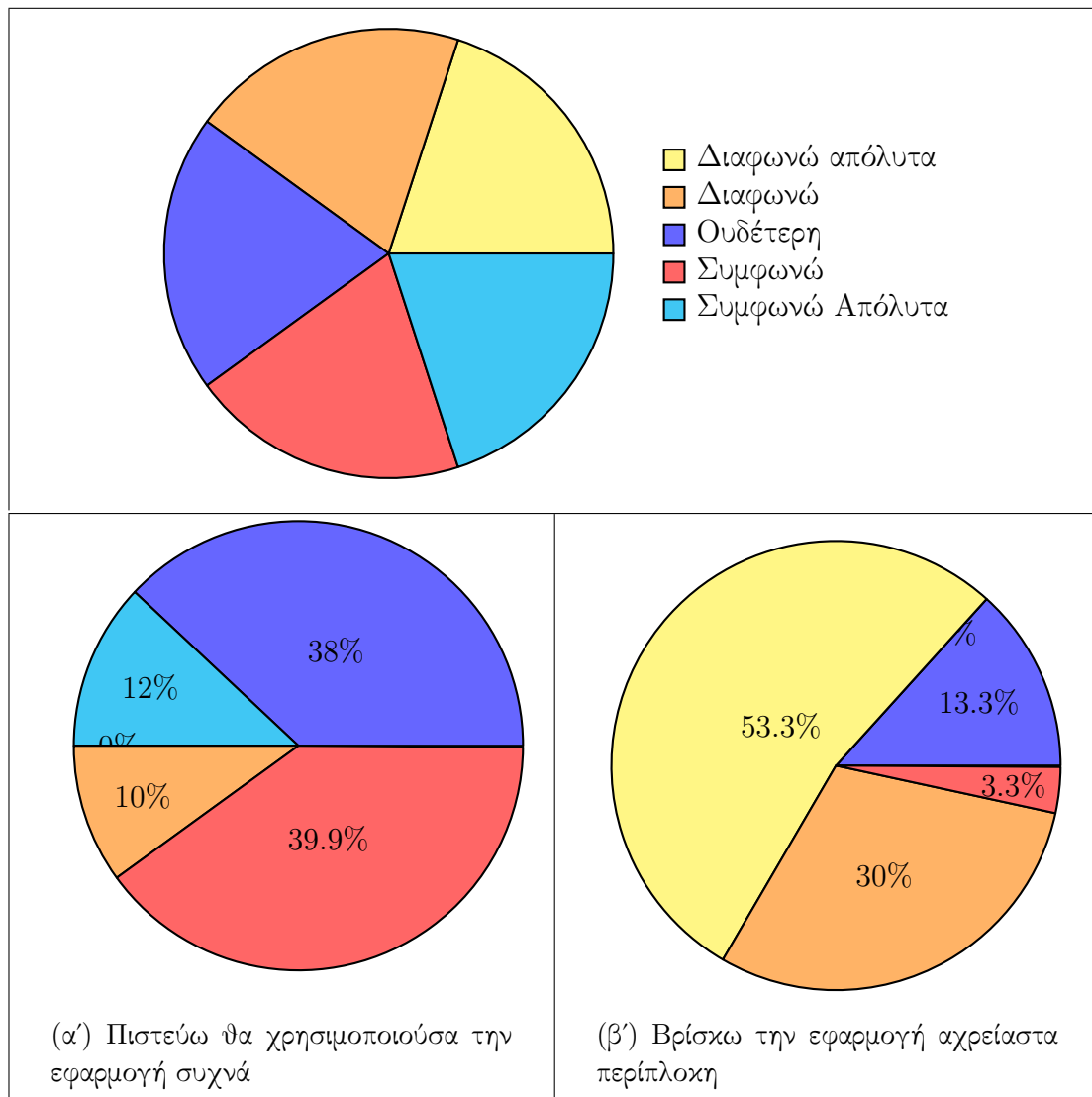
Σχήμα 20: App Screenshots

4 SUS Evaluation

Το **SUS evaluation** περιέχει ένα ερωτηματολόγιο με ερωτήσεις που βασίζονται στη λειτουργικότητα της εφαρμογής. Οι πιθανές επιλογές είναι οι εξής : Διαφωνώ Πολύ, Διαφωνώ, Ουδέτερος, Συμφωνώ, Συμφωνώ Πολύ. Κάτω από τον πίνακα με τις ερωτήσεις ακολουθούν και σχετικά **pie charts** που δείχνουν τι απάντησε η πλειοψηφία. Οι ερωτηθέντες ήταν όλοι ηλικίας από 21-24 ετών, και χρησιμοποιούν κινητές εφαρμογές σε καθημερινή βάση.

SUS Evaluation					
Ερωτήσεις	Διαφωνώ Πολύ	Διαφωνώ	Ουδέτερος	Συμφωνώ	Συμφωνώ Πολύ
Θα χρησιμοποιούσα συχνά το εργαλείο.					
Το βρήκα αχρειάστα δύσχρηστο το εργαλείο					
Πίστευα η εφαρμογή ήταν πιο εύχρηστη					
Πιστεύω χρειάζομαι τη συνδρομή τεχνικού για να χρησιμοποιήσω την εφαρμογή					
Οι διάφορες λειτουργίες της εφαρμογής είναι εφαρμόσιμες σωστά					
Πιστεύω υπάρχει αστάθεια στη χρήση της εφαρμογής					
Πιστεύω οι περισσότεροι μελλοντικοί χρήστες να μάθουν γρήγορα, να χρησιμοποιούν την εφαρμογή					
Βρήκα πολύ αργή την εφαρμογή					
Ένωσα μεγάλη αυτοπεποίθηση χρησιμοποιώντας το εργαλείο					
Χρειάστηκαν προηγούμενες γνώσεις για να χρησιμοποιήσω το εργαλείο					

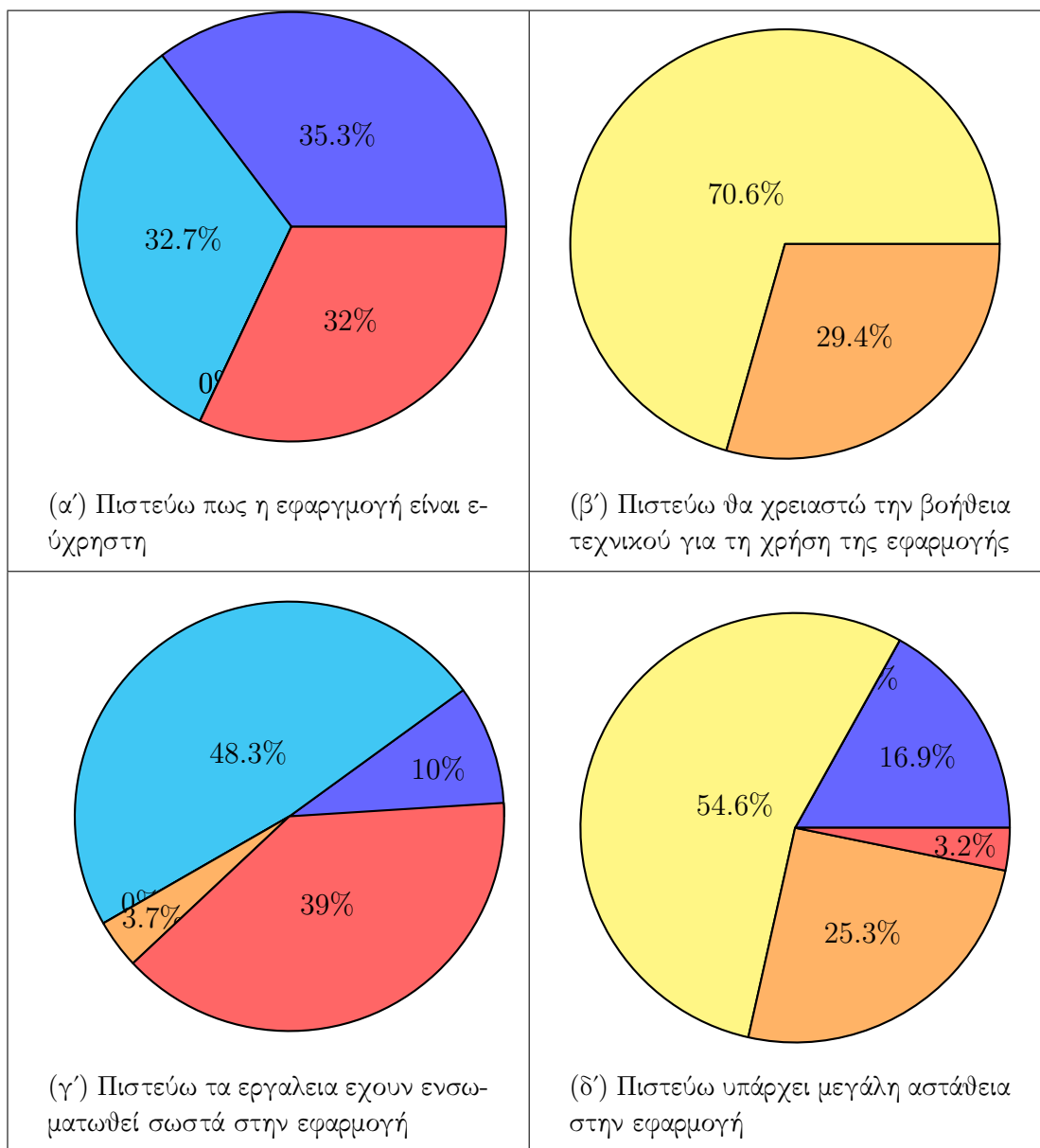
Πίνακας 1: SUS Evaluation



Σχήμα 21: Αποτελέσματα ερωτήσεων του SUS Evaluation

4.1 Σχολιασμός αποτελεσμάτων του **SUS evaluation**

Τα αποτελέσματα ήταν πολύ θετικά καθώς η κοινή γνώμη στρέφεται στο ότι η εφαρμογή είναι προσανατολισμένη στις ανάγκες των χρηστών, αν και δεν έχει ακόμη κερδίσει την εμπιστοσύνη τους. Συγκεκριμένα το 64.7% πιστεύει πως η



Σχήμα 22: Αποτελέσματα ερωτήσεων του SUS Evaluation (2)

εφαρμογή είναι εύχρηστη ενώ υπάρχει ένα αρκετά μεγάλο ποσοστό, 35.3% που είναι ουδέτερο. Εξίσου σημαντική ερώτηση είναι, αν βρίσκουν την εφαρμογή αχρείαστα περίπλοκη, αυτή τη ερώτηση κατάφερε να συγκεντρώσει το 83.3% των ερωτηθέντων να απαντάνε αρνητικά, καθώς ένα μικρό ποσοστό 3.3% συμφωνεί με αυτή τη πρόταση και ένα 13.3% είναι ουδέτερο. Τέλος μπορεί κάποιος να καταλάβει αν αυτές οι πληροφορίες έχουν κατανεμηθεί σωστά στην εφαρμογή

γή, από την ερώτηση 'Πιστεύω τα εργαλεία έχουν ενσωματωθεί σωστά στην εφαρμογή.' καθώς το > 70% απάντησε θετικά, ένα 10% έδειξε ουδετερότητα και ένα αμελητέο ποσοστό 3.7% να απαντά αρνητικά. Βεβαίως αυτά τα ποσοστά αποτελούν δείγμα και όχι ακριβή ποσοστά από απαντήσεις καθημερινών χρηστών, καθώς όλοι οι χρήστες ανήκαν στην ίδια ηλικιακή ομάδα, και είχαν μεγάλη εμπειρία στη χρήση εφαρμογών.

4.2 Αξιολόγηση κριτικών από χρήστες

Στον παρακάτω πίνακα ακολουθούν σχόλια χρηστών μαζί με τις κριτικές, οι αξιολογήσεις των κριτικών είναι στην κλίμακα 1-5. Κάθε κριτική χαρακτηρίζεται αρνητική αν έχει βαθμολογηθεί από 1 μέχρι 3 και θετική αν είναι 4 ή 5. Επίσης ακολουθεί και μία εκτίμηση του αλγορίθμου για το συγκεκριμένο σχόλιο καθώς και η βαθμολογία 2 αμερόληπτων κριτών. Τα σχόλια αυτά έχουν αντληθεί από το **Google Places API** με τη μεθοδολογία που αναφέρθηκε. Οι κριτές αυτοί χρησιμοποιούν κινητές εφαρμογές σε καθημερινή βάση, και ανήκουν στην ηλικιακή κατηγορία 21-25 ετών.

Αξιολόγηση κριτικών ως Θετικά/Ουδέτερα/Αρνητικά				
Σχόλιο	Χρήστης	Αλγόριθμος	Κριτής 1	Κριτής 2
Disappointing service. After being recommended a dish instead of the one I chose, which was unavailable, I found that it was actually for two persons. I had not been informed of this when it was recommended and therefore had no choice but to waste the food as I was not allowed to bring it to my hotel room. When I raised the fact that I'd been recommended a dish for two people but had not been informed of this when I ordered it, the restaurant refused to take any responsibility for this and basically told me I could take the remaining food with me. I was very unhappy about the waste of food and lack of responsibility for the mis/lack of information given me	Αρνητική	Αρνητική	Αρνητική	Αρνητική

Small portions, high prices, and much of the food needed salt or flavor.The sangria didn't taste or look like sangria. Tasted more like juice.But I will say that the food was alright.	Αρνητική	Θετική	Ουδέτερη	Αρνητική
Average cleanness, heating is off during daytime. Friendly staff, super location, value for money hotel	Ουδέτερη	Αρνητική	Θετική	Θετική
Went there, encouraged by a few mentions of vegan food on here and by the menu that's uploaded to Google..the place was completely empty and the lady had 4 trays with leftovers laid out buffet-style. Most tables were littered with dirty crockery and cutlery. I approached her and asked about food, she pointed to the trays and told me what the food was and after I said that I was vegan she said everything was vegan.including the tzatziki which definitely did not look vegan. I escaped	Αρνητική	Αρνητική	Αρνητική	Αρνητική

Πίνακας 2: Αξιολόγηση κριτικών, άνθρωποι vs. αλγόριθμος

5 Συμπεράσματα - μελλοντικές επεκτάσεις

Η εφαρμογή έχει ακόμα πολύ δρόμο μέχρι να φτάσει στο επίπεδο άλλων αντιστοιχών εφαρμογών. Οι επεκτάσεις που αντιλαμβάνομαι ότι πρέπει να γίνουν, είναι κυρίως για πειραματικούς λόγους, καθώς και για να κάνουν την επίδραση που θα έχει ο χρήστης με την εφαρμογή, πιο κοντά με αυτή σε άλλες εφαρμογές. Επίσης παραθέται και μία παράγραφος με κάποια από τα σημαντικότερα προβλήματα που υπήρξαν και το πως αντιμετωπίστηκαν. Τέλος το **SUS evaluation**, συγκεντρώνει βαθμολογία 76.8/100, αυτό εξάγεται με τον εξής τρόπο :

- Διαφωνώ απολύτως $\rightarrow$ 1 πόντος
 - Διαφωνώ $\rightarrow$ 2 πόντοι
 - Ουδέτερος $\rightarrow$ 3 πόντοι
 - Συμφωνώ $\rightarrow$ 4 πόντοι
 - Συμφωνώ Απολύτως $\rightarrow$ 5 πόντοι
1. Προσθέτουμε όλους τους πόντους που επίασαν ερωτήσεις που η σειρά τους μεταφράζεται σε μονό αριθμό (1, 3, 9), έπειτα αφαιρούμε το 5, και πέρνουμε τη τιμή X.
 2. Προσθέτουμε τις ζυγές ερωτήσεις, και αφαιρούμε το σκορ από την τιμή 25, παίρνοντας το Υ.
 3. Προσθέτουμε τα X,Υ και πολλαπλασιάζουμε με το 2.5

Είναι σημαντικό να προστεθεί ότι, ερωτήσεις που αναφέρονται στο πόσο καλή είναι η εφαρμογή, βρίσκονται σε ζυγή θέση (2, 4, 6), ενώ ερωτήσεις που θέλουν να εντοπίσουν τα μειονεκτήματα της εφαρμογής, βρίσκονται σε μονή (1, 3, 9).

5.1 Αλλαγές στο **Mobile App**

- Προσθήκη **loading wheel**, ώστε να αποσπά τη προσοχή του χρήστη κατά τη διάρκεια που φορτώνει τα δεδομένα.
- Προσθήκη **Custom UI Components**, ώστε να γίνει η εφαρμογή πιο φιλική στο χρήστη, αλλά και να αποκτήσει ένα ιδιαίτερο στοιχείο που θα τη διαφοροποιεί από τις υπόλοιπες εφαρμογές.
- Άντληση δεδομένων από περισσότερες πηγές ώστε να παράγεται μία πιο ολοκληρωμένη εικόνα των δεδομένων. Π.χ. Να λαμβάνει δεδομένα θερ-

μοκρασιών από περισσότερα API και να βγάζει ένα μέσο όρο ώστε να είναι πιο ακριβής η πρόβλεψη της θερμοκρασίας.

- Προσθήκη διαφορετικού API ώστε να καλύψει τις ελλείψεις του ήδη υπάρχων. Όπως π.χ. προσθήκη ιστοσελίδων που περιέχουν κάμερες από περισσότερα μέρη. Κατηγοριοποίηση ώστε να είναι πιο διακριτικά τα μέρη που δείχνουν κάμερες, όπως : Αεροδρόμιο, Δημόσιο Αξιοθέατο, Θέρετρο, Κεντρικός Δρόμος κλπ.
- Δημιουργία βάσης χρηστών όπου ο καθένας θα μπορεί να αποθηκεύει κάποιο μέρος και να μοιράζεται μέρη. Να μπορεί επίσης να δημιουργεί λογαριασμό μέσω του Google λογαριασμού του, χωρίς να πρέπει να δημιουργήσει νέο.
- Σελίδα με νέα, που θα εμφανίζει τις κυριότερες / πιο πρόσφατες ειδήσεις που αφορούν την πόλη.
- Αλλαγές στον κώδικα, αντικατάσταση κάποιων auto-generated αρχείων ώστε να βελτιωθεί η ταχύτητα της εφαρμογής

5.2 Αλλαγές στο **Sentiment Analysis**

- Θα ήταν ενδιαφέρον αν δοκίμαζονταν και τα υπόλοιπα μοντέλα και συγκρίνονταν μεταξύ τους.
- Θα μπορούσε να φτιαχτεί ένα καινούριο μοντέλο για πειραματικούς λόγους και για σύγκριση με τα ήδη υπάρχον μοντέλα.
- Δημιουργία ενός μοντέλου για την ελληνική γλώσσα.
- Δοκιμή μεθόδων που δεν περιλαμβάνουν μηχανική μάθηση ή νευρωνικό δίκτυο, και εξέταση της απόδοσης αυτών των λύσεων.

5.3 Προβλήματα που παρουσιάστηκαν κατά την υλοποίηση

Υπήρξαν πολλά προβλήματα κατά την υλοποίηση, πολλά μικρά και εύκολα στην υλοποίηση και κάποια δύσκολα. Λόγο περιορισμού σε χρόνο και σε resources το αποτέλεσμα είναι ερασιτεχνικό και σίγουρα όχι όπως θα κατάφερναν επαγγελματίες που ασχολούνται με την ανάπτυξη mobile apps, τη σχεδίαση αρχιτεκτονικής πληροφοριακού συστήματος ή το Sentiment Analysis ωστόσο έγινε μία μεγάλη προσπάθεια να μπορέσει να βγεί το καλύτερο δυνατό αποτέλεσμα που να συνδυάζει όλα τα παραπάνω.

- **Promises/callbacks** : Είναι ίσως το πιο κομβικό σημείο της εφαρμογής, καθώς η εφαρμογή βασίζεται σε δεδομένα που παίρνει από το διαδίκτυο. Είναι μία απ τις δυσκολότερες έννοιες στον προγραμματισμό καθώς είναι κάτι που συνάντησα τα τελευταία χρόνια. Πρόκειται για κώδικα που δεν εκτελείται σύγχρονα όπως όλοι οι παραδοσιακοί κώδικες, αλλά ασύγχρονα, που σημαίνει ότι, η εκτέλεση του, ήταν κάτι που συνέβαινε ταυτόχρονα με την εκτέλεση του υπόλοιπου κώδικα χωρίς να το καθυστερεί. Συγκεκριμένα αυτό χρησιμοποιείται ώστε να μην χρειάζεται ο χρήστης να περιμένει να φορτώσει όλη η ιστοσελίδα μέχρι να ξεκινήσει να βλέπει το περιεχόμενό της. Αυτό το χρησιμοποιούμε και για μια καλύτερη διάδραση χρήστη-μηχανής, αλλά και για τεχνικούς λόγους, διότι σε διαφορετική περίπτωση ένα κομμάτι της σελίδας που δεν φορτώνει λόγο προβλήματος, θα αχρήστευε όλα τα άλλα δεδομένα που στάλθηκαν με επιτυχία.
- **Αρχιτεκτονική συστήματος** : Είναι ένα πρόβλημα που υποτιμήθηκε ιδιαίτερα για αυτό και αποτέλεσε τόσο σημαντικό πρόβλημα. Ήταν πολύ δύσκολο να φτιαχτεί μία εφαρμογή που να διαχειρίζεται πληροφορίες από τόσες πηγές, χωρίς να υπάρχει ένα σχέδιο για το με ποιο τρόπο θα γίνεται η ροή των δεδομένων. Κάτι τέτοιο δεν μπορούσε να γίνει πριν την υλοποίηση της εφαρμογής, μιας και δεν υπήρχε η κατάλληλη εμπειρία για να προβλεφθούν τέτοιου είδους λάθη αλλά και έλλειψη γνώσεων για το τι μορφή θα πρέπει να έχουν τα δεδομένα. Ωστόσο λόγω του ότι γνώριζα ότι οι υπηρεσίες θα ήταν πολλές φρόντισα να κάνω τα δεδομένα που στέλνονται μέσα στην εφαρμογή, να είναι όσο το δυνατόν λιγότερα και μικρότερα σε μέγεθος.
- **Δημιουργία τεχνητού API** : Μπορεί να είχε τη μικρότερη δυσκολία από τα υπόλοιπα αλλά σίγουρα ήταν η πιο κουραστική διαδικασία από όλες. Ίσως λάθος μου να μπω στη διαδικασία να φτιάξω ένα τεχνητό API χρειάστηκε, χρειάστηκε πάρα πολύ **scraping** ιστοσελίδων και δεδομένων ώστε να υπάρχει ένα θεμιτό αποτέλεσμα. Ευτυχώς όμως υπήρξε επιτυχία και δρα σαν ένα κανονικό API καθώς έχει την κατάλληλη ταχύτητα και είναι διαθέσιμο χωρίς να υπάρξει αυθεντικοποίηση, που σημαίνει ότι είναι και γρήγορο. Συγκεκριμένα το **World Cams API**, λαμβάνει σαν δεδομένο το όνομα της πόλης, και επιστρέφει **links** από το **youtube** που δείχνουν ζωντανό υλικό απ' τις κάμερες. Θεώρησα ότι ήταν κάτι σημαντικό και ενδιαφέρον για μία ταξιδιωτική εφαρμογή, καθώς έδινε μία αίσθηση ρεαλισμού σε κάποιον που ενδιαφέρονταν για ένα συγκεκριμένο μέρος. Το χαρακτήρισα λάθος γιατί χρειάστηκε πολύ χρόνο η υλοποίηση του, αν αυτός ο χρόνος ξοδεύονταν σε κάτι άλλο η εφαρμογή θα είχε λιγότερες

δυνατότητες άλλα ίσως ήταν πιο φιλική προς το χρήστη.

- **Sentiment Analysis** : Είναι τελευταίο σε δυσκολία κυρίως λόγω του ενδιαφέροντος που υπήρχε για το αντικείμενο, αλλά και για το ότι δεν χρειάστηκε πολύς κώδικας ή σκέψη για την αρχιτεκτονική του, μιας και τα μοντέλα και ο κώδικας που χρησιμοποίησα ήταν όλα έτοιμα, και δεν χρειάστηκα προσωπική δουλειά πάνω σε αυτό το κομμάτι, το οποίο θα μπορούσε να είναι το πιο δύσκολο.

6 Κομμάτια Κώδικα

6.1 Client-side

6.1.1 screens.js

```
1 import React from 'react';
2 import { createBottomTabNavigator } from '@react-navigation/bottom-tabs'
3 ;
4 import { MaterialCommunityIcons } from '@expo/vector-icons';
5
6 import Forecast from './forecast';
7 import Webcams from './webcams';
8 import Prices from './prices';
9 import Stores from './stores';
10
11 const Tab = createBottomTabNavigator();
12
13 const Screens = ({ route,navigation }) => {
14   const params = {city : route.params.city, country : route.params.
15     country}
16
17   return (
18     <Tab.Navigator screenOptions={{ headerShown: false }} >
19       <Tab.Screen
20         name="Stores"
21         component={Stores}
22         initialParams={params}
23         options={{
24           tabBarIcon: () => (
25             <MaterialCommunityIcons name="comment" size={24} color="
26               black" />
27           )
28         }}
29       />
30     )
31   )
32 }
```

```

28     />
29     <Tab.Screen
30       name="Forecast"
31       component={Forecast}
32       initialParams={params}
33       options={{
34         tabBarIcon: () => (
35           <MaterialCommunityIcons name="weather-cloudy" size={24}
36             color="black" />
37         ),
38       }}
39     />
40     <Tab.Screen
41       name="Webcams"
42       component={Webcams}
43       initialParams={params}
44       options={{
45         tabBarIcon: () => (
46           <MaterialCommunityIcons name="webcam" size={24} color="black
47             " />
48         ),
49       }}
50     />
51     <Tab.Screen
52       name="Prices"
53       component={Prices}
54       initialParams={params}
55       options={{
56         tabBarIcon: () => (
57           <MaterialCommunityIcons name="chart-bar" size={24} color="
58             black" />
59         ),
60       }}
61     />
62   </Tab.Navigator>
63 );
64 }
65
66 export default Screens;

```

6.1.2 googleReviews.js

```

1 const fs = require('fs');
2 const axios = require('axios');
3 dataStores = require("./dataStores.json")
4 dataReviews = require("./dataReviews.json")
5

```

```

6
7 const cout = (input) => {
8   console.log(input)
9 }
10
11 const sortStores = (dict) => {
12
13   sortedDict = {}
14
15   // Create items array
16   stores = Object.keys(dict).map(key => [key, Object.values(dict[key])
17     [0], Object.values(dict[key])[1], Object.values(dict[key])[2]])
18   keys = Object.keys(Object.values(dict)[0])
19
20   // Sort the array based on the second element
21   stores.sort((first, second) => first[1] - second[1])
22
23   stores.map(el => sortedDict[el[0]] = { [keys[0]]: el[1], [keys[1]]:
24     el[2], [keys[2]]: el[3] })
25
26   return sortedDict
27 }
28
29 const sortReviews = (dict) => {
30
31   sortedDict = {}
32
33   // Create items array
34   reviews = Object.keys(dict).map(key => [key, Object.values(dict[key]
35     [])[0], Object.values(dict[key])[1]])
36   keys = Object.keys(Object.values(dict)[0])
37
38   // Sort the array based on the second element
39   reviews.sort((first, second) => first[1] - second[1])
40
41   reviews.map(el => sortedDict[el[0]] = { [keys[0]]: el[1], [keys[1]]:
42     el[2] })
43
44   return sortedDict
45 }
46
47 writeToJson = (city, stores) => {
48   dataStores[city] = stores
49
50   fs.writeFile("./dataStores.json", JSON.stringify(dataStores), err =>

```

```

    {
49     if (err) {
50         console.error(err)
51         return
52     }
53 })
54 }
55
56 // =====
57 // GET STORES
58
59 const getStores = async (city,api_key) => {
60
61     const type = "restaurant"
62
63     return await axios.get(`https://maps.googleapis.com/maps/api/place/
        findplacefromtext/json?input=${city}&inputtype=textquery&fields=
        geometry&key=${api_key}`)
64         .then(response => {
65
66             coordinates = response.data.candidates[0].geometry.location
67
68             return axios.get(`https://maps.googleapis.com/maps/api/place
                /nearbysearch/json?location=${coordinates.lat + "," +
                coordinates.lng}&radius=30000&type=${type}&key=${api_key}
                `)
69         })
70         .then(response => {
71
72             stores = {}
73             response.data.results.map(el => stores[el.name] = { "rating"
                : el.rating, "ratingsTotal": el.user_ratings_total, "
                place_id": el.place_id })
74
75             stores = sortStores(stores)
76
77             storesTop = Object.keys(stores).filter((store, index) => {
78                 if(index<5){
79                     return stores[store]
80                 }
81             })
82
83             storesTopDict = {}
84             storesTop.map((store,i) => storesTopDict[store] = stores[
                store])
85
86             return {[city] : storesTopDict}

```

```

87     })
88     .catch(err => err)
89 }
90
91 // =====
92 // REVIEWS SENTIMENT ANALYSIS
93
94 const getReviews = async (place_id,api_key) => {
95
96     return await axios(`https://maps.googleapis.com/maps/api/place/
97     details/json?place_id=${place_id}&fields=name,reviews&key=${
98     api_key}`)
99     .then(response => {
100
101         reviews = {}
102
103         response.data.result.reviews.map(el => reviews[el.text] = {
104             "S_rating": parseFloat(sentiment.analyze(el.text).
105             comparative.toFixed(1) * 10), "rating": el.rating, })
106
107         return sortReviews(reviews)
108     })
109     .catch((err) => {
110         return err
111     })
112 }
113
114 module.exports = {getStores, getReviews, cout, writeToJson}

```

6.1.3 resultsCities.js

```

1  import React, { useState, useEffect } from 'react';
2  import { StyleSheet, Text, View, Pressable, ScrollView} from 'react-
3  native';
4  import { MaterialCommunityIcons } from '@expo/vector-icons';
5  import uuid from 'react-native-uuid';
6
7  const ResultsCities = ({navigation,route}) => {
8
9      const [data, setData] = useState([]);
10
11      const api_key = "Forecast key"
12
13      const fetchData = async (url) => {
14          const response = await fetch(url);
15          return response.json();
16      };

```

```

16
17 const getData = () => {
18   try {
19     fetchData(`http://api.weatherapi.com/v1/search.json?key=${api_key}
20               &q=${route.params.place}`).then(data => {
21       setData(data)
22     });
23   } catch (error) {
24     console.error(error);
25   }
26 };
27
28 useEffect(() => {
29   getData();
30 }, []);
31
32 return (
33   <View style={styles.container}>
34     <Text style={styles.titles}>Places Not To Visit</Text>
35     <ScrollView style={{ marginTop : 50}}>
36       <View style={styles.borderline}></View>
37       {data.length > 0 ? data.slice(0,5).map(result =>
38         <View key = {uuid.v4()}>
39           <Pressable style={{alignSelf : 'center'}}
40             onPress = {() => {navigation.navigate('
41               screens',{city : result.name , country :
42               result.country}})}>
43             <Text> </Text>
44             <Text style={styles.result}>{result.name}, {
45               result.country}</Text>
46             <Text> </Text>
47           </Pressable>
48           <View style={{borderWidth: 1}}></View>
49         </View> ) :
50         <Not_found place={route.params.place}/>
51       </ScrollView>
52     </View>
53   )
54 }
55
56 const Not_found = ({place}) => {
57   return(
58     <View style = {{alignSelf: 'center', marginTop : 50}}>
59       <Text style={styles.titles}>'{place}'</Text>
60       <Text style={styles.error_text}>Place not found</Text>
61       <MaterialCommunityIcons style={{ alignSelf : 'center'}} name

```

```

58         ="alert-circle-outline" size={24} color="black" />
59     </View>
60 )
61 }
62 const styles = StyleSheet.create({
63
64     container: {
65         flex: 1,
66     },
67     result: {
68         fontSize: 20
69     },
70     error_text : {
71         fontWeight : 'bold',
72         fontSize : 20,
73     },
74     titles: {
75         marginTop : 100,
76         alignSelf: 'center',
77         fontSize: 30,
78     },
79     borderline:{
80         borderWidth: 1,
81     }
82 });
83
84 export default ResultsCities;

```

6.2 webcamsDisplay.js

```

1  import React, {useState, useEffect} from 'react';
2  import { Text } from 'react-native'
3  import { WebView } from 'react-native-webview';
4
5  const Display = ({route}) => {
6
7      const [link,setLink] = useState({})
8
9      const fetchData = async (url) => {
10         const response = await fetch(url);
11         return response.json();
12     };
13
14     const getData = () => {
15         try {
16             fetchData("https://places-not-to-visit.herokuapp.com/" +

```

```

17         route.params.link).then(data => {
18             setLink(data)
19         });
20     } catch (error) {
21         <Text style={{alignSelf: 'center', fontSize: 30}}>error
22             getting webcams from {route.params.link.split('/')[0]}</
23             Text>
24         console.log(error)
25     }
26 };
27
28 useEffect(() => {
29     getData();
30 }, []);
31
32 return (
33     <WebView
34         style={{flex : 1}}
35         source={ link }
36     />
37 );
38 }
39
40 export default Display;

```

6.3 Server-side)

6.3.1 server.js

```

1  const express = require('express');
2  const wc = require('./worldcams')
3  const gr = require('./googleReviews.js')
4  const cors = require('cors');
5
6  const app = express()
7
8  app.use(cors())
9
10 const wCameras = async (req, res, next) => {
11     res.locals.wCameras = await wc.cameras(req.params.country, req.
12         params.city)
13     next()
14 }
15
16 app.get('/stores/:city/:api_key', async (req, res) => {

```

```

17     city = req.params.city
18     key = req.params.api_key
19
20     stores = await gr.getStores(city, key)
21     res.send(stores)
22 });
23
24 app.get('/reviews/:place_id/:api_key', async (req, res) => {
25
26     place_id = req.params.place_id
27     api_key = req.params.api_key
28
29     const { spawn } = require('child_process');
30
31     reviews = await gr.getReviews(place_id, api_key)
32
33     reviewsArray = Object.keys(reviews)
34
35     reviewsStr = ""
36     reviewsArray.map(review => reviewsStr+= review + "@@")
37
38     const pythonProcess = spawn('python3', ['./SentAnalysis.py',
39         reviewsStr]);
40
41     pythonProcess.stdout.on('data', data => {
42         rating = JSON.parse(data.toString().replace(/\'/g, '\"'));
43
44         reviewsArray.map((review,i) => reviews[review]['sentRating'] =
45             rating[i])
46         res.send(reviews)
47     });
48 });
49 app.get('/cameras/:country/:city/:api_key', wCameras, async (req, res)
50     => {
51     if (api_key = "Webcams key") {
52         res.send(res.locals.wCameras)
53     }
54     else {
55         res.send("error")
56     }
57 });
58 app.get('/cameras/:country/:city/:camera/:api_key', wCameras, async (req
59     , res) => {

```

```

60     if (api_key = "Webcams key") {
61         // link is the website's link for the video
62         // wc.iframe(link) returns an html page containing just the
           video, nothing else
63         link = res.locals.wCameras[req.params.camera][1]
64
65         res.send({ 'uri': await wc.iframe(link) })
66     }
67     else {
68         res.send("error")
69     }
70
71 });
72
73 const PORT = process.env.PORT || 7001
74
75 app.listen(PORT, () => console.log(`server started on localhost:${PORT}
  `));

```

6.3.2 stores.js

```

1  import React, { useState, useEffect } from 'react';
2  import { StyleSheet, Text, View, Image, ScrollView, Pressable } from '
    react-native';
3  import uuid from 'react-native-uuid';
4  import CountryFlag from "react-native-country-flag";
5  import countriesCodes from '../data/countries.json'
6
7
8  const Stores = ({ navigation, route }) => {
9
10     const city = route.params.city
11     const country = route.params.country
12     const api_key = "Google key"
13
14     const [stores, setStores] = useState({});
15
16     const fetchData = async (url) => {
17         const response = await fetch(url);
18         return response.json();
19     };
20
21     const getData = () => {
22         try {
23             fetchData(`https://places-not-to-visit.herokuapp.com/stores/
                ${city}/${api_key}`).then(data => {
24                 setStores(data)

```

```

25         });
26     } catch (error) {
27         <Text>error getting {country},{city}</Text>
28     }
29 };
30
31
32 useEffect(() => {
33     getData();
34 }, []);
35
36 return (
37     <ScrollView>
38         <View >
39             <Text style={styles.placesTitle}>Places Not To Visit</Text>
40             <Text style={styles.cityTitle}>{city}, {country}</Text>
41             <CountryFlag isoCode={countriesCodes[country]} size={45}
42                 style={{ marginTop: "5%", alignSelf: 'center' }} />
43         </View>
44         <View>
45             <Image source={{ uri: 'https://www.revivifymarketing.com
46                 /wp-content/uploads/2020/06/google-reviews-logo.png'
47             }} style={{ marginTop: "5%", alignSelf: 'center',
48                 height: 80, width: 190 }}></Image>
49         </View>
50         <View style={styles.container}>
51             <View style={{ flexDirection: 'row' }}>
52                 <Text style={{ marginLeft: "40%" }}>Rating</Text>
53                 <Text style={{ marginLeft: "5%" }}>S. Rating</Text>
54                 <Text style={{ marginLeft: "5%" }}>Ratings Total</Text>
55             </View>
56             {Object.keys(stores).map(store =>
57                 <View key={uuid.v4()} style={{ flexDirection: 'row'
58                     }}>
59                     <Pressable style={{ width: 120, height: 100 }}
60                         onPress = {() => {navigation.navigate('
61                             resultsReviews',{city : city , country :
62                             country, storeName : store, storeId : stores
63                             [store].place_id)}}}>
64                         <Text style={{ fontSize: 20 }}>{store.
65                             slice(0, 25)}...:</Text> :
66                         <Text style={{ fontSize: 20 }}>{store}</Text>
67                     </Pressable>
68                 </View>
69             )}
70         </View>
71     </ScrollView>
72 )

```

```

59         </Pressable>
60         <View style={{ marginTop: "5%", flexDirection: '
        row' }}>
61             {stores[store]["rating"] < 3.5 ?
62                 <Text style={ [styles.ratingStyle, {
                    color: 'firebrick' }] }>{stores[store
63                     ]["rating"]}</Text> :
64                 stores[store]["rating"] < 4 ?
65                 <Text style={ [styles.ratingStyle, {
                    color: 'green' }] }>{stores[store
66                     ]["rating"]}</Text> :
67                 <Text style={ [styles.ratingStyle, {
                    color: 'limegreen' }] }>{stores[
68                     store]["rating"]}</Text>
69             }
70             {stores[store]["rating"] < 3.5 ?
71                 <Text style={ [styles.ratingSStyle, {
                    color: 'firebrick' }] }>{stores[store
72                     ]["rating"]}</Text> :
73                 stores[store]["rating"] < 4 ?
74                 <Text style={ [styles.ratingSStyle,
                    {color: 'green' }] }>{stores[
75                     store]["rating"]}</Text> :
76                 <Text style={ [styles.ratingSStyle,
                    {color: 'limegreen' }] }>{stores[
77                     store]["rating"]}</Text>
78             }
79             {stores[store]["ratingsTotal"] > 200 ?
80                 <Text style={ [styles.ratingsTotalStyle,
                    {color: 'deepskyblue' }] }>{stores[
81                     store]["ratingsTotal"]}</Text> :
82                 stores[store]["ratingsTotal"] > 100 ?
83                 <Text style={ [styles.ratingsTotalStyle, {color: '
                    dodgerblue' }] }>{stores[store]["
84                     ratingsTotal"]}</Text> :
85                 <Text style={ [styles.ratingsTotalStyle, {color: '
                    darkslateblue' }] }>{stores[store
86                     ]["ratingsTotal"]}</Text>
87             }
88         </View>
89     </View>
90 </ScrollView>
91 );

```

```

85 }
86
87 const styles = StyleSheet.create({
88   container: {
89     marginTop: "10%",
90     marginLeft: "5%",
91   },
92   placesTitle: {
93     alignSelf: 'center',
94     fontSize: 30,
95     marginTop: "10%"
96   },
97   cityTitle : {
98     alignSelf: 'center',
99     fontSize: 23,
100    color: 'blue'
101  },
102  ratingStyle : {
103    fontSize: 20,
104    marginLeft: 30,
105  },
106  ratingSStyle:{
107    fontSize: 20,
108    marginLeft: 50
109  },
110  ratingsTotalStyle : {
111    fontSize: 20,
112    marginLeft: 35
113  },
114  error_text: {
115    fontWeight: 'bold',
116    fontSize: 20,
117  }
118 });
119
120 export default Stores;

```

6.3.3 worldcams.js

```

1  const axios = require('axios')
2  const cheerio = require('cheerio')
3
4  const cameras = async (country,city) => {
5    [country,city] = [country.toLowerCase(),city.toLowerCase()]
6    const city_cameras = await axios(`https://worldcams.tv/${country
7      }/${city}`)
      .then(response => {

```

```

8         const $ = cheerio.load(response.data);
9         const images = $(".cam-promo > a > picture ")
10        const links = $(".cam-promo__title a")
11        const cameras_dict = {}
12
13        images.map(image => cameras_dict[links[image].children
14        [0].data.split(':')[1].replace(/ /,')] = ["https://
15        worldcams.tv" + images[image].children[1].attribs.
16        srcset, "https://worldcams.tv" + links[image].attribs
17        .href])
18
19        return cameras_dict
20    })
21    .catch(() => {
22        return {}
23    })
24    return city_cameras
25 }
26
27 const iframe = async (link) => {
28
29     const camera = await axios(link)
30     .then(response => {
31         const $ = cheerio.load(response.data);
32         const link = $(".iframe")
33
34         return link[0].attribs.src
35     })
36     .catch( () => {
37         return "error : invalid link"
38     })
39
40     return camera
41 }
42
43 module.exports = {cameras, iframe}

```

6.3.4 SentAnalysis.py

```

1 from transformers import pipeline
2 import sys
3
4 reviewsStr = sys.argv[1]
5
6 reviewsArray = reviewsStr.strip('[]').split('@@')[:-1]
7
8 sentiment_pipeline = pipeline("sentiment-analysis")

```

```

9 results = sentiment_pipeline(reviewsArray)
10
11 resultsArray = []
12
13 for i in range(len(results)):
14     resultsArray.append(results[i]['label'])
15
16 print(resultsArray)

```

References

- [1] "Wikipedia picture of ai-ml-dl, avimanyu786svg."
- [2] T. M. Mitchell, "Does machine learning really work?," *AI magazine*, vol. 18, no. 3, pp. 11–11, 1997.
- [3] I. El Naqa and M. J. Murphy, *What Is Machine Learning?*, pp. 3–11. Cham: Springer International Publishing, 2015.
- [4] H. Wang, C. Ma, and L. Zhou, "A brief review of machine learning and its application," in *2009 International Conference on Information Engineering and Computer Science*, pp. 1–4, 2009.
- [5] M. Marcus and H. Minc, *Introduction to Linear Algebra*. Series of advanced mathematics texts, Macmillan, 1965.
- [6] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006.
- [7] D. E. Rumelhart, R. Durbin, R. Golden, and Y. Chauvin, "Backpropagation: The basic theory," *Backpropagation: Theory, architectures and applications*, pp. 1–34, 1995.
- [8] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1253, 2018.
- [9] "Biological neurons."
- [10] S.-C. Wang, *Artificial Neural Network*, pp. 81–100. Boston, MA: Springer US, 2003.
- [11] J. Brownlee, "A gentle introduction to sigmoid function," 2021.
- [12] N. Donges, "A guide to rnn: Understanding recurrent neural networks and lstm networks," 2021.

- [13] R. Prabowo and M. Thelwall, "Sentiment analysis: A combined approach," *Journal of Informetrics*, vol. 3, no. 2, pp. 143–157, 2009.
- [14] H. Kanayama, T. Nasukawa, and H. Watanabe, "Deeper sentiment analysis using machine translation technology," in *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pp. 494–500, 2004.
- [15] T. K. Shivaprasad and J. Shetty, "Sentiment analysis of product reviews: A review," in *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pp. 298–301, 2017.
- [16] "True positives."
- [17] V. Hatzivassiloglou and K. McKeown, "Predicting the semantic orientation of adjectives," in *35th annual meeting of the association for computational linguistics and 8th conference of the european chapter of the association for computational linguistics*, pp. 174–181, 1997.
- [18] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014.
- [19] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [20] "Huggingface, transfer learning sentiment analysis with python."
- [21] "Tripadvisor. review why is it the best travel app today ?," 2020.
- [22] "Foursquare app review," 2022.
- [23] "Redhat. what is a rest-api," 2020.
- [24] "Amazon web services. what is mobile application development ?."
- [25] "Apache cordova documentation."
- [26] "Smartphone use statistics."
- [27] J. Gao, X. Bai, W. Tsai, and T. Uehara, "Mobile application testing: A tutorial," *Computer*, vol. 47, pp. 46–55, feb 2014.
- [28] "Weather api documentation."

[29] “Numbeo api documentation.”

