ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Συστηματων Μεταδοσης Πληροφοριας και Τεχνολογιας Υλικων

Σχεδίαση και Ανάπτυξη Εφαρμογής για την Επεξεργασία Κειμένου σε Κινητές Συσκευές με τη χρήση Αλγορίθμων Μηχανικής Μάθησης

Διπλωματικη Εργασια

της

ΔΑΒΑΡΗ Β. ΕΛΕΝΗΣ

Επιβλέπων: Ιάκωβος Βενιέρης
Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Συστηματων Μεταδοσης Πληροφοριας και Τεχνολογιας Υλικων

Σχεδίαση και Ανάπτυξη Εφαρμογής για την Επεξεργασία Κειμένου σε Κινητές Συσκευές με τη χρήση Αλγορίθμων Μηχανικής Μάθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΔΑΒΑΡΗ Β. ΕΛΕΝΗΣ

Επιβλέπων: Ιάκωβος Βενιέρης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 01 Μαρτίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Ιάκωβος Βενιέρης	Δήμητρα - Θεοδώρα Κακλαμάνη	Αθανάσιος Παναγόπουλος
Καθηγητής Ε.Μ.Π.	Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Συστηματων Μεταδοσης Πληροφοριας και Τεχνολογιας Υλικων

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστική συγγραφέας της παρούσας Διπλωματικής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάσει επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογη έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στη Διπλωματική μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Διπλωματική Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

(Υπογραφή)

.....
Δάβαρη Ελένη

01/03/2022

Περίληψη

Στην παρούσα διπλωματική εργασία παρουσιάζεται ο σχεδιασμός και η υλοποίηση μιας εφαρμογής για κινητές συσκευές Android. Πιο συγκεκριμένα, στην εφαρμογή αυτή γίνεται ενσωμάτωση προεκπαιδευμένων αλγορίθμων Βαθιάς Μηχανικής Μάθησης που εκτελούν εργασίες Επεξεργασίας Φυσικής Γλώσσας (Natural Language Processing). Ουσιαστικά, σχεδιάζεται μια εφαρμογή μέσω της οποίας ο χρήστης έχει την δυνατότητα να συλλέγει κείμενο από οποιαδήποτε σελίδα επιλέξει στο διαδίκτυο, προκειμένου στη συνέχεια να το τροφοδοτήσει ως είσοδο σε έναν αριθμό διαθέσιμων Νευρωνικών Δικτύων μέσα στην εφαρμογή. Κάποια από τα Νευρωνικά Δίκτυα αυτά εκτελούν την συμπερασματολογία τους πάνω στην εργασία της Απάντησης Ερωτήσεων και κάποια στην εργασία της Ταξινόμησης Κειμένων, ανάλογα με την χροιά τους.

Σκοπός της παρούσας διπλωματικής εργασίας είναι τόσο η ανάπτυξη της εφαρμογής που προαναφέρθηκε με την ενσωμάτωση αλγορίθμων Βαθιάς Μηχανικής Μάθησης, οι οποίοι θα εκτελούνται τοπικά στη συσκευή, όσο και η αξιολόγηση των χρόνων εκτέλεσης των αλγορίθμων αυτών. Στις μέρες μας, το να γίνεται τοπική εκτέλεση τέτοιων αλγορίθμων σε κινητές συσκευές είναι κάτι που έχει αναπτύξει γύρω του μεγάλο ενδιαφέρον. Ωστόσο, υπάρχουν πολλές δυσκολίες για την αποτελεσματική εκτέλεση τέτοιων αλγορίθμων οι οποίες οφείλονται στους περιορισμένους υπολογιστικούς πόρους που μπορεί να έχει μια κινητή συσκευή. Προκειμένου οι αλγόριθμοι να ενσωματωθούν σωστά, καθώς και για να εκτελούνται σε λογικό χρόνο, εκτελείται μια σειρά τεχνικών, που είναι διαθέσιμες τα τελευταία χρόνια, και κάνουν πλέον δυνατό το παραπάνω εγχείρημα.

Στη μελέτη αυτή αρχικά θα παρουσιαστεί το θεωρητικό υπόβαθρο, καθώς και οι τεχνολογίες και τεχνικές που χρησιμοποιούνται προκειμένου να αναπτυχθεί η ζητούμενη εφαρμογή. Τέλος, θα παρουσιαστεί η ανάπτυξη της εφαρμογής καθώς θα μετρηθεί και χρόνος λειτουργιών που σχετίζονται τόσο με την εύρεση του κειμένου εισόδου και την προεπεξεργασία του όσο και με την συμπερασματολογία κάθε μοντέλου προκειμένου να αξιολογηθούν οι αποδόσεις τους στη συσκευή.

Λέξεις Κλειδιά

Βαθιά Μάθηση, Επεξεργασία Φυσικής Γλώσσας, Εφαρμογή Android, Κινητές Συσκευές, Νευρωνικά Δίκτυα, Συμπερασματολογία, BERT, TensorFlow Lite

Abstract

In this diploma thesis, an application for Android mobile devices is designed and developed. More specifically, this application integrates the running of some pre-trained Deep Machine Learning algorithms that perform Natural Language Processing tasks with a mobile Application. Essentially, an application is designed through which users have the ability to collect text from any page on the internet, in order to feed it as an input to a number of available Neural Networks within the application. Some of these Neural Networks perform their inference on the task of Question and Answering and others on the task of Text Classification, depending on their tone.

The purpose of this analysis is both the development of the application mentioned above with the integration of Deep Machine Learning algorithms, which will be executed locally on the device, and the evaluation of the execution times of these algorithms. Nowadays, running such algorithms locally on mobile devices has gathered great interest. However, there are many difficulties in performing such algorithms efficiently due to the limited computing resources that a mobile device may have. In order for the algorithms to be properly integrated, as well as to run in a reasonable time, a number of optimisation techniques are performed, which are available in recent years, and now make the above task possible.

This study will present the theoretical background, as well as the technologies and techniques used in order to develop the requested application. Finally, both of the development of the application and a set of measurements will be presented. This measurements are related with the procedure of finding the input text, the procedure of pre-processing and the inference of each model in order to evaluate their performance on the device.

Keywords

Android Application, Deep Learning, Natural Language Processing, Mobile Devices, Neural Networks, Inference, BERT, TensorFlow Lite

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέπων καθηγητή αυτής της διπλωματικής, κ. Ιάκωβο Βενιέρη, για την ευκαιρία που μου έδωσε να ασχοληθώ με το συγκεκριμένο θέμα, το ερέθισμα που μου προσέφερε να γνωρίσω τον τομέα του Android και της Μηχανικής Μάθησης, καθώς και για το ενδιαφέρον που μου καλλιέργησε κατά τη διάρκεια των σπουδών μου.

Ιδιαίτερες ευχαριστίες θα ήθελα να δώσω στον υποψήφιο διδάκτορα Ιωάννη Πανόπουλο για την καθοδήγησή του και τη διαρκή και άμεση στήριξή του. Τόσο οι τεχνικές όσο και οι θεωρητικές συζητήσεις μας πάνω στο θέμα της διπλωματικής αυτής αλλά και άλλων εργασιών μου δίδαξαν πολλά και ενίσχυσαν το ενδιαφέρον μου να συνεχίσω συνεχώς να μαθαίνω.

Ευχαριστώ, ακόμα, τα μέλη της επιτροπής, κα. Δήμητρα - Θεοδώρα Κακλαμάνη και κ. Αθανάσιο Παναγόπουλο, για την συμμετοχή τους στην τριμελή εξεταστική επιτροπή καθώς και ολόκληρο το εργαστήριο Ευφυών Επικοινωνιών και Δικτύων Ευρείας Ζώνης.

Έπειτα, θέλω να ευχαριστήσω τους φίλους και κοντινούς μου ανθρώπους, με τους οποίους είμαστε συνοδοιπόροι όλα αυτά τα αξέχαστα χρόνια. Ευχαριστώ ιδιαίτερα τον Λευτέρη, που υπήρξε το μεγαλύτερο στήριγμά μου καθ όλη τη διάρκεια αυτής της μελέτης.

Τέλος, το μεγαλύτερο ευχαριστώ οφείλω στην οικογένεια μου, στους γονείς μου και στις αδερφές μου για την ανιδιοτελή στήριξη και αγάπη τους όλα αυτά τα χρόνια.

Αθήνα, Μάρτιος 2022

Δάβαρη Ελένη

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Στο κεφάλαιο αυτό θα δοθούν βασικές πληροφορίες σχετικά με έννοιες που θα χρησιμοποιηθούν στην συνέχεια, προκειμένου να δοθεί στον αναγνώστη η απαραίτητη εξοικείωση με το αντικείμενο της μελέτης που θα ακολουθήσει. Τελικώς θα παρουσιαστεί συνοπτικά ο στόχος της παρούσας διπλωματικής εργασίας.

1.1 Τα κινητά τηλέφωνα και οι εφαρμογές σήμερα

Τα έξυπνα τηλέφωνα (smartphones) σήμερα αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας μας. Το κινητό μέσω των εφαρμογών του μπορεί να προσφέρει μια πληθώρα λειτουργιών. Στις μέρες μας το κινητό τηλέφωνο είναι απαραίτητο για πολλές ενέργειες μέσα στην καθημερινότητα μας για αυτό η πλειοψηφία των ανθρώπων του σύγχρονου κόσμου διαθέτει τουλάχιστον μια τέτοια συσκευή, αφού πέρα από ένα προσωπικό τηλέφωνο πολλοί κατέχουν και ένα επαγγελματικό. Ουσιαστικά πλέον τα κινητά τηλέφωνα προσφέρουν δυνατότητες ενός υπολογιστή δίνοντας την ευκαιρία σύνδεσης δικτύου και την εγκατάστασης εφαρμογών λογισμικού, ενώ παράλληλα είναι μικρά σε μέγεθος, μετακινούνται εύκολα και είναι πιο απλά και πιο κατανοητά για τον μέσο χρήστη.

Έτσι λοιπόν, λόγω της χωρίς όρια αγοράς που έχουν δημιουργήσει τα κινητά τηλέφωνα και λόγω της όλο και αυξανόμενης ζήτησης, ο όγκος των διαθέσιμων εφαρμογών συνεχώς αυξάνεται. Μέχρι τώρα έχουν δημιουργηθεί εφαρμογές που καλύπτουν από πολύ σημαντικές μέχρι λιγότερο σημαντικές ανάγκες. Η δύναμη των τηλεφώνων και των εφαρμογών είναι τόσο μεγάλη που πλέον αυτά είναι τα μέσα που, όχι μόνο καλύπτουν υπάρχουσες ανάγκες, αλλά τελικά δημιουργούν νέες. Στις μέρες μας, είναι αλήθεια ότι υπάρχουν εφαρμογές για οτιδήποτε μπορείς να φανταστείς και πλέον χρησιμοποιούνται όλο και πιο εξελιγμένες τεχνικές που εκμεταλλεύονται ένα πολύ μεγάλο ποσοστό των δυνατοτήτων των κινητών τηλεφώνων τόσο σε επίπεδο λογισμικού όσο και σε επίπεδο υλικού.

Στον τομέα της οικονομίας, ένα μεγάλο ποσοστό επιχειρήσεων, εκτός από μια σελίδα στο διαδίκτυο, έχουν και την προσωπική τους εφαρμογή μέσα από όπου γίνεται καλύτερα και ευκολότερα η ανταλλαγή προϊόντων ή υπηρεσιών. Μελέτες δείχνουν ότι μέσω εφαρμογών όλες οι επιχειρήσεις μπορούν να έχουν μεγαλύτερο ποσοστό κερδών. Αυτό οφείλεται στο ποσοστό χρήσης των εφαρμογών σήμερα. Συγκεκριμένα, ο μέσος χρήστης ελέγχει το κινητό του κάθε 18 λεπτά. Όλα συμβαίνουν σε κάποια εφαρμογή και αυτός είναι και ο λόγος που κάθε επιχείρηση θέλει να έχει τη δική της, αφού έτσι της δίνεται η δυνατότητα να συνδεθεί

με τον χρήστη και να τον ενημερώνει σχετικά με την εξέλιξη των προϊόντων ή των υπηρεσιών της. Έτσι λοιπόν πλέον υπάρχει πάνω από μια εφαρμογή για οτιδήποτε χρειαστείς.

Η πληθώρα εφαρμογών έχει οδηγήσει στην αύξηση του ανταγωνισμού. Ο αριθμός των εφαρμογών που χρησιμοποιούνται από τους χρήστες αυξάνονται. Ο αριθμός των εφαρμογών που σχεδιάζονται από τις επιχειρήσεις αυξάνονται. Στην πραγματικότητα, υπάρχουν εφαρμογές που διαχειρίζονται άλλες εφαρμογές στο κινητό. Το IoT (Internet of Things) μαζί με τις τελευταίες τεχνολογικές τάσεις στη δημιουργία εφαρμογών για κινητά, όπως η Μηχανική Μάθηση, προσφέρουν στους χρήστες έναν πιο έξυπνο και πιο απλό τρόπο να κάνουν πράγματα [16]. Για κάποιον εκτός του τομέα της πληροφορικής δεν είναι κατανοητό το πόσο καθοριστική συμβολή έχει η εφαρμογή της Μηχανικής Μάθησης σε διάφορες εφαρμογές για αυτό και αξίζει να δοθούν ορισμένα παραδείγματα. Ένα απλό παράδειγμα είναι ότι η Βαθιά Μηχανική Μάθηση συμβάλλει στη βελτίωση της εμπειρίας αυτόματης μετάφρασης κειμένου επιτρέποντας μεταφράσεις τόσο από κείμενο, όσο και από εικόνες σε πλήθος γλωσσών εύκολα, γρήγορα και με ακρίβεια. Στη συνέχεια, η αναγνώριση γλώσσας σε εφαρμογές είναι και αυτή μια εφαρμογή της Μηχανικής Μάθησης, όπως και εφαρμογές με ηχητικές εντολές, και εφαρμογές μετατροπής κειμένου από ήχο σε γραπτό κείμενο. Επίσης, στο τηλέφωνο μας μπορούμε να παρατηρήσουμε κατηγοριοποίηση των φωτογραφιών στην συλλογή μας με βάση τα άτομα, τις τοποθεσίες και τις καταστάσεις, κατηγοριοποίηση που γίνεται μέσω αλγορίθμων αναγνώρισης εικόνας. Τέλος, πολύ σημαντική συμβολή της μηχανικής Μάθησης στις εφαρμογές είναι η δυνατότητα επεξεργασίας δεδομένων. Ένας αλγόριθμος Μηχανικής Μάθησης μπορεί να επεξεργαστεί έναν μεγάλο όγκο πληροφοριών πιο γρήγορα και με μεγαλύτερη ακρίβεια από τον άνθρωπο ενώ σε ορισμένες περιπτώσεις κάποια μοτίβα δεν θα ήταν καν ορατά με μια απλή ανάλυση.

Όλα αυτά τα παραπάνω παραδείγματα και πολλά περισσότερα έχουν αλλάξει ριζικά την εμπειρία του χρήστη μέσα στις εφαρμογές γεγονός που λειτουργεί ως κίνητρο για την συνεχή βελτίωση τους. Δεδομένου λοιπόν του τόσο μεγάλου ενδιαφέροντος που έχει αναπτυχθεί γύρω από τις εφαρμογές, καθώς και την ενσωμάτωση της Μηχανικής Μάθησης σε αυτές, στην παρούσα ανάλυση ασχολούμαστε εκτενώς με την ανάπτυξη μιας τέτοιας εφαρμογής στην οποία αλγόριθμοι Μηχανικής Μάθησης τρέχουν τοπικά στη συσκευή και εκτελούν εργασίες που σχετίζονται με την επεξεργασία κειμένου.

1.2 Αντικείμενο της διπλωματικής εργασίας

Όπως προαναφέρθηκε τα τελευταία χρόνια τα Νευρωνικά Δίκτυα συμπεριλαμβάνονται και σε εφαρμογές για κινητές συσκευές αυξάνοντας έτσι την αποτελεσματικότητα τους. Στην παρούσα διπλωματική εργασία θα γίνει μια ανάλυση της λειτουργίας διαφόρων γνωστών Νευρωνικών Δικτύων σε εργασίες Επεξεργασίας Φυσικής Γλώσσας. Συγκεκριμένα, προεκπαιδευμένα Νευρωνικά Δίκτυα θα ενσωματωθούν σε μια εφαρμογή για Android συσκευή, όπου και θα εκτελούνται τοπικά. Με λίγα λόγια στόχος της παρούσας εργασίας είναι η δημιουργία μιας εφαρμογής η οποία θα δίνει την δυνατότητα εξαγωγής κειμένων από οποιαδήποτε σελίδα του διαδικτύου δίνοντας έτσι τη δυνατότητα δοκιμής διαφόρων προεκπαιδευμένων μοντέλων στις εργασίες της Απάντησης Ερωτήσεων και της Ταξινόμησης Κειμένων σε κινητές συσκευές. Μετά την υλοποίηση της εφαρμογής θα πραγματοποιηθεί υπολογισμός

μετρήσεων σχετικά με τον χρόνο εκτέλεσης των μοντέλων και σχετικά με την προεπεξεργασία του κειμένου. Ο χρόνος προεπεξεργασίας είναι ο χρόνος μέσα στον οποίο γίνεται η συλλογή του κειμένου μέσω της εφαρμογής και στη συνέχεια ο χρόνος μέσα στον οποίο εφαρμόζονται τεχνικές προεπεξεργασίας του κειμένου προκειμένου να δημιουργηθεί η μορφή της εισόδου όπως απαιτείται από το εκάστοτε μοντέλο.

1.3 Δομή και Οργάνωση της εργασίας

Η εργασία αυτή χωρίζεται σε 7 κεφάλαια. Στο Κεφάλαιο 1 αρχικά γίνεται μια παρουσίαση των όσων θα ασχοληθούμε, του στόχου της διπλωματικής καθώς και της δομής της εργασίας. Στη συνέχεια, στο Κεφάλαιο 2 γίνεται μια εκτενής παρουσίαση του θεωρητικού υπόβαθρου, όπου αναλύονται γενικά γνωστές τεχνικές στην Επεξεργασία Φυσικής Γλώσσας και ειδικά οι τεχνικές που χρησιμοποιούνται στην παρούσα μελέτη τόσο στο επίπεδο προεπεξεργασίας του κειμένου προκειμένου να εισαχθεί στο εκάστοτε μοντέλο, όσο και σε επίπεδο εκτέλεσης του μοντέλου. Ύστερα, στο Κεφάλαιο 3 παρουσιάζονται οι βασικές τεχνικές και εργαλεία που χρησιμοποιήθηκαν κατά την ανάλυση αυτή. Αφού έχει γίνει η παραπάνω ανάλυση στο Κεφάλαιο 4, δίνονται περισσότερες πληροφορίες για τα μοντέλα της εφαρμογής καθώς και τις τεχνικές που χρησιμοποιήθηκαν προκειμένου να γίνει η ενσωμάτωση και εκτέλεση τους στην κινητή συσκευή. Αφού έχει γίνει η παρουσίαση του μέρους της Βαθιάς Μηχανικής Μάθησης που εφαρμόζεται στην εφαρμογή στο Κεφάλαιο 5 αναλύεται η σχεδίαση και ανάπτυξη της εφαρμογής. Τέλος, στο Κεφάλαια 6, παρουσιάζονται αποτελέσματα μετρήσεων για την καταγραφή του χρόνου εκτέλεσης διαφόρων λειτουργιών της εφαρμογής και στη συνέχεια στο Κεφάλαιο 7 δίνεται ο επίλογος, λαμβάνονται συμπεράσματα και δίνονται προτάσεις για τυχόν επέκταση της παρούσας μελέτης.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

Στο κεφάλαιο αυτό παρουσιάζονται περισσότερες πληροφορίες σχετικά με τις έννοιες που θα μας απασχολήσουν στην παρούσα εργασία και αφορούν τον τομέα της Μηχανικής Μάθησης.

2.1 Μηχανική Μάθηση

Προκειμένου να ξεκινήσουμε την ανάλυση του αντικειμένου της παρούσας εργασίας, ξεκινάμε από την ανάλυση βασικών όρων. Η εξέλιξη της τεχνολογίας φέρνει μαζί της επαναστατικές ιδέες και τεχνικές. Εδώ και περισσότερο από πέντε δεκαετίες, ο τομέας της Μηχανικής Μάθησης αναπτύσσεται συνεχώς, με όλο και περισσότερα μέλη της τεχνολογικής κοινότητας να καταπιάνονται με τις τεχνικές της. Σε ένα γενικό πλαίσιο, η Μηχανική Μάθηση είναι το πεδίο μελέτης μέσα στο οποίο οι υπολογιστές έχουν πλέον την ικανότητα να μαθαίνουν μια διαδικασία, χωρίς να έχουν ρητά προγραμματιστεί για αυτή και μάλιστα, αφού την μάθουν, να αυτοβελτιώνονται συνεχώς, προκειμένου να αποκτήσουν ένα μεγάλο ποσοστό ακρίβειας εκτέλεσης. Σήμερα, η Μηχανική Μάθηση χρησιμοποιείται ευρέως σε πράγματα που χρησιμοποιούμε στην καθημερινή μας ζωή και κρύβεται πίσω από πολλές ευκολίες που μας προσφέρονται πλέον μέσω εφαρμογών.

Πιο αναλυτικά, η Μηχανική Μάθηση είναι ένας κλάδος της Τεχνητής Νοημοσύνης (AI: Artificial Inteligence) και της Επιστήμης των Υπολογιστών που εστιάζει στη χρήση δεδομένων και αλγορίθμων για τη μίμηση του τρόπου με τον οποίο μαθαίνουν οι άνθρωποι με σταδιακά βελτιωνόμενη ακρίβεια. Η Μηχανική Μάθηση είναι ένα σημαντικό συστατικό του αναπτυσσόμενου τομέα της επιστήμης δεδομένων (data science). Σε αυτή, μέσω της χρήσης στατιστικών μεθόδων, οι αλγόριθμοι εκπαιδεύονται να κάνουν ταξινομήσεις, προβλέψεις ή συσχετίσεις, αποκαλύπτοντας βασικές γνώσεις στα έργα εξόρυξης δεδομένων (data mining projects). Οι αλγόριθμοι Μηχανικής Μάθησης, αφού δημιουργηθούν, τροφοδοτούνται συνεχώς με νέες εισόδους συνόλων δεδομένων (data set). Στις μέρες μας, υπάρχει μια τεράστια βιβλιοθήκη διαθέσιμων συνόλων δεδομένων για διάφορες εργασίες, που σχετίζονται με την εικόνα, το κείμενο και τον ήχο. Ο όγκος των διαθέσιμων συνόλων δεδομένων συνεχώς αυξάνεται και κάποιος που καταπιάνεται με αλγορίθμους Μηχανικής Μάθησης μπορεί να βρει εύκολα πλήθος δειγμάτων τόσο με ετικέτες (labels), με πληροφορίες δηλαδή για το είδος των δεδομένων, όσο και χωρίς. Οι αλγόριθμοι Μηχανικής Μάθησης μέσω της "επαφής" τους με ένα πλήθος διαφορετικών εισόδων παράγουν αποτελέσματα και με βάση αυτά, ανεξαρτήτως

της ορθότητας ή μη των αποτελεσμάτων τους, αυτοβελτιώνονται, έως ότου παράγουν το όσο το δυνατόν καλύτερο αποτέλεσμα πάνω στην εργασία για την οποία δουλεύουν. Προφανώς, το πόσο καλά αποτελέσματα μπορεί να παράγει ένα μοντέλο εξαρτάται τόσο από τον κώδικα αυτόν καθ' αυτόν όσο και από τον αριθμό των διαφορετικών εισόδων για τις οποίες εκτελείται. Η διαδικασία αυτή τροφοδότησης εισόδων ονομάζεται εκπαίδευση (training). Σύμφωνα με το πανεπιστήμιο του Berkeley το εκπαιδευτικό σύστημα ενός αλγορίθμου Μηχανικής Μάθησης χωρίζεται σε τρία κύρια στάδια:

1. **Τη διαδικασία απόφασης:** Γενικά, οι αλγόριθμοι Μηχανικής Μάθησης χρησιμοποιούνται για να κάνουν προβλέψεις ή ταξινόμηση δεδομένων. Με βάση τα δεδομένα εισόδου, ένας αλγόριθμος Μηχανικής Μάθησης θα παράγει μια εκτίμηση σχετικά με ένα μοτίβο που μπορεί να υπάρχει στα δεδομένα αυτά.

2. **Τη συνάρτηση σφάλματος:** Μια συνάρτηση σφάλματος χρησιμεύει για την αξιολόγηση της πρόβλεψης του μοντέλου. Εάν υπάρχουν γνωστά παραδείγματα, μια συνάρτηση σφάλματος μπορεί να κάνει σύγκριση, για να αξιολογήσει την ακρίβεια του μοντέλου.

3. **Τη διαδικασία βελτιστοποίησης του μοντέλου:** Εάν το μοντέλο μπορεί να "ταιριάζει" καλύτερα στα σημεία δεδομένων του συνόλου εκπαίδευσης, τότε το μοντέλο αναπροσαρμόζεται, για να μειωθεί η απόκλιση μεταξύ του γνωστού παραδείγματος και της εκτίμησης τους. Ο αλγόριθμος θα επαναλάβει αυτή τη διαδικασία αξιολόγησης και βελτιστοποίησης, έως ότου επιτευχθεί ένα όριο ακρίβειας [17].

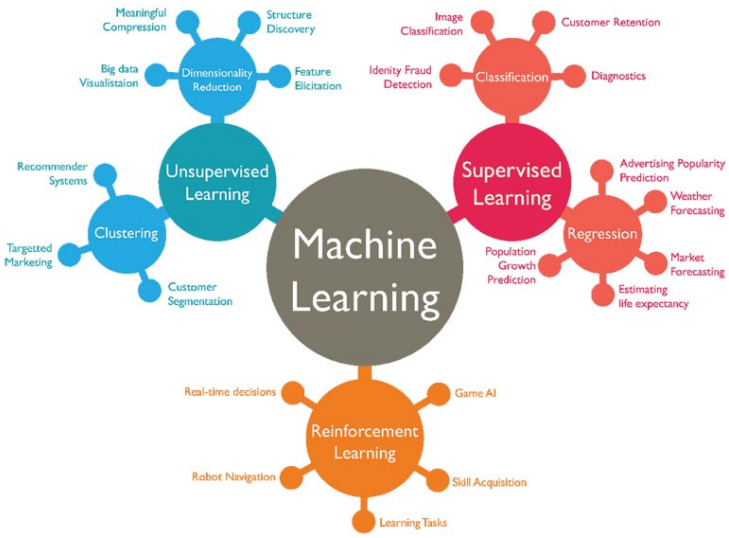
Η Μηχανική Μάθηση χωρίζεται σε τρεις βασικές κατηγορίες, στην Επιβλεπόμενη (Supervised), την μη Επιβλεπόμενη (Unsupervised) και την Ενισχυτική (Reinforcement) Μηχανική Μάθηση, οι οποίες φαίνονται και στην Εικόνα 2.1 [17]. Περισσότερες πληροφορίες δίνονται παρακάτω:

- **Επιβλεπομένη Μάθηση.** Στις περισσότερες περιπτώσεις, κατά την εφαρμογή της Μηχανικής Μάθησης, χρησιμοποιείται η Επιβλεπόμενη Μάθηση. Η Επιβλεπόμενη Μάθηση ορίζεται από τη χρήση επισημασμένων συνόλων δεδομένων για την εκπαίδευση αλγορίθμων που προβλέπουν αποτελέσματα με ακρίβεια [18]. Με λίγα λόγια, στην Επιβλεπόμενη Μάθηση το σύνολο δεδομένων που χρησιμοποιείται έχει προεπισημανθεί και ταξινομηθεί με ανθρώπινη παρέμβαση, για να επιτρέψει στην συνέχεια στον αλγόριθμο να δει πόσο ακριβής είναι η απόδοσή του [19]. Πρακτικά, χρησιμοποιείται μια μεταβλητή εισόδου, μια εξόδου και μια συνάρτηση που τις συνδέει. Στόχος της διαδικασίας είναι η συνάρτηση αυτή να προσεγγιστεί με τέτοια ακρίβεια, έτσι ώστε, αν υπάρξουν νέα δεδομένα εισόδου, να μπορεί να προβλέψει σωστά τα αντίστοιχα δεδομένα εξόδου [18].

- **Μη Επιβλεπομένη Μάθηση.** Από την άλλη, η μη Επιβλεπόμενη Μάθηση χρησιμοποιεί αλγόριθμους Μηχανικής Μάθησης για την ανάλυση και την ομαδοποίηση συνόλων δεδομένων χωρίς ετικέτες, δηλαδή το σύνολο δεδομένων που χρησιμοποιείται είναι ακατέργαστο και ο αλγόριθμος προσδιορίζει μοτίβα και σχέσεις εντός των δεδομένων χωρίς εξωτερική βοήθεια [19]. Η ικανότητα ενός αλγορίθμου Μη Επιβλεπόμενης

Μάθησης να ανακαλύπτει ομοιότητες και διαφορές στις πληροφορίες το καθιστά την ιδανική λύση για διερευνητική ανάλυση δεδομένων και αναγνώριση εικόνας. Χρησιμοποιείται επίσης για τη μείωση του αριθμού των χαρακτηριστικών σε ένα μοντέλο μέσω της διαδικασίας μείωσης διαστάσεων [20].

- **Ενισχυτική Μάθηση**. Τέλος, η Ενισχυτική Μάθηση είναι η διαδικασία μάθησης κατά την οποία γίνεται εκπαίδευση μοντέλων Μηχανικής Μάθησης για τη λήψη μιας σειράς αποφάσεων. Το μοντέλο που εκπαιδεύεται μαθαίνει να επιτυγχάνει έναν στόχο σε ένα αβέβαιο, δυνητικά περίπλοκο περιβάλλον. Στην Ενισχυτική μάθηση, το κάθε μοντέλο αντιμετωπίζει μια κατάσταση που μοιάζει με παιχνίδι. Ο υπολογιστής χρησιμοποιεί τη δοκιμή και το σφάλμα για να βρει μια λύση στο πρόβλημα. Προκειμένου η Τεχνητή Νοημοσύνη να κάνει αυτό που θέλει ο προγραμματιστής, λαμβάνει είτε ανταμοιβές, είτε ποινές για τις ενέργειες που εκτελεί. Στόχος αυτής της διαδικασίας είναι η μεγιστοποίηση της συνολικής ανταμοιβής. Ο σχεδιαστής του προβλήματος ορίζει την πολιτική ανταμοιβής, δηλαδή τους κανόνες του παιχνιδιού, ενώ δεν δίνει στο μοντέλο υποδείξεις ή προτάσεις για το πώς να λύσει το πρόβλημα. Με το τέλος της διαδικασίας το μοντέλο πρέπει να καταλάβει πώς να εκτελέσει την εργασία για να μεγιστοποιήσει την ανταμοιβή, ξεκινώντας από εντελώς τυχαίες δοκιμές και τελειώνοντας με εξελιγμένες τακτικές και υπεράνθρωπες δεξιότητες. Αξιοποιώντας τη δύναμη της αναζήτησης και πολλών δοκιμών, η ενισχυτική μάθηση είναι επί του παρόντος ο πιο αποτελεσματικός τρόπος εκπαίδευσης μιας μηχανής. Σε αντίθεση με τα ανθρώπινα όντα, η Τεχνητή Νοημοσύνη μπορεί να συγκεντρώσει εμπειρία από χιλιάδες παράλληλα παιχνίδια, εάν ένας αλγόριθμος Ενισχυτικής μάθησης εκτελείται σε μια αρκετά ισχυρή υποδομή υπολογιστή [21].



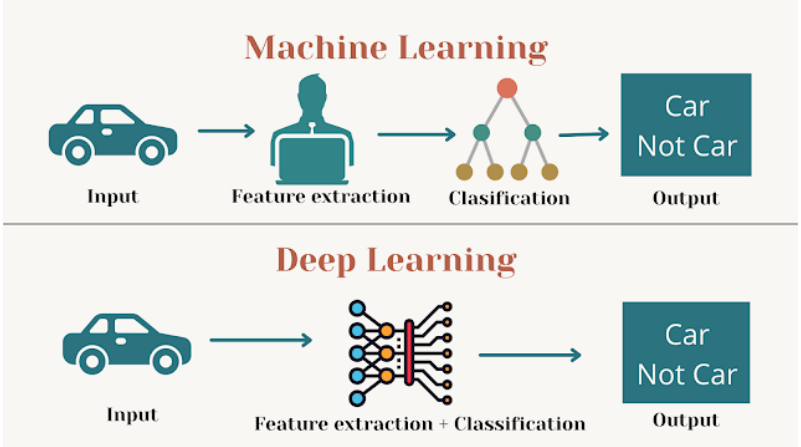
Εικόνα 2.1: *Κατηγορίες Μηχανικής Μάθησης και πεδία εφαρμογής [1]*

2.2 Βαθιά Μηχανική Μάθηση

Στη συνέχεια, δεδομένου ότι η Βαθιά Μηχανική Μάθηση ή απλώς Βαθιά Μάθηση και η Μηχανική Μάθηση τείνουν να χρησιμοποιούνται εναλλακτικά, αξίζει να σημειωθούν οι διαφορές μεταξύ των δύο [17]. Η προσέγγιση της Βαθιάς Μάθησης προσπαθεί να μοντελοποιήσει τον τρόπο που ο ανθρώπινος εγκέφαλος επεξεργάζεται το φως και τον ήχο και τα μετατρέπει σε όραση και ακοή. Για αυτόν ακριβώς τον λόγο ορισμένες επιτυχείς εφαρμογές της Βαθιάς μάθησης είναι η μηχανική όραση και η αναγνώριση ομιλίας [22]. Πιο συγκεκριμένα, η Μηχανική Μάθηση, η Βαθιά Μάθηση καθώς και τα Νευρωνικά Δίκτυα είναι όλα υποπεδία της Τεχνητής Νοημοσύνης. Στην πραγματικότητα, η Βαθιά Μάθηση είναι ένα υποπεδίο της Μηχανικής Μάθησης. Ο τρόπος με τον οποίο η Βαθιά Μάθηση και η Μηχανική Μάθηση διαφέρουν είναι στο πώς μαθαίνει κάθε αλγόριθμος.

Αναλυτικότερα, η Βαθιά Μάθηση αυτοματοποιεί μεγάλο μέρος του τμήματος εξαγωγής χαρακτηριστικών της διαδικασίας, εξαλείφοντας μέρος της χειροκίνητης ανθρώπινης παρέμβασης που απαιτείται επιτρέποντας τη χρήση μεγαλύτερων συνόλων δεδομένων. Η κλασική ή «μη βαθιά» Μηχανική Μάθηση εξαρτάται περισσότερο από την ανθρώπινη παρέμβαση για μάθηση. Οι ειδικοί καθορίζουν το σύνολο των χαρακτηριστικών, για να κατανοήσουν τις διαφορές μεταξύ των εισροών δεδομένων, που συνήθως απαιτούν περισσότερα δομημένα δεδομένα για την εκμάθηση. Κατ' αυτόν τον τρόπο η Βαθιά Μάθηση μπορεί να απορροφήσει μη δομημένα δεδομένα στην ακατέργαστη μορφή τους (πχ. κείμενο, εικόνες) και μπορεί να προσδιορίσει αυτόματα το σύνολο των χαρακτηριστικών που διακρίνουν διαφορετικές κατηγορίες δεδομένων μεταξύ τους επιτρέποντάς μας έτσι να κλιμακώνουμε τη Μηχανική Μάθηση με πιο ενδιαφέροντες τρόπους. Στην Εικόνα 2.2 γίνεται μια οπτική αναπαράσταση των διαφορών Μηχανικής Μάθησης και Ναθιάς Μάθησης για περαιτέρω κατανόηση.

Το «βαθύ» στη Βαθιά Μάθηση αναφέρεται στο βάθος των στρωμάτων σε ένα Νευρωνικό Δίκτυο. Ένα Νευρωνικό Δίκτυο που αποτελείται από περισσότερα από τρία επίπεδα - τα οποία θα περιλαμβάνουν τις εισόδους και τις εξόδους - μπορεί να θεωρηθεί αλγόριθμος Βαθιάς Μάθησης ή Βαθύ Νευρωνικό Δίκτυο. Ένα Νευρωνικό Δίκτυο που έχει μόνο δύο ή τρία επίπεδα είναι απλώς ένα βασικό Νευρωνικό Δίκτυο [17]. Περισσότερες πληροφορίες σχετικά με τα Νευρωνικά Δίκτυα θα δοθούν στο επόμενο κεφάλαιο.



Εικόνα 2.2: *Διαφορά μεταξύ Μηχανικής Μάθησης και Βαθιάς Μάθησης [2]*

2.2.1 Νευρωνικά Δίκτυα

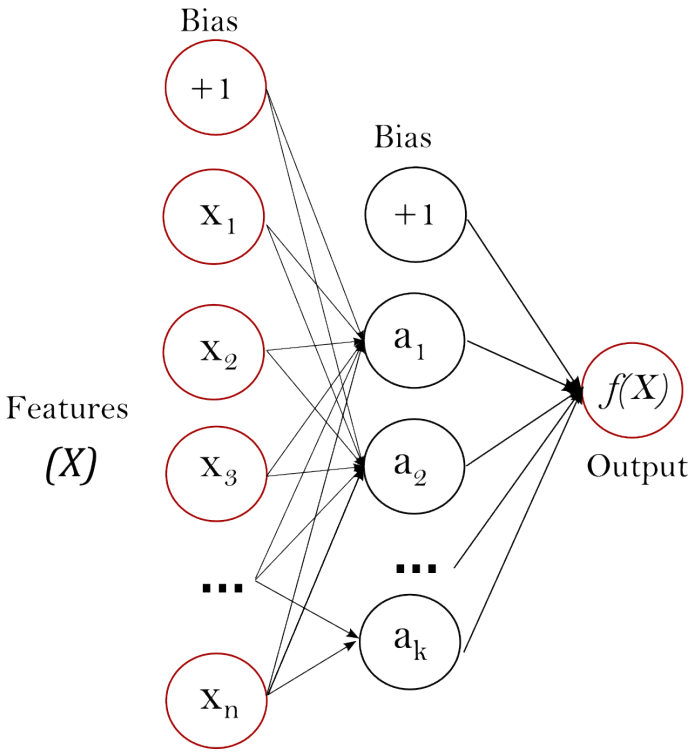
Τα Τεχνητά Νευρωνικά Δίκτυα (ANNs: Artificial Neural Networks), επίσης γνωστά ως Νευρωνικά Δίκτυα ή αλλιώς υπολογιστικά μοντέλα είναι ουσιαστικά αλγόριθμοι που έχουν μια μοναδική ικανότητα να εξάγουν νόημα από ανακριβή ή σύνθετα δεδομένα, για να βρουν μοτίβα και να ανιχνεύσουν τάσεις που είναι πολύ περίπλοκες για τον ανθρώπινο εγκέφαλο ή για άλλες τεχνικές υπολογιστών. Τα Νευρωνικά Δίκτυα μας έχουν προσφέρει μεγαλύτερη ευκολία με πολλούς τρόπους. Εκπαιδεύονται είτε με διαδικασία Επιβλεπομένης, είτε μη Επιβλεπομένης ή άλλα είδη μάθησης, επιλογή που εξαρτάται από τις εφαρμογές για τις οποίες προορίζονται. Τα Νευρωνικά Δίκτυα, όπως φανερώνει και το όνομα τους μιμούνται τους ανθρώπινους εγκεφάλους, οι οποίοι αποτελούνται από νευρώνες, το θεμελιώδες δομικό στοιχείο της μετάδοσης πληροφοριών, τόσο του ανθρώπου όσο και του Νευρωνικού Δικτύου [23]. Με απλά λόγια, ένα Νευρωνικό Δίκτυο προσομοιώνει τη λειτουργία του νευρικού συστήματος, είναι δηλαδή ένα απλοποιημένο μοντέλο του τρόπου με τον οποίο ο ανθρώπινος εγκέφαλος επεξεργάζεται τις πληροφορίες. Ακολουθώντας αυτό το μοτίβο λοιπόν οι αλγόριθμοι αυτοί εκπαιδεύονται, μαθαίνουν, βελτιώνονται και προσαρμόζονται όπως ακριβώς και ο ανθρώπινος εγκέφαλος [24].

Τα Νευρωνικά Δίκτυα όπως προαναφέρθηκε στηρίζονται στον τρόπο λειτουργίας του νευρικού συστήματος έτσι λοιπόν οι βασικές μονάδες τους είναι οι νευρώνες, οι οποίοι είναι συνήθως οργανωμένοι σε στρώματα [23]. Υπάρχουν συνήθως τρία μέρη σε ένα Νευρωνικό Δίκτυο: το επίπεδο εισόδου, με μονάδες που αντιπροσωπεύουν τα πεδία εισόδου, ένα ή περισσότερα κρυφά επίπεδα και το επίπεδο εξόδου, με μια μονάδα ή μονάδες που αντιπροσωπεύουν το πεδίο στόχο ή τα πεδία στόχους. Οι μονάδες συνδέονται με διαφορετικές αντοχές ή αλλιώς βάρη σύνδεσης. Τα δεδομένα εισόδου εισάγονται στο πρώτο επίπεδο και οι τιμές διαδίδονται από κάθε νευρώνα σε κάθε νευρώνα στο επόμενο επίπεδο [24]. Κατά τη διάδοση των τιμών μέσα στο Νευρωνικό Δίκτυο σε κάθε επίπεδο ανάλογα με την είσοδο ενεργοποιείται κάθε φορά ένας νευρώνας από κάθε επίπεδο μέσω της Συνάρτησης Ενεργοποίησης.

Προκειμένου να γίνουν περισσότερο κατανοητοί οι παραπάνω όροι θα αναλυθεί το Multi Layer Perceptron (MLP) που αποτελεί το πιο απλό είδος Νευρωνικού Δικτύου και εκπαιδεύεται με μεθόδους Επιβλεπόμενης Μηχανικής Μάθησης. Στην Εικόνα 2.3 φαίνεται ένα MLP δίκτυο με ένα κρυφό επίπεδο που είναι και η πιο απλή μορφή που μπορούμε να το συναντήσουμε. Δοσμένου ενός διανύσματος χαρακτηριστικών $X = x_1, x_2, ..., x_n$ και έναν στόχο y μπορεί να μάθει μια μη γραμμική προσέγγιση συνάρτησης είτε για ταξινόμηση (classification) είτε για παλινδρόμηση (regression). Η παλινδρόμηση στη Μηχανική Μάθηση αποτελείται από μαθηματικές μεθόδους που επιτρέπουν στους επιστήμονες δεδομένων να προβλέψουν ένα αποτέλεσμα y με βάση την τιμή μιας ή περισσότερων μεταβλητών πρόβλεψης x. Η γραμμική παλινδρόμηση είναι ίσως η πιο δημοφιλής μορφή ανάλυσης παλινδρόμησης λόγω της ευκολίας χρήσης της στην πρόβλεψη. Το αριστερότερο επίπεδο, γνωστό ως επίπεδο εισόδου, αποτελείται από νευρώνες χαρακτηριστικών $\{x_i|x_1, x_2, ..., x_m\}$ που αντιπροσωπεύουν τα χαρακτηριστικά εισόδου. Κάθε νευρώνας στο κρυφό στρώμα μετασχηματίζει τις τιμές από το προηγούμενο επίπεδο με ένα σταθμισμένο γραμμικό άθροισμα $w_1x_1 + w_1x_1 + ... + w_mx_m$, όπου w_i το βάρος του κάθε χαρακτηριστικού x_i. Αυτό ακολουθείται από μια μη γραμμική

συνάρτηση ενεργοποίησης και τέλος το επίπεδο εξόδου λαμβάνει τις τιμές από το τελευταίο κρυφό στρώμα και τις μετατρέπει σε τιμές εξόδου. Η συνάρτηση ενεργοποίησης ουσιαστικά προσθέτει στο γραμμικό άθροισμα έναν σταθερό όρο, που ορίζεται ως προκατάληψη με τον εξής τρόπο:

$$y = f(X) = \sum_{n=1}^{m} w_n x_n + b$$



Εικόνα 2.3: *MLP με ένα κρυφό επίπεδο [3]*

Στην περίπτωση σωστής πρόβλεψης από τον αλγόριθμο, αλλά και στην περίπτωση λανθασμένης πρόβλεψης, οι νευρώνες προσαρμόζουν συνεχώς τον τρόπο που αντιδρούν με βάση τα ερεθίσματα. Εάν κάτι γίνει σωστά, θα ληφθεί θετική ανατροφοδότηση από νευρώνες, οι οποίοι στη συνέχεια θα είναι ακόμη πιο πιθανό να ενεργοποιηθούν σε μια παρόμοια, μελλοντική περίπτωση. Αντίθετα, εάν οι νευρώνες λάβουν αρνητική ανάδραση, καθένας από αυτούς θα μάθει να είναι λιγότερο πιθανό να ενεργοποιηθεί σε μια μελλοντική περίπτωση. Στόχος της διαδικασίας αυτής είναι η όσο το δυνατόν καλύτερη πρόβλεψη του μοντέλου. Η συνάρτηση σύμφωνα με την οποία υπολογίζεται το πόσο διαφέρει η προβλεπόμενη τιμή από την αναμενόμενη ονομάζεται Συνάρτηση Απώλειας (Loss Function), επομένως στόχος είναι η βελτιστοποίηση της συνάρτησης αυτής. Μέθοδοι που χρησιμοποιούνται για εκπαίδευση είναι η Κατάβαση Κλίσης (Gradient Descent) και η Οπισθοδρόμηση (Backpropagation) [23, 3].

Για να ανακεφαλαιώσουμε και να μιλήσουμε με όρους, ένα Νευρωνικό Δίκτυο αρχικά περνάει από το στάδιο της εκπαίδευσης (Training), όπου ένα σύνολο εισόδων εισάγεται στο μοντέλο και αυτό παράγει μια πρόβλεψη για κάθε μια από αυτές. Ανάλογα με το αν η πρόβλεψη είναι σωστή ή λανθασμένη, γίνεται μια αναπροσαρμογή των βαρών του και στη συνέχεια ακολουθεί το στάδιο της συμπερασματολογίας (Inference), όπου το εκπαιδευμένο

Νευρωνικό Δίκτυο εφαρμόζει τις δεξιότητές του σε νέα δεδομένα προκειμένου να διαπιστωθεί η ακρίβεια του σε άλλα σύνολα δεδομένων με τα οποία δεν έχει έρθει ξανά σε επαφή. Εδώ αξίζει να αναφερθεί και ο όρος του Finetuning όπου σε ένα μοντέλο κατά την εκπαίδευση επιταχύνεται η διαδικασία με την αρχικοποίηση των βαρών του μοντέλου με βάση ένα άλλο εκπαιδευμένο μοντέλο στην ίδια εργασία [25].

Στις μέρες μας, τα Νευρωνικά Δίκτυα έχουν συνεχώς αυξανόμενες εφαρμογές. Για αυτό το λόγο όλο και περισσότεροι προγραμματιστές καταπιάνονται με αυτά γεγονός που οδηγεί τόσο στην εμφάνιση νέων μοντέλων όσο και στην εκπαίδευση και την βελτιστοποίηση ήδη υπάρχοντων μοντέλων. Σήμερα, γίνεται εκπαίδευση πολλών Νευρωνικών Δικτύων για μια μεγάλη ποικιλία από εργασίες (tasks). Κρίσιμες εργασίες που εκτελούν είναι:

- **Η ταξινόμηση**, όπου οργανώνουν μοτίβα ή σύνολα δεδομένων σε προκαθορισμένες κλάσεις.
- **Η πρόβλεψη**, όπου παράγουν την αναμενόμενη έξοδο από δεδομένη είσοδο.
- **Η ομαδοποίηση**, όπου προσδιορίζουν ένα μοναδικό χαρακτηριστικό των δεδομένων και το ταξινομούν χωρίς καμία γνώση προηγούμενων δεδομένων.
- **Η συσχέτιση**, όπου εκπαιδεύονται να «θυμούνται» μοτίβα. Όταν εμφανίζεται μια άγνωστη έκδοση ενός μοτίβου, το δίκτυο τη συσχετίζει με την πιο συγκρίσιμη έκδοση στη μνήμη του και επιστρέφει στην τελευταία.

Τα Νευρωνικά Δίκτυα είναι θεμελιώδη για τη Βαθιά Μάθηση, και μέσω αυτών προσφέρεται υλικό για την επίλυση αφηρημένων προβλημάτων, όπως προβλήματα βιοπληροφορικής, ο σχεδιασμός φαρμάκων, το φιλτράρισμα κοινωνικών δικτύων και η μετάφραση φυσικής γλώσσας. Η Βαθιά Μάθηση θεωρείται το μέσο για να λύσουμε τα πιο περίπλοκα ζητήματα στην επιστήμη και τη μηχανική, συμπεριλαμβανομένης της προηγμένης ρομποτικής. Καθώς τα Νευρωνικά Δίκτυα γίνονται πιο έξυπνα και πιο γρήγορα, κάνουμε προόδους σε καθημερινή βάση [23].

2.2.2 Συνελικτικά Νευρωνικά Δίκτυα

Τα Συνελικτικά Νευρωνικά Δίκτυα (CNN: Convolutional Neural Networks) δεδομένης της υπεροχής τους στο πεδίο της Όρασης Υπολογιστών, είναι φυσικό να δοκιμάζονται σε διαφορετικά πεδία Μηχανικής Μάθηση με πολλές προοπτικές στην επεξεργασία φυσικής γλώσσας [26]. Συγκεκριμένα, τα CNN είναι οι πιο ευρέως χρησιμοποιούμενες αρχιτεκτονικές Βαθιάς Μάθησης στην επεξεργασία και την αναγνώριση εικόνας. Το όνομα τους προέρχεται από το ότι ένα ή περισσότερα επίπεδα τους βασίζονται στην μαθηματική πράξη της συνέλιξης. Από μαθηματικής απόψεως η συνέλιξη δυο συναρτήσεων x(t) και w(t) ορίζεται ως η συνάρτηση s(t), όπου:

$$s(t) = (x * w)(t) = \int x(a)w(t-a)\,da$$

Στην περίπτωση δικτύων όπου εφαρμόζεται η συνέλιξη, συνήθως η συνάρτηση x αναφέρεται στην είσοδος, και η συνάρτηση w αναφέρεται ως πυρήνας (kernel) ή φίλτρο (filter).

Η έξοδος ορισμένες φορές αναφέρεται ως χάρτης χαρακτηριστικών (feature map) ή αλλιώς χάρτης ενεργοποίησης (activation map) [27]. Σχετικά με την γενική δομή του ένα CNN αποτελείται από:

- **Το Επίπεδο Συνέλιξης (Convolutional Layer)**, το οποίο αποτελεί την βασική μονάδα ενός CNN. Τα επίπεδα περιλαμβάνουν νευρώνες που σαρώνουν την είσοδο για μοτίβα. Οι παράμετροι ενός τέτοιου επιπέδου αποτελούνται από ένα σύνολο φίλτρων, τα οποία είναι μικρότερα σε μέγεθος από την είσοδο. Τα φίλτρα αυτά μαθαίνουν κατά την φάση εκπαίδευσης του δικτύου. Κάθε φίλτρο εντοπίζει και ένα διαφορετικό χαρακτηριστικό. Καθώς τα φίλτρα εφαρμόζονται στην εικόνα εισόδου, μέσω της συνέλιξης, παράγονται οι λεγόμενοι χάρτες χαρακτηριστικών (feature maps).

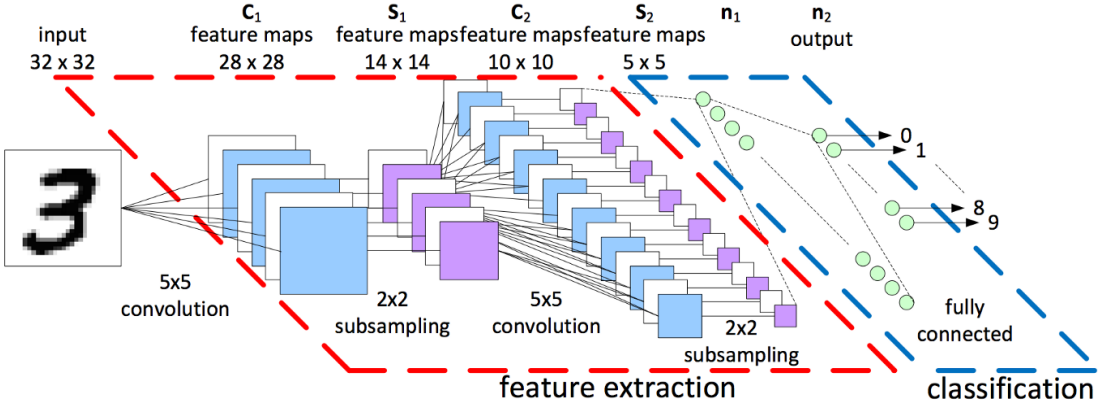
- **Το Επίπεδο Συγκέντρωσης (Pooling Layer)**, το οποίο είναι συχνά τοποθετημένο μετά από το Επίπεδο Συνέλιξης σε ένα CNN. Το επίπεδο αυτό κυρίως υπάρχει για τη μείωση της διάστασης του χάρτη χαρακτηριστικών με αποτέλεσμα την μεγαλύτερη υπολογιστική απόδοση η οποία μπορεί με τη σειρά της να βελτιώσει την πραγματική απόδοση. Η μείωση της διάστασης πρέπει να γίνει με παράλληλη διατήρηση των χαρακτηριστικών του χάρτη. Αυτό γίνεται αντικαθιστώντας μικρές περιοχές νευρώνων με μια στατιστική τιμή που υπολογίζεται από τις τιμές των νευρώνων αυτής της περιοχής. Υπάρχουν διάφορα είδη ομαδοποίησης όπως το Mean Pooling, όπου γίνεται η εξαγωγή του μέσου όρου των στοιχείων από τον χάρτη χαρακτηριστικών, το Max Pooling, όπου γίνεται η εξαγωγή του μεγαλύτερου στοιχείου από τον χάρτη και το Sum Pooling, όπου γίνεται η εξαγωγή αθροίσματος όλων των στοιχείων στον χάρτη.

- **Το Επίπεδο Κανονικοποίησης (Normalization Layer)**, το οποίο θα εφαρμόσει κάποια συνάρτηση κανονικοποίησης, όπως η συνάρτηση softmax έτσι ώστε να γίνει ταξινόμηση ενός αντικειμένου με πιθανολογικές τιμές μεταξύ 0 και 1 (κανονικοποίηση).

- **Το Πλήρως Συνδεδεμένο Δίκτυο**, το οποίο είναι το τελικό στάδιο του δικτύου, όπου ο πίνακας μετασχηματίζεται σε διάνυσμα και τροφοδοτείται σε ένα πλήρως συνδεδεμένο επίπεδο, όπως ένα MLP [27, 4].

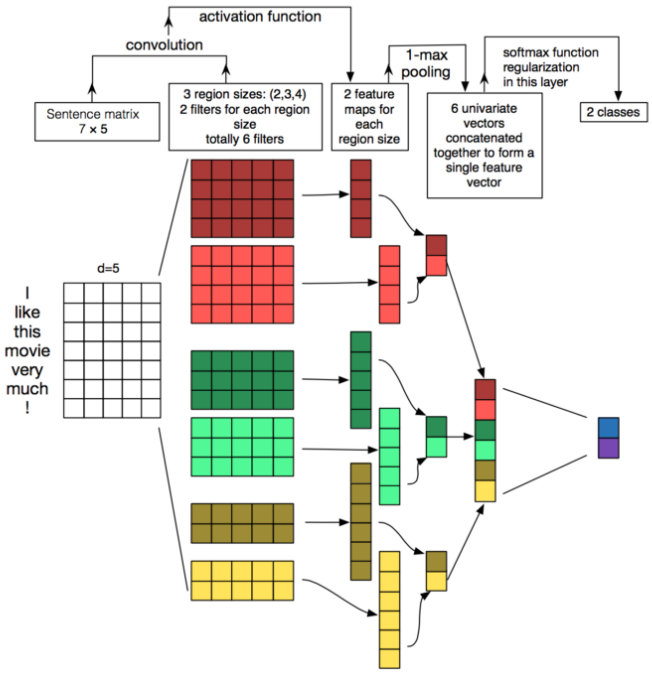
Για περισσότερη κατανόηση, στην Εικόνα 2.4 παρουσιάζεται η δομή ενός CNN, το οποίο χρησιμοποιείται σε εργασία ανάγνωσης από εικόνα το οποίο πραγματοποιεί αναγνώριση ψηφίου από σειρά εικόνων με χειρόγραφους χαρακτήρες για είσοδο, αφού όπως είπαμε η εφαρμογή των CNNs είναι πολύ διευρυμένη σε τέτοιου τύπου εργασίες. Μια εικόνα για να εισαχθεί στην είσοδο ενός Νευρωνικού Δικτύου, μετατρέπεται σε έναν δισδιάστατο πίνακα αριθμών, οι οποίοι αντιστοιχούν σε τιμές εικονοστοιχείων (pixel values). Μέσω των παραπάνω διαδικασιών εξάγονται χαρακτηριστικά μέσω αυτού του πίνακα τα οποία συνθέτουν τους χάρτες χαρακτηριστικών και στη συνέχεια την απάντηση του δικτύου.

Τώρα που υπάρχει μια βασική κατανόηση του πώς λειτουργεί ένα CNN με την παρουσίαση της εκτέλεσης του στην Όραση Υπολογιστών, θα συζητήσουμε περαιτέρω για την εφαρμογή του στην Επεξεργασία Φυσικής Γλώσσας. Ακριβώς όπως οι εικόνες μπορούν να



Εικόνα 2.4: *Γενική αρχιτεκτονική ενός CNN αναγνώρισης εικόνας [4]*

αναπαρασταθούν ως μια σειρά από τιμές εικονοστοιχείων (τιμές τύπου float), ομοίως μπορούμε να αναπαραστήσουμε το κείμενο ως μια σειρά διανυσμάτων (κάθε λέξη αντιστοιχίζεται σε ένα συγκεκριμένο διάνυσμα σε ένα διανυσματικό χώρο που αποτελείται από ολόκληρο το λεξιλόγιο) το οποίο μπορεί να υποβληθεί σε επεξεργασία με τη βοήθεια ενός CNN. Όταν εργαζόμαστε με διαδοχικά δεδομένα, όπως κείμενο, εργαζόμαστε με μονοδιάστατες συνελίξεις, αλλά η ιδέα και η εφαρμογή παραμένουν ίδια. Εξακολουθούμε να θέλουμε να συλλέξουμε μοτίβα στην ακολουθία που γίνονται πιο περίπλοκα με κάθε προστιθέμενο συνελικτικό στρώμα. Για περισσότερη κατανόηση στην Εικόνα 2.5 παρουσιάζεται η δομή ενός CNN το οποίο χρησιμοποιείται σε εργασία NLP [4].



Εικόνα 2.5: *Γενική αρχιτεκτονική ενός CNN ταξινόμησης κειμένου σε δεδομένες κλάσεις [4]*

2.2.3 Αναδρομικά Νευρωνικά Δίκτυα

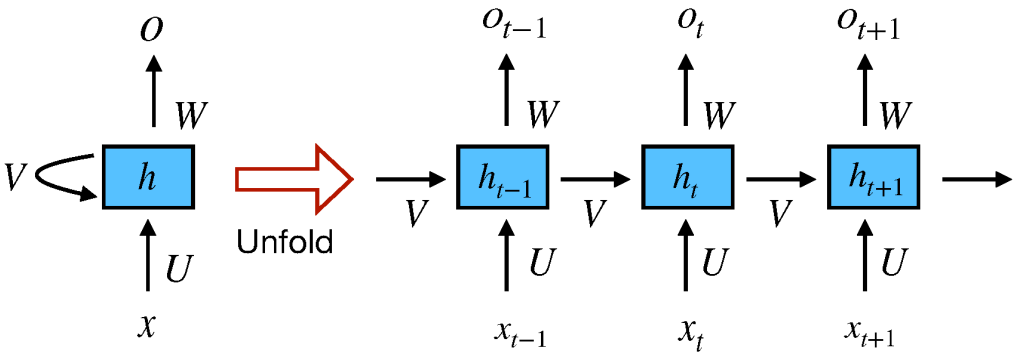
Τα Αναδρομικά Νευρωνικά Δίκτυα ή RNN (Recurrent and Recursive Net), είναι μια οικογένεια Νευρωνικών Δικτύων για την επεξεργασία ακολουθιακών δεδομένων. Τα RNNs χρησιμοποιούν την ιδέα της επεξεργασίας διαδοχικών δεδομένων. Ο όρος "recurrent" αποδίδεται στο ότι εκτελούν την ίδια εργασία σε κάθε στιγμιότυπο της ακολουθίας έτσι ώστε η έξοδος να εξαρτάται από τους προηγούμενους υπολογισμούς και αποτελέσματα. Γενικά, ένα διάνυσμα σταθερού μεγέθους παράγεται για να αναπαραστήσει μια ακολουθία τροφοδοτώντας ένα προς ένα tokens σε μια επαναλαμβανόμενη μονάδα. Κατά κάποιο τρόπο, τα RNNs έχουν «μνήμη» σε σχέση με προηγούμενους υπολογισμούς και χρησιμοποιούν αυτές τις πληροφορίες στην τρέχουσα επεξεργασία. Αυτό το πρότυπο είναι φυσικά κατάλληλο για πολλές εργασίες NLP, όπως η μοντελοποίηση γλώσσας, η αυτόματη μετάφραση, αναγνώριση ομιλίας και η δημιουργία υποτίτλων εικόνων. Αυτό έκανε τα RNNs όλο και πιο δημοφιλή για εφαρμογές NLP τα τελευταία χρόνια [28].

Δεδομένου ότι ένα RNN εκτελεί διαδοχική επεξεργασία μοντελοποίησης μονάδων, όπου οι μονάδες είναι χαρακτήρες, λέξεις ή ακόμα και προτάσεις, με τη σειρά, έχει την ικανότητα να συλλαμβάνει την εγγενή διαδοχική φύση που υπάρχει στη γλώσσα. Οι λέξεις σε μια γλώσσα αναπτύσσουν τη σημασιολογική τους σημασία με βάση τις προηγούμενες λέξεις της πρότασης. Ένα απλό παράδειγμα που δηλώνει αυτό θα ήταν η διαφορά στη σημασία μεταξύ του «dog» και του «hot dog». Τα RNNs είναι ειδικά κατασκευασμένα για τη μοντελοποίηση τέτοιων εξαρτήσεων πλαισίου στη γλώσσα και παρόμοιων εργασιών μοντελοποίησης ακολουθίας. Ένας άλλος παράγοντας που βοηθά την καταλληλότητα των RNNs για εργασίες μοντελοποίησης ακολουθίας έγκειται στην ικανότητά του να μοντελοποιεί μεταβλητό μήκος κειμένου, συμπεριλαμβανομένων πολύ μεγάλων προτάσεων, παραγράφων και ακόμη και εγγράφων. Σε αντίθεση με τα CNNs, τα RNNs έχουν ευέλικτα υπολογιστικά βήματα που παρέχουν καλύτερη ικανότητα μοντελοποίησης και δημιουργούν τη δυνατότητα αποτύπωσης απεριόριστου περιβάλλοντος. Αυτή η ικανότητα χειρισμού εισροών αυθαίρετου μήκους έγινε ένας από τους κύριους λόγους για να χρησιμοποιηθούν τα RNNs σε τέτοιες εργασίες.

Πολλές εργασίες NLP απαιτούν σημασιολογική μοντελοποίηση σε ολόκληρη την πρόταση. Αυτό περιλαμβάνει τη δημιουργία μιας αναπαράστασης της πρότασης σε έναν υπερχώρο σταθερών διαστάσεων. Η ικανότητα του RNN να συνοψίζει προτάσεις οδήγησε στην αυξημένη χρήση τους για εργασίες όπως η μηχανική μετάφραση, όπου ολόκληρη η πρόταση συνοψίζεται σε ένα σταθερό διάνυσμα και στη συνέχεια χαρτογραφείται πίσω στην ακολουθία στόχο μεταβλητού μήκους. Πιο συγκεκριμένες, περιπτώσεις χρήσης περιλαμβάνουν εφαρμογές όπως η κατηγοριοποίηση κειμένου πολλαπλών ετικετών, η πολυτροπική ανάλυση συναισθήματος και η ανίχνευση υποκειμενικότητας.

Τα παραπάνω σημεία επιστρατεύουν μερικούς από τους εστιακούς λόγους που παρακίνησαν τους ερευνητές να επιλέξουν τα RNNs. Ωστόσο, θα ήταν πολύ λάθος να βγάλουμε συμπεράσματα σχετικά με την ανωτερότητα των RNNs έναντι άλλων Βαθιών Νευρωνικών Δικτύων. Ακόμη και σε εργασίες που ταιριάζουν με RNNs, όπως η μοντελοποίηση γλώσσας, τα CNNs πέτυχαν ανταγωνιστικές επιδόσεις έναντι των RNNs. Τόσο τα CNNs όσο και τα RNNs έχουν διαφορετικούς στόχους κατά τη μοντελοποίηση μιας πρότασης. Ενώ τα RNNs προσπαθούν να δημιουργήσουν μια σύνθεση μιας αυθαίρετα μεγάλης πρότασης μαζί με απεριόριστο

περιβάλλον, τα CNNs προσπαθούν να εξαγάγουν τα πιο σημαντικά n-grams. Αν και αποδεικνύονται ένας αποτελεσματικός τρόπος για την καταγραφή χαρακτηριστικών n-grams, που είναι περίπου επαρκής σε ορισμένες εργασίες ταξινόμησης προτάσεων, η ευαισθησία τους στη σειρά λέξεων περιορίζεται τοπικά και οι μακροπρόθεσμες εξαρτήσεις συνήθως αγνοούνται [28].



Εικόνα 2.6: *Απλή δομή Αναδρομικού Νευρωνικού Δικτύου [4]*

Όπως ένα CNN, ένα RNN είναι ένα Νευρωνικό Δίκτυο που είναι εξειδικευμένο για την επεξεργασία ενός πλέγματος τιμών x, δηλαδή μιας ακολουθίας τιμών $x(1), x(2)..., x(t)$. Όπως τα CNNs μπορούν εύκολα να κλιμακωθούν σε εικόνες με μεγάλο πλάτος και ύψος, και ορισμένα RNNs μάλιστα μπορούν να επεξεργάζονται εικόνες μεταβλητού μεγέθους. Τα RNNs μπορούν να κλιμακωθούν σε πολύ μεγαλύτερες ακολουθίες από ό,τι θα ήταν πρακτικό για δίκτυα χωρίς εξειδίκευση βάσει ακολουθίας. Τα περισσότερα RNNs μπορούν επίσης να επεξεργαστούν ακολουθίες μεταβλητού μήκους. Στο πλαίσιο του NLP, τα RNNs βασίζονται κυρίως στο δίκτυο Elman. Ένα δίκτυο Elman είναι ένα δίκτυο τριών επιπέδων (οριζόντια διατεταγμένο ως προς x, y και z), με την προσθήκη ενός συνόλου ενοτήτων περιβάλλοντος. Στην Εικόνα 2.6 μπορούμε να δούμε μια πιο γενική δομή ενός RNN που ξεδιπλώνεται με την πάροδο του χρόνου για να φιλοξενήσει μια ολόκληρη ακολουθία. Στην εικόνα αυτή το x_t αναπαριστά την είσοδο στο δίκτυο τη χρονική στιγμή t και το h_t αναπαριστά την κρυφή κατάσταση (Hidden State) τη χρονική στιγμή t. Ο υπολογισμός του h_t στηρίζεται στον ακόλουθο τύπο:

$$h_t = f(U \times X_t + V \times h_{t-1})$$

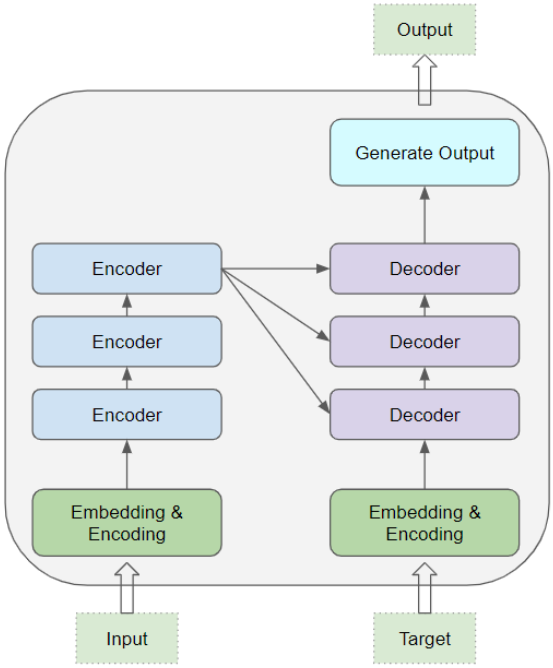
Με αυτόν τον τρόπο, το h_t υπολογίζεται με βάση την τρέχουσα είσοδο και την κρυφή κατάσταση του προηγούμενου χρονικού βήματος. Η συνάρτηση f θεωρείται ότι είναι ένας μη γραμμικός μετασχηματισμός, όπως οι tanh και ReLU, ενώ τα U, V, W είναι τα βάρη που μοιράζονται με την πάροδο του χρόνου. Στο πλαίσιο του NLP, το x_t τυπικά περιλαμβάνει μονοδιάστατα διανύσματα (one-hot), κωδικοποιήσεις (encodings) ή αναπαραστάσεις (embeddings). Μερικές φορές, μπορεί επίσης να είναι αφηρημένες αναπαραστάσεις περιεχομένου ενός κειμένου. Το o_t απεικονίζει την έξοδο του δικτύου, η οποία επίσης συχνά υπόκειται σε μη γραμμικότητα, ειδικά όταν το δίκτυο περιέχει περαιτέρω επίπεδα. Η κρυφή κατάσταση του RNN θεωρείται συνήθως το πιο κρίσιμο στοιχείο του. Όπως αναφέρθηκε προηγουμένως, μπορεί να θεωρηθεί ως το στοιχείο μνήμης του δικτύου που συγκεντρώνει

πληροφορίες από άλλα χρονικά βήματα. Στην πράξη, ωστόσο, τα απλά δίκτυα RNNs υποφέρουν από το διαβόητο πρόβλημα εξαφάνισης των κλίσεων (vanishing gradient problem), το οποίο καθιστά πραγματικά δύσκολο να μάθουμε και να συντονίσουμε τις παραμέτρους των προηγούμενων στρωμάτων στο δίκτυο. Αυτός ο περιορισμός ξεπεράστηκε από διάφορα δίκτυα όπως τα Δίκτυα Βραχυπρόθεσμης Μνήμης (LSTM: Long Short-Term Memory), όπου το δίκτυο επιτρέπει στο σφάλμα να επαναδιαδίδεται μέσω απεριόριστου αριθμού χρονικών βημάτων.

2.2.4 Tranformers

Τα τελευταία χρόνια συναντάμε συχνά τη χρήση της αρχιτεκτονικής των Transformers σε εργασίες NLP. Η αρχιτεκτονική των Transformers χρησιμοποιεί τη λειτουργία του μηχανισμού προσοχής (attention) για να βελτιώσει σημαντικά την απόδοση των μοντέλων Βαθιάς Μάθησης σε εργασίες NLP. Οι αρχιτεκτονική αυτή εισήχθη για πρώτη φορά στην εργασία με τίτλο "Attention Is All You Need", το 2017 και γρήγορα καθιερώθηκε ως η κορυφαία αρχιτεκτονική για τις περισσότερες εφαρμογές δεδομένων κειμένου [29]. Έκτοτε, πολλά έργα, συμπεριλαμβανομένων του μοντέλου BERT της Google και του GPT της OpenAI, έχουν βασιστεί σε αυτό το θεμέλιο και έχουν αποτελέσματα απόδοσης που ξεπερνούν εύκολα τα υπάρχοντα σημεία αναφοράς τελευταίας τεχνολογίας.

Πιο αναλυτικά, η αρχιτεκτονική των Transformers υπερέχει στο χειρισμό δεδομένων κειμένου που δίνονται διαδοχικά. Με λίγα λόγια, είναι αλγόριθμοι που δέχονται μια ακολουθία κειμένου ως είσοδο και παράγουν μια άλλη ακολουθία κειμένου ως έξοδο (πχ μετάφραση από μια γλώσσα σε μια άλλη). Αρχικά, στη δομή τους συναντάμε μια στοίβα κωδικοποιητών (encoders) και μια στοίβα αποκωδικοποιητών (decoders) οι οποίες έχουν τις αντίστοιχες διανυσματικές αναπαραστάσεις για τις αντίστοιχες εισόδους τους. Τέλος, υπάρχει ένα επίπεδο εξόδου για τη δημιουργία της τελικής εξόδου, όπως φαίνεται και στην Εικόνα 2.7. Ο κωδικοποιητής περιέχει το πολύ σημαντικό επίπεδο αυτοπροσοχής (self-attention) που υπολογίζει τη σχέση μεταξύ διαφορετικών λέξεων στην ακολουθία, καθώς και ένα επίπεδο προώθησης (feed-forward). Ο αποκωδικοποιητής εκτός από το βασικό αυτό επίπεδο, έχει ένα δεύτερο επίπεδο προσοχής κωδικοποιητή-αποκωδικοποιητή. Κάθε κωδικοποιητής και αποκωδικοποιητής έχει το δικό του σύνολο βαρών. Ο κωδικοποιητής είναι μια επαναχρησιμοποιούμενη μονάδα που είναι το καθοριστικό στοιχείο όλων των αρχιτεκτονικών του Transformer. Υπάρχουν πολλές παραλλαγές της αρχιτεκτονικής Transformer. Ορισμένες αρχιτεκτονικές δεν έχουν καθόλου αποκωδικοποιητή και βασίζονται μόνο στον κωδικοποιητή. Το κλειδί για την πρωτοποριακή απόδοση του Transformer είναι η χρήση του μηχανισμού προσοχής. Κατά την επεξεργασία μιας λέξης, η λειτουργία του μηχανισμού προσοχής επιτρέπει στο μοντέλο να εστιάζει σε άλλες λέξεις στην είσοδο που σχετίζονται στενά με αυτήν τη λέξη. Για παράδειγμα, στην πρόταση 'The boy is holding a blue ball' η λέξη 'ball' είναι στενά συνδεδεμένη με τις λέξεις 'blue' και 'holding'. Από την μεριά της, η λέξη 'blue' δεν σχετίζεται με την λέξη 'boy' [5].



Εικόνα 2.7: *Βασική Δομή Transformer [5]*

2.3 Επεξεργασία Φυσικής Γλώσσας

Η Επεξεργασία Φυσικής Γλώσσας, γνωστή ως NLP (Natural Language Processing) είναι μια σειρά υπολογιστικών τεχνικών που βασίζεται στη θεωρία για την αυτόματη ανάλυση και αναπαράσταση της ανθρώπινης γλώσσας. Η αρχή του NLP ήταν τη δεκαετία του 1950 ως το σημείο τομής της Τεχνητής Νοημοσύνης και της γλωσσολογίας. Η θεωρητική ανάλυση των γλωσσικών γραμματικών παρείχε μια εκτίμηση της δυσκολίας του προβλήματος ανάλυσης φυσικής γλώσσας και επηρέασε τη δημιουργία του BNF (Backus-Naur Form), το οποίο δημιουργήθηκε (1963) για να καθορίσει μια «γραμματική χωρίς πλαίσιο» (CFG: Context-Free Grammar) και χρησιμοποιείται συνήθως για να αναπαραστήσει τη σύνταξη μιας γλώσσας προγραμματισμού. Η προδιαγραφή BNF μιας γλώσσας είναι ένα σύνολο κανόνων παραγωγής που επικυρώνουν συλλογικά τον κώδικα προγράμματος συντακτικά.

Το πολύ μεγάλο μέγεθος, η απεριόριστη φύση και η ασάφεια της φυσικής γλώσσας οδήγησαν σε δύο προβλήματα κατά τη χρήση τυπικών προσεγγίσεων ανάλυσης που βασίζονταν καθαρά σε συμβολικούς, χειροποίητους κανόνες. Το NLP πρέπει τελικά να εξάγει τη σημασιολογία του κειμένου και τυπικές γραμματικές που καθορίζουν τη σχέση μεταξύ των ενοτήτων του κειμένου και των μερών του λόγου όπως ουσιαστικά, ρήματα και επίθετα. Μπορεί κανείς να επεκτείνει τις γραμματικές για να αντιμετωπίσει τη σημασιολογία της φυσικής γλώσσας επεκτείνοντας σε μεγάλο βαθμό την υποκατηγοριοποίηση, με πρόσθετους κανόνες και περιορισμούς.

Δυστυχώς, οι κανόνες μπορεί τώρα να γίνουν αδιαχείριστα πολυάριθμοι, συχνά αλληλεπιδρώντας απρόβλεπτα, με πιο συχνές διφορούμενες αναλύσεις. Έτσι λοιπόν, η δεκαετία του 1980 οδήγησε σε έναν θεμελιώδη αναπροσανατολισμό, ο οποίος είχε ως αποτέλεσμα τη γέννηση του στατιστικού NLP. Η αξιολόγηση έγινε πιο αυστηρή ενώ οι μέθοδοι Μηχανικής Μάθησης που χρησιμοποιούσαν πιθανότητες χρησιμοποιούνται ευρέως. Έτσι, λιγότε-

ροι, ευρύτεροι κανόνες αντικαθιστούν πολυάριθμους λεπτομερείς κανόνες, με πληροφορίες στατιστικής συχνότητας που αναζητούνται για να αποσαφηνιστούν. Άλλες προσεγγίσεις δημιουργούν πιθανολογικούς «κανόνες» παρόμοιους με αλγόριθμους Μηχανικής Μάθησης που δημιουργούν δέντρα απόφασης από δεδομένα χαρακτηριστικών-διανυσμάτων. Σε κάθε περίπτωση, ένας στατιστικός αναλυτής καθορίζει την πιο πιθανή ανάλυση μιας πρότασης ή φράσης. Οι στατιστικές προσεγγίσεις δίνουν καλά αποτελέσματα στην πράξη απλώς και μόνο επειδή, μαθαίνοντας με άφθονα πραγματικά δεδομένα, χρησιμοποιούν τις πιο συνηθισμένες περιπτώσεις: όσο πιο άφθονα και αντιπροσωπευτικά είναι τα δεδομένα, τόσο καλύτερα γίνονται.

Νευρωνικά Δίκτυα Βαθιάς Μάθησης έχουν χρησιμοποιηθεί με μεγάλη επιτυχία στο NLP. Προτού οι αλγόριθμοι αυτοί επικρατήσουν στο NLP χρησιμοποιούνταν άλλες τεχνικές Μηχανικής Μάθησης. Μερικές κοινές μέθοδοι Μηχανικής Μάθησης που χρησιμοποιούνται σε εργασίες NLP συνοψίζονται παρακάτω:

Support Vector Machines (SVMs)

Το SVM είναι μια διακριτική προσέγγιση μάθησης για ταξινομήσεις εισροών (π.χ. λέξεις) σε κατηγορίες (π.χ. μέρη ομιλίας) με βάση ένα σύνολο χαρακτηριστικών. Η είσοδος μπορεί να μετασχηματιστεί μαθηματικά χρησιμοποιώντας μια «συνάρτηση πυρήνα» που επιτρέπει τον γραμμικό διαχωρισμό των σημείων δεδομένων από διαφορετικές κατηγορίες. Δηλαδή, στην απλούστερη περίπτωση δύο χαρακτηριστικών, μια ευθεία γραμμή θα τους χώριζε σε μια γραφική παράσταση X–Y: στη γενική περίπτωση N-χαρακτηριστικών, ο διαχωριστής θα είναι ένα $(N - 1)$ υπερεπίπεδο. Η πιο κοινή συνάρτηση πυρήνα που χρησιμοποιείται είναι η Gaussian (η βάση της κανονικής κατανομής στη στατιστική). Στη διαδικασία διαχωρισμού επιλέγεται ένα υποσύνολο των δεδομένων εκπαίδευσης (τα διανύσματα υποστήριξης—σημεία δεδομένων πλησιέστερα στο υπερεπίπεδο) που διαφοροποιεί καλύτερα τις κατηγορίες. Το διαχωριστικό υπερεπίπεδο μεγιστοποιεί την απόσταση για να υποστηρίξει διανύσματα από κάθε κατηγορία.

Hidden Markov Models (HMMs)

Ένα HMM είναι ένα σύστημα όπου μια μεταβλητή μπορεί να εναλλάσσεται (με ποικίλες πιθανότητες) μεταξύ πολλών καταστάσεων, δημιουργώντας ένα από τα πολλά πιθανά σύμβολα εξόδου με κάθε εναλλαγή. Τα σύνολα των πιθανών καταστάσεων και των μοναδικών συμβόλων μπορεί να είναι μεγάλα, αλλά πεπερασμένα και γνωστά. Μπορούμε να παρατηρήσουμε τις εξόδους, αλλά τα εσωτερικά στοιχεία του συστήματος (δηλαδή, οι πιθανότητες μεταγωγής κατάστασης και οι πιθανότητες εξόδου) είναι κρυμμένα. Τα προβλήματα που πρέπει να επιλυθούν είναι το πως θα γίνει η συμπερασματολογία (inference), η εκπαίδευση (training) καθώς και η αντιστοίχιση μοτίβων (pattern matching), που είναι η εύρεση ακολουθιών εναλλαγής κατάστασης η οποία είναι πιο πιθανό να έχει δημιουργήσει μια συγκεκριμένη ακολουθία συμβόλων εξόδου.

Τα HMMs χρησιμοποιούν ένα παραγωγικό μοντέλο. Για την επίλυση αυτών των προβλημάτων, ένα HMM χρησιμοποιεί δύο απλοποιητικές υποθέσεις (οι οποίες ισχύουν για πολλά φαινόμενα της πραγματικής ζωής):

1. Η πιθανότητα μετάβασης σε μια νέα κατάσταση (ή πίσω στην ίδια κατάσταση) εξαρτάται από τις προηγούμενες N καταστάσεις. Στην απλούστερη περίπτωση «πρώτης τάξης» (N=1), αυτή η πιθανότητα καθορίζεται μόνο από την τρέχουσα κατάσταση.

2. Η πιθανότητα δημιουργίας μιας συγκεκριμένης εξόδου σε μια συγκεκριμένη κατάσταση εξαρτάται μόνο από αυτή την κατάσταση [28].

Αυτές οι παραδοχές επιτρέπουν τον υπολογισμό της πιθανότητας μιας δεδομένης ακολουθίας μεταγωγής καταστάσεων (και μιας αντίστοιχης ακολουθίας παρατήρησης εξόδου) με απλό πολλαπλασιασμό των επιμέρους πιθανοτήτων. Θεωρητικά, τα HMMs θα μπορούσαν να επεκταθούν σε ένα πολυπαραγοντικό σενάριο, αλλά το πρόβλημα της εκπαίδευσης μπορεί τώρα να γίνει δυσεπίλυτο. Στην πράξη, οι εφαρμογές πολλαπλών μεταβλητών των HMMs χρησιμοποιούν μεμονωμένες, τεχνητές μεταβλητές που είναι μοναδικά καθορισμένες σύνθετες συνθέσεις υπαρχουσών κατηγορικών μεταβλητών: τέτοιες προσεγγίσεις απαιτούν πολύ περισσότερα δεδομένα εκπαίδευσης.

Conditional Random Fields (CRFs)

Τα CRFs είναι μια οικογένεια διακριτικών μοντέλων. Τα συνηθέστερα CRFs, γραμμικής αλυσίδας μοιάζουν με HMMs καθώς κάθε επόμενη κατάσταση εξαρτάται από την τρέχουσα κατάσταση (γραμμική αλυσίδα εξάρτησης). Συγκεκριμένα, τα CRFs χρησιμοποιούνται για την πρόβλεψη των μεταβλητών κατάστασης (Ys) με βάση τις παρατηρούμενες μεταβλητές (Xs). Τα CRF ταιριάζουν καλύτερα σε διαδοχικά πολυμεταβλητά δεδομένα από τα HMM, ενώ παρότι απαιτεί περισσότερα παραδείγματα δεδομένων από ένα μονομεταβλητό HMM, εξακολουθεί να είναι επιλύσιμο.

n-grams

Ένα n-gram είναι μια ακολουθία από n στοιχεία όπως γράμματα, λέξεις ή φωνήματα. Για παράδειγμα, το "San Frasinsco" είναι ένα n-gram με το n να ισούται με το 2. Γνωρίζουμε ότι ορισμένα ζεύγη αντικειμένων (ή τριάδες, ή τετράδες, κ.λ.π.) είναι πιθανό να εμφανίζονται πολύ πιο συχνά από άλλα. Για παράδειγμα, στις αγγλικές λέξεις, το U ακολουθεί πάντα το Q και το αρχικό T δεν ακολουθείται ποτέ από το K. Με επαρκή δεδομένα, μπορούμε να υπολογίσουμε δεδομένα κατανομής συχνότητας για όλα τα n-grams που εμφανίζονται σε αυτά τα δεδομένα. Τα n-grams είναι ένα είδος μοντέλου Markov πολλαπλών τάξεων: η πιθανότητα ενός συγκεκριμένου στοιχείου στη n-οστή θέση εξαρτάται από τα προηγούμενα $n - 1$ στοιχεία και μπορεί να υπολογιστεί από δεδομένα. Αφού υπολογιστούν, τα δεδομένα n-gram μπορούν να χρησιμοποιηθούν για διάφορους σκοπούς όπως προτεινόμενη αυτόματη συμπλήρωση λέξεων και φράσεων, διόρθωση ορθογραφίας ή και αναγνώριση ομιλίας.

Η Google έχει υπολογίσει δεδομένα λέξης n-gram ($n <= 5$) από τα δεδομένα ιστού της και από το και τα έχει διαθέσει ελεύθερα. Τα δεδομένα n-gram είναι ογκώδη, συγκεκριμένα η βάση δεδομένων n-gram της Google απαιτεί όλο και περισσότερο χώρο. Ειδικές δομές δεδομένων, που ονομάζονται ευρετήρια n-gram, επιταχύνουν την αναζήτηση τέτοιων δεδομένων. Οι ταξινομητές που βασίζονται σε n-gram αξιοποιούν το ακατέργαστο εκπαιδευτικό κείμενο χωρίς ρητές γλωσσικές γνώσεις, ενώ αποδίδουν καλές επιδόσεις, αφήνουν περιθώρια

βελτίωσης και επομένως συμπληρώνονται με άλλες προσεγγίσεις [30].

Όλες αυτές οι παραπάνω τεχνικές ήταν η αρχή της εφαρμογής της Μηχανικής Μάθησης σε εργασίες NLP. Για δεκαετίες, οι προσεγγίσεις αυτές βασίζονταν σε ρηχά μοντέλα που εκπαιδεύονται σε πολύ υψηλές διαστάσεις και αραιά χαρακτηριστικά. Τα τελευταία χρόνια, τα Νευρωνικά Δίκτυα που βασίζονται σε αναπαραστάσεις πυκνών διανυσμάτων έχουν παράγει ανώτερα αποτελέσματα σε διάφορες εργασίες NLP. Αυτή η τάση πυροδοτείται από την επιτυχία της διανυσματικής αναπαράστασης λέξεων (word embeddings) και των μεθόδων Βαθιάς Μάθησης [28].

2.3.1 Τεχνικές αναπαράστασης λέξεων

Το NLP στηριζόμενο σε στατιστικές τεχνικές έχει αναδειχθεί ως η κύρια επιλογή για τη μοντελοποίηση σύνθετων εργασιών φυσικής γλώσσας. Στην αρχή του, συχνά υπέφερε από την περιβόητη κατάρα της διαστατικότητας κατά την μάθηση κοινών συναρτήσεων πιθανότητας γλωσσικών μοντέλων. Αυτό οδήγησε στο κίνητρο της εκμάθησης κατανεμημένων αναπαραστάσεων λέξεων που υπάρχουν σε χώρο χαμηλών διαστάσεων.

Διανυσματικές αναπαραστάσεις λέξεων

Τα Διανύσματα Κατανομής (Distributional Vectors) ή διανύσματα αναπαράστασης λέξεων (word embeddings) είναι η αναπαράσταση λέξεων για ανάλυση κειμένου με τη μορφή ενός διανύσματος πραγματικής αξίας το οποίο κωδικοποιεί τη σημασία της λέξης. Λέξεις λοιπόν που βρίσκονται πιο κοντά μεταξύ τους στον διανυσματικό χώρο αναμένεται να είναι παρόμοιες σε έννοια ακολουθώντας ουσιαστικά την υπόθεση κατανομής, σύμφωνα με την οποία λέξεις με παρόμοια σημασία τείνουν να εμφανίζονται σε παρόμοιο πλαίσιο κειμένου. Τα διανύσματα αυτά επομένως προσπαθούν να συλλάβουν τα χαρακτηριστικά των γειτόνων κάθε λέξης. Το κύριο πλεονέκτημα των Διανυσμάτων Κατανομής είναι ότι καταγράφουν την ομοιότητα μεταξύ των λέξεων. Η μέτρηση της ομοιότητας μεταξύ των διανυσμάτων είναι δυνατή, χρησιμοποιώντας μέτρα όπως η ομοιότητα συνημιτόνου. Τα διανύσματα αναπαράστασης λέξεων χρησιμοποιούνται συχνά ως το πρώτο επίπεδο επεξεργασίας δεδομένων σε ένα μοντέλο Βαθιάς Μάθησης. Αυτά, συνήθως προεκπαιδεύονται με τη βελτιστοποίηση ενός βοηθητικού στόχου σε ένα μεγάλο σύνολο δεδομένων χωρίς ετικέτα. Για παράδειγμα, εκπαιδεύονται στη πρόβλεψη μιας λέξης σε ένα κείμενο ή πρόταση, όπου τα διανύσματα αναπαράστασης λέξεων που μαθαίνονται μπορούν να συλλάβουν γενικά συντακτικές και σημασιολογικές πληροφορίες. Έτσι, αυτές οι αναπαραστάσεις έχουν αποδειχθεί αποτελεσματικές στην καταγραφή της ομοιότητας περιβάλλοντος, των αναλογιών και λόγω της μικρότερης διάστασής τους, είναι γρήγορες και αποτελεσματικές στον υπολογισμό βασικών εργασιών NLP.

Με τα χρόνια, τα μοντέλα που δημιουργούσαν τέτοια διανύσματα ήταν ρηχά Νευρωνικά Δίκτυα έτσι, δεν υπήρξε ανάγκη από Βαθιά Δίκτυα για τη δημιουργία καλών Διανυσμάτων Κατανομής. Ωστόσο, τα μοντέλα NLP που βασίζονται σε Βαθιά Μάθηση αναπαριστούν πάντα τις λέξεις, τις φράσεις και ακόμη και τις προτάσεις χρησιμοποιώντας αυτά τα διανύσματα. Αυτή είναι στην πραγματικότητα μια σημαντική διαφορά μεταξύ των παραδοσιακών μοντέλων που βασίζονται στον αριθμό λέξεων και των μοντέλων που βασίζονται σε Βαθιά Μάθηση. Οι

ενσωματώσεις λέξεων είναι υπεύθυνες για αποτελέσματα αιχμής σε ένα ευρύ φάσμα εργασιών NLP.

Word2vec

Το Word2vec είναι μια μέθοδος για την κατασκευή ενός διανύσματος αναπαράστασης λέξης. Τα διανύσματα αναπαράστασης λέξεων έφεραν επανάσταση και προτάθηκαν μαζί με αυτά δύο νέες τεχνικές για τη δημιουργία τους, με βάση μεγάλα σύνολα δεδομένων. Και οι δύο τεχνικές είναι παρόμοιες μεταξύ τους. Η τεχνική CBOW υπολογίζει την υπό όρους πιθανότητα μιας λέξης-στόχου, δεδομένων των λέξεων που την περιβάλλουν (context words) σε ένα παράθυρο περιβάλλοντος (context window) μεγέθους k. Το παράθυρο περιβάλλοντος είναι ο αριθμός των λέξεων που πρέπει να προβλεφθούν προκειμένου να οριστεί το πλαίσιο της λέξης-στόχου. Το μοντέλο Skip-gram κάνει το ακριβώς αντίθετο από το μοντέλο CBOW, έτσι δεδομένης μια λέξης-στόχου προβλέπει τις λέξεις που την περιβάλουν. Οι λέξεις που περιβάλλουν την λέξη-στόχο ή αλλιώς λέξεις πλαισίου θεωρείται ότι βρίσκονται συμμετρικά με τις λέξεις-στόχους σε απόσταση ίση με το μέγεθος του παραθύρου και προς τις δύο κατευθύνσεις. Σε διαδικασίες χωρίς επίβλεψη, η ιδιότητα διανύσματος αναπαράστασης λέξεων καθορίζεται από την ακρίβεια της πρόβλεψης. Καθώς η διάσταση του διανύσματος αναπαράστασης λέξεων αυξάνεται, η ακρίβεια της πρόβλεψης αυξάνεται επίσης, μέχρι να συγκλίνει σε κάποιο σημείο, το οποίο θεωρείται η βέλτιστη διάσταση του διανύσματος και είναι το μικρότερο χωρίς να επηρεάζεται η ακρίβεια.

Διανυσματικές αναπαραστάσεις χαρακτήρων

Οι διανυσματικές αναπαραστάσεις λέξεων είναι σε θέση να εξάγουν συντακτικές και σημασιολογικές πληροφορίες, ωστόσο για εργασίες όπως η προσθήκη ετικετών, οι μορφολογικές πληροφορίες και οι πληροφορίες σχήματος εντός της λέξης μπορούν επίσης να είναι πολύ χρήσιμες. Γενικά, η δημιουργία συστημάτων κατανόησης φυσικής γλώσσας σε επίπεδο χαρακτήρων έχει προσελκύσει ορισμένη ερευνητική προσοχή. Μάλιστα με ανάλυση σε αυτό το επίπεδο προκύπτουν καλύτερα αποτελέσματα σε μορφολογικά πλούσιες γλώσσες σε ορισμένες εργασίες NLP.

Ένα συνηθισμένο φαινόμενο για γλώσσες με μεγάλο λεξιλόγιο είναι το ζήτημα της άγνωστης λέξης ή το ζήτημα της λέξης εκτός λεξιλογίου (out-of-vocabulary word issue). Οι διανυσματικές αναπαραστάσεις χαρακτήρων (character embeddings) ασχολούνται φυσικά με αυτό, καθώς κάθε λέξη δεν θεωρείται τίποτα περισσότερο από μια σύνθεση μεμονωμένων γραμμάτων. Σε γλώσσες όπου το κείμενο δεν αποτελείται από χωριστές λέξεις αλλά μεμονωμένους χαρακτήρες και η σημασιολογική σημασία των λέξεων αντιστοιχεί στους συνθετικούς χαρακτήρες του (όπως τα κινέζικα), η οικοδόμηση συστημάτων σε επίπεδο χαρακτήρων είναι μια φυσική επιλογή για την αποφυγή τμηματοποίησης λέξεων (word segmentation). Έτσι, στις εργασίες που χρησιμοποιούνται εφαρμογές Βαθιάς Μάθησης σε τέτοιες γλώσσες η τάση είναι να προτιμώνται διανυσματικές αναπαραστάσεις χαρακτήρων από τις διανυσματικές αναπαραστάσεις λέξεων.

Διανυσματικές αναπαραστάσεις λέξεων με βάση τα συμφραζόμενα

Η ποιότητα των διανυσματικών αναπαραστάσεων λέξεων γενικά μετράται από την ικανότητά τους να κωδικοποιούν συντακτικές πληροφορίες και να χειρίζονται πολυσημασιακή συμπεριφορά. Αυτές οι ιδιότητες έχουν ως αποτέλεσμα βελτιωμένες σημασιολογικά αναπαραστάσεις λέξεων. Πρόσφατες προσεγγίσεις σε αυτόν τον τομέα κωδικοποιούν τέτοιες πληροφορίες στα διανύσματα τους, αξιοποιώντας το γενικό πλαίσιο. Αυτές οι μέθοδοι παρέχουν βαθύτερα δίκτυα που υπολογίζουν τις αναπαραστάσεις λέξεων ως συνάρτηση του πλαισίου τους.

Οι παραδοσιακές μέθοδοι δημιουργίας διανυσματικών αναπαραστάσεων λέξεων, όπως το Word2Vec λαμβάνουν υπόψη όλες τις προτάσεις στις οποίες υπάρχει μια λέξη, προκειμένου να δημιουργήσουν μια καθολική διανυσματική αναπαράσταση αυτής της λέξης. Ωστόσο, μια λέξη μπορεί να έχει εντελώς διαφορετικές έννοιες ή έννοιες στα συμφραζόμενα. Για παράδειγμα, ας εξετάσουμε αυτές τις δύο προτάσεις: 1) The bank will not be accepting cash on Saturdays. 2) The river overflowed the bank. Οι έννοιες της λέξης "bank" είναι διαφορετικές σε αυτές τις δύο προτάσεις ανάλογα με το περιεχόμενό της αφού στην πρώτη πρόταση η σημασία της λέξης είναι τράπεζα ενώ στη δεύτερη πρόταση είναι όχθη. Για αυτόν ακριβώς τον λόγο είναι χρήσιμο να υπάρχουν δύο διαφορετικές διανυσματικές αναπαραστάσεις της λέξης "bank" με βάση τις δύο διαφορετικές έννοιες των λέξεων. Η νέα κατηγορία μοντέλων υιοθετεί αυτό το σκεπτικό αποκλίνοντας από την έννοια των καθολικών αναπαραστάσεων λέξεων και προτείνοντας αντ' αυτού Αναπαραστάσεις Διανυσμάτων λέξεων με βάση τα συμφραζόμενα (contextualized word embeddings) [26].

2.3.2 Βασικές τεχνικές προεπεξεργασίας δεδομένων εισόδου

Όλα όσα εκφράζουμε, είτε προφορικά είτε γραπτά, φέρουν τεράστιες ποσότητες πληροφοριών. Προκειμένου να βρεθεί ένας τρόπος ανάλυσης των πληροφοριών αυτών το NLP δίνει στις μηχανές τη δυνατότητα να διαβάζουν, να κατανοούν και να αντλούν νόημα από ανθρώπινες γλώσσες μέσα από πολλές και διαφορετικές τεχνικές. Τα κύρια μειονεκτήματα που αντιμετωπίζουμε αυτές τις μέρες με το NLP σχετίζονται με το γεγονός ότι η γλώσσα είναι πολύ δύσκολη. Η διαδικασία της κατανόησης και του χειρισμού της γλώσσας είναι εξαιρετικά περίπλοκη και γι' αυτό το λόγο είναι σύνηθες να χρησιμοποιούνται διαφορετικές τεχνικές για να χειρίζονται διαφορετικές προκλήσεις πριν να δέσουν τα πάντα μεταξύ τους. Ας συνοψίσουμε και εξηγήσουμε μερικούς από τους πιο συχνά χρησιμοποιούμενους αλγόριθμους στο NLP κατά τον ορισμό του λεξιλογίου των όρων:

Bag of Words

Το Bag of Words είναι ένα ευρέως χρησιμοποιούμενο μοντέλο που επιτρέπει την μέτρηση όλων των λέξεων σε ένα κομμάτι κειμένου. Βασικά δημιουργεί μια μήτρα συχνότητας εμφάνισης για μια πρόταση ή ένα κείμενο, αδιαφορώντας για τη γραμματική και τη σειρά των λέξεων. Αυτές οι συχνότητες ή εμφανίσεις λέξεων χρησιμοποιούνται στη συνέχεια ως χαρακτηριστικά για την εκπαίδευση ενός ταξινομητή. Σε αυτήν την τεχνική παράγεται ένας πίνακας στον οποίο σημειώνεται το πόσες φορές εμφανίζεται κάθε λέξη στο κείμενο το οποίο

αναλύουμε. Η συχνότητα εμφάνισης κάθε λέξης ονομάζεται αλλιώς ως βάρος της λέξης αυτής. Ένα παράδειγμα από την φράση "Words are flowing out like endless rain into a paper cup, They slither while they pass, they slip away across the universe" στίχους των Beatles αναλύεται στην Εικόνα 2.8.

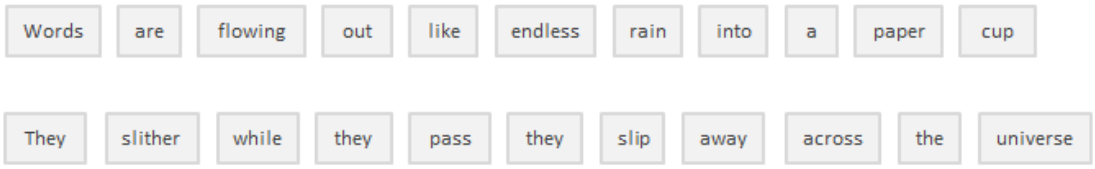
	words	rain	a	paper	they	slip	the	universe	...
Words are flowing out like endless rain into a paper cup,	1	1	1	1	0	0	0	0	...
They slither while they pass, they slip away across the universe	0	0	0	0	3	1	1	1	...

Εικόνα 2.8: *Αποτέλεσμα του Bag of Words για φράση σε στίχους των Beatles*

Αυτή η προσέγγιση ωστόσο μπορεί να ενέχει πολλά μειονεκτήματα, όπως η απουσία σημασιολογικού νοήματος και πλαισίου, ή το γεγονός ότι λέξεις τερματισμού (stop words), όπως οι «το» ή «α» προσθέτουν θόρυβο στην ανάλυση και ορισμένες λέξεις δεν σταθμίζονται ανάλογα. Για να λυθεί αυτό το πρόβλημα, μια προσέγγιση είναι να αναπροσαρμόσουμε τη συχνότητα των λέξεων ανάλογα με τη συχνότητα εμφάνισης τους σε όλα τα κείμενα (όχι μόνο σε αυτό που αναλύουμε), έτσι ώστε οι βαθμολογίες για συχνές λέξεις όπως το "το", που είναι επίσης συχνές σε άλλα κείμενα, τιμωρούνται όπως λέγεται δηλαδή σημειώνονται σαν λέξεις που δεν επηρεάζουν τόσο πολύ το νόημα και το ύφος του κειμένου. Αυτή η προσέγγιση βαθμολόγησης ονομάζεται «Συχνότητα όρου — Αντίστροφη συχνότητα εγγράφων» (TFIDF: Term Frequency — Inverse Document Frequency) και βελτιώνει το σύνολο των λέξεων κατά βάρη. Μέσω του TFIDF οι συχνοί όροι στο κείμενο «ανταμείβονται» (όπως η λέξη «αυτοί» στο παράδειγμά μας), αλλά επίσης «τιμωρούνται» εάν αυτοί οι όροι είναι συχνοί σε άλλα κείμενα που συμπεριλαμβάνουμε και στον αλγόριθμο. Αντίθετα, αυτή η μέθοδος αναδεικνύει και «ανταμείβει» μοναδικούς ή σπάνιους όρους λαμβάνοντας υπόψη όλα τα κείμενα. Ωστόσο, αυτή η προσέγγιση δεν έχει ακόμη κανένα πλαίσιο ούτε σημασιολογία.

Τμηματοποίηση (Tokenization)

Η Τμηματοποίηση είναι η διαδικασία διαχωρισμού του τρέχοντος κειμένου σε κομμάτια που ονομάζονται tokens, ενώ ταυτόχρονα πραγματοποιείται και η αφαίρεση ορισμένων χαρακτήρων από το τελικό κείμενο που θα αναλυθεί, όπως τα σημεία στίξης. Ακολουθώντας το παράδειγμά μας παραπάνω, το αποτέλεσμα της Τμηματοποίησης φαίνεται στην Εικόνα 2.9 .



Εικόνα 2.9: *Αποτέλεσμα της τμηματοποίησης για τους παραπάνω στίχους των Beatles*

Μπορεί να φαίνεται πολύ βασικό σε μια γλώσσα το να διαχωρίζονται οι λέξεις μεταξύ τους με ένα κενό διάστημα (τμηματικές γλώσσες), ωστόσο δεν συμπεριφέρονται όλες οι γλώσσες

το ίδιο όταν η μόνη προεπεξεργασία που πραγματοποιείται είναι ο διαχωρισμός αυτός, αφού είναι προφανές ότι τα κενά διαστήματα από μόνα τους δεν επαρκούν στο να παραχθούν τα κατάλληλα διακριτικά. Ο διαχωρισμός μέσω κενών διαστημάτων, δηλαδή ο διαχωρισμός που οδηγεί σε tokens που αποτελούν ένα σύνολο χαρακτήρων από ένα κενό και μέχρι το επόμενο, μπορεί να διαλύσει αυτό που θα πρέπει να θεωρείται ως ένα ενιαίο token, όπως στην περίπτωση ορισμένων ονομάτων (π.χ. Σαν Φρανσίσκο ή Νέα Υόρκη) ή ξένων φράσεων. Η τμηματοποίηση μπορεί επίσης να αφαιρέσει τα σημεία στίξης, διευκολύνοντας τη διαδρομή προς μια σωστή κατάτμηση λέξεων αλλά και πυροδοτώντας πιθανές επιπλοκές. Προκειμένου να αποφευχθούν πολλά προβλήματα η τμηματοποίηση αποκτά πιο περίπλοκες τεχνικές και έτσι οι λέξεις ενός κειμένου με τη σειρά τους διαχωρίζονται σε "υπολέξεις", οι οποίες στη συνέχεια μετατρέπονται σε αναγνωριστικά (ids) μέσω ενός πίνακα αναζήτησης. Τα αναγνωριστικά αυτά λοιπόν αποτελούν την είσοδο στο εκάστοτε μοντέλο και αναπαρίστανται σε τύπους αριθμών όπως int και float.

Αφαίρεση λέξεων τερματισμού (Stop words removal)

Αυτή η τεχνική περιλαμβάνει την απαλλαγή από άρθρα, αντωνυμίες και προθέσεις κοινής γλώσσας όπως "and", "the" ή "to" στα αγγλικά. Σε αυτή τη διαδικασία ορισμένες πολύ συνηθισμένες λέξεις που φαίνεται να παρέχουν μικρή ή και καθόλου αξία στον NLP στόχο φιλτράρονται και αποκλείονται από το προς επεξεργασία κειμένου, καταργώντας, ως εκ τούτου, ευρέως διαδεδομένους και συχνούς όρους που δεν είναι χρήσιμοι για το εκάστοτε κείμενο του οποίου η σημασία μελετάται. Οι λέξεις τερματισμού μπορούν να αγνοηθούν με ασφάλεια πραγματοποιώντας μια αναζήτηση σε μια προκαθορισμένη λίστα λέξεων-κλειδιών, ελευθερώνοντας χώρο στη βάση δεδομένων και βελτιώνοντας κατά αυτόν τον τρόπο τον χρόνο επεξεργασίας.

Αναχαίτιση (Stemming)

Η διαδικασία αυτή είναι ουσιαστικά η διαδικασία κοπής του τέλους ή της αρχής των λέξεων με σκοπό την αφαίρεση προσθηκών. Οι προσθήκες που επισυνάπτονται στην αρχή της λέξης ονομάζονται προθήματα (π.χ. η λέξη «astro» στη λέξη «astrobiology») και αυτές που επισυνάπτονται στο τέλος της λέξης ονομάζονται επιθήματα (π.χ. «ful» στη λέξη «helpful»). Το πρόβλημα είναι ότι οι προσθήκες μπορούν να δημιουργήσουν ή να επεκτείνουν νέους τύπους της ίδιας λέξης, ή ακόμη και να δημιουργήσουν από μόνα τους νέες λέξεις. Στα αγγλικά, τα προθήματα είναι πάντα παράγωγα (η προσθήκη δημιουργεί μια νέα λέξη όπως στο παράδειγμα του προθήματος "eco" στη λέξη "ecosystem"), αλλά τα επιθήματα μπορούν να είναι παράγωγα (η προσθήκη δημιουργεί μια νέα λέξη όπως στο παράδειγμα της προσθήκης «ist» στη λέξη «guitarist») ή κλίση (η προσθήκη δημιουργεί μια νέα μορφή λέξης όπως στο παράδειγμα της προσθήκης «er» στη λέξη «faster»).

Μια πιθανή προσέγγιση είναι να εξετάσουμε μια λίστα κοινών προσθηκών και κανόνων και να εκτελέσουμε αναχαίτιση βάσει αυτών, αλλά φυσικά αυτή η προσέγγιση παρουσιάζει περιορισμούς. Δεδομένου ότι οι αλγόριθμοι που πραγματοποιούν την διαδικασία αναχαίτισης χρησιμοποιούν αλγοριθμικές προσεγγίσεις, το αποτέλεσμα της αναχαίτισης διαδικασίας μπορεί να μην είναι μια πραγματική λέξη ή ακόμη και να είναι μια λέξη με άλλο νόημα

τελικά (γεγονός που οδηγεί σε πρόταση με άλλο νόημα). Για την αντιστάθμιση αυτού, γίνεται επεξεργασία αυτών των προκαθορισμένων μεθόδων με την προσθήκη ή την αφαίρεση προσθηκών και κανόνων, ωστόσο πρέπει να ληφθεί υπόψη ότι ενδέχεται να γίνεται βελτίωση της απόδοσης σε μια περιοχή ενώ γίνεται υποβάθμιση σε μια άλλη. Ωστόσο παρά τους σοβαρούς περιορισμούς της αναχαίτισης συνεχίζουμε να τη χρησιμοποιούμε αφού μας προσφέρει άλλες ευκολίες.

Λημματοποίηση (Lemmatization)

Η διαδικασία αυτή έχει στόχο να φτάσει μια λέξη στη "βασική της μορφή" και να ομαδοποιήσει διαφορετικές μορφές της ίδιας λέξης. Για παράδειγμα, τα ρήματα σε παρελθόντα χρόνο αλλάζουν σε παρόν (π.χ. το "went" αλλάζει σε "go") και τα συνώνυμα ενοποιούνται (π.χ. το "best" αλλάζει σε "good"), επομένως τυποποιούνται λέξεις με παρόμοια σημασία με τη ρίζα τους. Παρόλο που φαίνεται στενά συνδεδεμένη με τη διαδικασία της ρίζας, η διαδικασία της λημματοποίησης χρησιμοποιεί μια διαφορετική προσέγγιση για να φτάσει στις ρίζες των λέξεων. Η διαδικασία της λημματοποίησης επιστρέφει τις λέξεις στη μορφή του λεξικού τους (γνωστή ως λήμμα) για την οποία απαιτεί λεπτομερή λεξικό στα οποία ο αλγόριθμος μπορεί να εξετάσει και να συνδέσει λέξεις με τα αντίστοιχα λήμματα τους. Για παράδειγμα, οι λέξεις «running», «runs» και «ran» είναι όλες οι μορφές της λέξης «run», οπότε το «τρέξιμο» είναι το λήμμα όλων των προηγούμενων λέξεων.

Η λημματοποίηση λαμβάνει επίσης υπόψη το πλαίσιο της λέξης για να λύσει άλλα προβλήματα όπως η αποσαφήνιση, που σημαίνει ότι μπορεί να κάνει διάκριση μεταξύ πανομοιότυπων λέξεων που έχουν διαφορετική σημασία ανάλογα με το συγκεκριμένο πλαίσιο. Σκεφτείτε λέξεις όπως «bat» (που μπορεί να αντιστοιχεί στο ζώο ή στο μεταλλικό/ξύλινο ρόπαλο που χρησιμοποιείται στο μπέιζμπολ) ή «bank» (που αντιστοιχεί στο χρηματοπιστωτικό ίδρυμα ή στη γη δίπλα σε ένα υδάτινο σώμα). Παρέχοντας μια παράμετρο τμήματος του λόγου σε μια λέξη (είτε είναι ουσιαστικό, ρήμα κ.λ.π.) είναι δυνατό να οριστεί ένας ρόλος για αυτήν τη λέξη στην πρόταση και να αφαιρέσετε την αποσαφήνιση.

Η λημματοποίηση είναι μια εργασία που απαιτεί πολύ περισσότερους πόρους από την εκτέλεση μιας διαδικασίας αναχαίτισης. Ταυτόχρονα, δεδομένου ότι απαιτεί περισσότερη γνώση σχετικά με τη δομή της γλώσσας από μια προσέγγιση βασικής προσέγγισης, απαιτεί περισσότερη υπολογιστική ισχύ από τη δημιουργία ή την προσαρμογή ενός βασικού αλγορίθμου.

Μοντελοποίηση Θεμάτων (Topic Modeling)

Η συγκεκριμένη μέθοδος χρησιμοποιείται για την αποκάλυψη κρυφών δομών σε σύνολα κειμένων ή εγγράφων. Ουσιαστικά συγκεντρώνει κείμενα για να ανακαλύψει κρυμμένα θέματα με βάση το περιεχόμενό τους, επεξεργάζεται μεμονωμένες λέξεις και τους αποδίδει τιμές με βάση την κατανομή τους. Αυτή η τεχνική βασίζεται στην υπόθεση ότι κάθε έγγραφο αποτελείται από ένα μείγμα θεμάτων και ότι κάθε θέμα αποτελείται από ένα σύνολο λέξεων, πράγμα που σημαίνει ότι αν μπορέσουμε να εντοπίσουμε αυτά τα κρυμμένα θέματα, μπορούμε να ξεκλειδώσουμε το νόημα των κειμένων μας. Από το σύμπαν των τεχνικών μοντελοποίησης θεμάτων, η Latent Dirichlet Allocation (LDA) είναι ίσως η πιο συχνά χρη-

σιμοποιούμενη. Αυτός ο σχετικά νέος αλγόριθμος (που εφευρέθηκε πριν από λιγότερα από 20 χρόνια) λειτουργεί ως μέθοδος μάθησης χωρίς επίβλεψη που ανακαλύπτει διαφορετικά θέματα κάτω από μια συλλογή εγγράφων. Σε μεθόδους Μη Επιβλεπόμενης Μάθησης όπως αυτή, δεν υπάρχει μεταβλητή εξόδου που να καθοδηγεί τη διαδικασία εκμάθησης και τα δεδομένα διερευνώνται από αλγόριθμους για την εύρεση προτύπων. Για να είμαστε πιο συγκεκριμένοι, το LDA βρίσκει ομάδες σχετικών λέξεων από:

1. αντιστοίχιση κάθε λέξης σε ένα τυχαίο θέμα, όπου ο χρήστης ορίζει τον αριθμό των θεμάτων που θέλει να αποκαλύψει. Δεν ορίζονται τα ίδια τα θέματα αυτά καθ' αυτά και ο αλγόριθμος θα αντιστοιχίσει όλα τα έγγραφα στα θέματα με τρόπο που οι λέξεις σε κάθε έγγραφο καταγράφονται κυρίως από αυτά τα φανταστικά θέματα. Ο αλγόριθμος περνάει από κάθε λέξη επαναληπτικά και επαναπροσδιορίζει τη λέξη σε ένα θέμα λαμβάνοντας υπόψη την πιθανότητα ότι η λέξη ανήκει σε ένα θέμα και την πιθανότητα ότι το έγγραφο θα δημιουργηθεί από ένα θέμα. Αυτές οι πιθανότητες υπολογίζονται πολλές φορές μέχρι ο αλγόριθμος να συγκλίνει.

2. Ο αλγόριθμος περνάει από κάθε λέξη επαναληπτικά και επαναπροσδιορίζει τη λέξη λαμβάνοντας υπόψη την πιθανότητα ότι η λέξη ανήκει σε ένα θέμα και την πιθανότητα ότι το έγγραφο θα δημιουργηθεί από ένα θέμα. Αυτές οι πιθανότητες υπολογίζονται πολλές φορές, μέχρι τη σύγκλιση του αλγορίθμου.

Η διαδικασία Topic Modeling είναι εξαιρετικά χρήσιμη για την ταξινόμηση κειμένων, τη δημιουργία συστημάτων συστάσεων (π.χ. για να σας προτείνουμε βιβλία με βάση τις προηγούμενες αναγνώσεις σας) ή ακόμα και για τον εντοπισμό τάσεων σε διαδικτυακές εκδόσεις [31].

Η έρευνα του NLP έχει εξελιχθεί σε τεράστιο βαθμό αφού από εκεί που η ανάλυση μιας πρότασης θα μπορούσε να διαρκέσει έως και κάποια λεπτά σήμερα στην εποχή της Google και άλλων τεχνολογικών κολοσών, εκατομμύρια ιστοσελίδες μπορούν να υποβληθούν σε επεξεργασία σε λιγότερο από ένα δευτερόλεπτο. Το NLP επιτρέπει στους υπολογιστές να εκτελούν ένα ευρύ φάσμα εργασιών που σχετίζονται με τη φυσική γλώσσα σε όλα τα επίπεδα, που κυμαίνονται από την ανάλυση και την επισήμανση μέρους του λόγου (POS), έως τα συστήματα αυτόματης μετάφρασης και διαλόγου. Μοντέλα Μηχανικής Μάθησης που χρησιμοποιούνται στο NLP είναι μοντέλα, όπως τα Αναδρομικά Νευρωνικά Δίκτυα (RNNs) και τα Συνελικτικά Νευρωνικά Δίκτυα (CNNs), που χρησιμοποιούνται για την επίλυση διαφόρων εργασιών και εφαρμογών NLP, όπως η αυτόματη μετάφραση, η απάντηση ερωτήσεων και τα συστήματα διαλόγου [28].

2.3.3 Περισσότερα για την Τμηματοποίηση

Η τμηματοποίηση είναι μία από τις πιο κοινές τεχνικές προεργασίας όταν πρόκειται για εργασία με δεδομένα κειμένου. Αλγόριθμοι οι οποίοι εκτελούν τμηματοποίηση ονομάζονται tokenizers. Υπάρχουν διάφοροι tokenizers και γενικά, ανάλογα με το μοντέλο που πρόκειται να εξυπηρετήσουν χρησιμοποιούν διαφορετικές τεχνικές. Ένας βασικός tokenizer είναι ο WordPiece tokenizer. Ο αλγόριθμος αυτός μοιάζει πολύ με τον αλγόριθμο BPE (Byte-Pair

Encoding) τον οποίο θα αναλύσουμε προκειμένου να γίνει καλύτερη κατανόηση του WordPiece αλγορίθμου. Όσον αφορά τον BPE λοιπόν ήταν ένας αλγόριθμος που παρουσιάστηκε στην εργασία με τίτλο "Neural Machine Translation of Rare Words with Subword Units" από τους Rico Sennrich, Barry Haddow και Alexandra Birch, το 2015 [32]. Αναλυτικά, ο αλγόριθμος BPE στηρίζεται σε έναν pre-tokenizer αλγόριθμο ο οποίος χωρίζει τα δεδομένα εκπαίδευσης σε λέξεις. Μετά την εκ των προτέρων προσδιορισμό, έχει δημιουργηθεί ένα σύνολο μοναδικών λέξεων και έχει προσδιοριστεί η συχνότητα κάθε λέξης που εμφανίστηκε στα δεδομένα εκπαίδευσης. Στη συνέχεια, το BPE δημιουργεί ένα βασικό λεξιλόγιο που αποτελείται από όλα τα σύμβολα που εμφανίζονται στο σύνολο των μοναδικών λέξεων και μαθαίνει κανόνες συγχώνευσης για να σχηματίσει ένα νέο σύμβολο από δύο σύμβολα του βασικού λεξιλογίου. Το κάνει έως ότου το λεξιλόγιο αποκτήσει το επιθυμητό μέγεθος λεξιλογίου. Το επιθυμητό μέγεθος λεξιλογίου είναι μια υπερπαράμετρος που πρέπει να οριστεί πριν από την εκπαίδευση του tokenizer. Για παράδειγμα, ας υποθέσουμε ότι μετά το pre-tokenization έχει προκύψει το παρακάτω σύνολο από λέξεις:

("hug", 10), ("pug", 5), ("pun", 12), ("bun", 4), ("hugs", 5)

Κατά συνέπεια, το βασικό λεξιλόγιο είναι ["b", "g", "h", "n", "p", "s", "u"]. Διαχωρίζοντας όλες τις λέξεις σε σύμβολα του βασικού λεξιλογίου, παίρνουμε:

("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s" ,5)

Στη συνέχεια, το BPE μετράει τη συχνότητα κάθε πιθανού ζεύγους συμβόλων και επιλέγει το ζεύγος συμβόλων που εμφανίζεται πιο συχνά. Στο παραπάνω παράδειγμα σημεία όπου το "h" ακολουθείται από το "u" υπάρχει 10 + 5 = 15 φορές (10 φορές στις 10 εμφανίσεις του "hug", 5 φορές στις 5 εμφανίσεις του "hugs"). Ωστόσο, το πιο συχνό ζεύγος συμβόλων είναι εκεί όπου "u" ακολουθείται από το "g", που εμφανίζεται 10 + 5 + 5 = 20 φορές συνολικά. Έτσι, ο πρώτος κανόνας συγχώνευσης που μαθαίνει ο tokenizer είναι να ομαδοποιεί όλα τα "u" σύμβολα που ακολουθούνται από ένα "g" σύμβολο μαζί. Στη συνέχεια, το ζεύγος "ug" προστίθεται στο λεξιλόγιο. Το σύνολο των λέξεων γίνεται τότε:

("h" "ug", 10), ("p" "ug", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "ug" "s", 5)

Στη συνέχεια, το BPE προσδιορίζει το επόμενο πιο κοινό ζεύγος στο νέο σύνολο συμβόλων. Αυτό είναι όταν το "u" ακολουθείται από το "n", το οποίο εμφανίζεται 16 φορές. Επομένως τα σύμβολα "u", "n" συγχωνεύονται στο ζεύγος "un" και αυτό προστίθεται στο λεξιλόγιο. Συνεχίζοντας την ίδια διαδικασία, το επόμενο πιο συχνό ζεύγος συμβόλων είναι όταν το "h" ακολουθείται από το "ug", το οποίο εμφανίζεται 15 φορές. Και πάλι το ζευγάρι συγχωνεύεται και προκύπτει το "hug" το οποίο προστίθεται στο λεξιλόγιο. Σε αυτό το στάδιο, το λεξιλόγιο είναι το: ["b", "g", "h", "n", "p", "s", "u", "ug", "un", "hug"] και το σύνολο των μοναδικών μας λέξεων είναι το:

("hug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("hug" "s", 5)

Υποθέτοντας ότι η εκπαίδευση Κωδικοποίησης Byte-Pair θα σταματήσει σε αυτό το σημείο, οι κανόνες συγχώνευσης θα εφαρμόζονταν στη συνέχεια σε νέες λέξεις (εφόσον αυτές οι νέες λέξεις δεν περιλαμβάνουν σύμβολα που δεν υπήρχαν στο βασικό λεξιλόγιο). Για παράδειγμα, η λέξη "bug" θα μεταφραστεί στο διακριτικό, ["b", "ug"] αλλά η λέξη "mug" μπορεί να μεταφραστεί σε ["<unk>", "ug"], όπου το "<unk>" είναι αναγνωριστικό για μια άγνωστη στο λεξιλόγιο λέξη αν για παράδειγμα το σύμβολο "m" δεν βρίσκεται στο βασικό λεξιλόγιο. Γενικά, μεμονωμένα γράμματα όπως "m" δεν αντικαθίστανται από το "<unk>" αναγνωριστικό επειδή τα δεδομένα εκπαίδευσης συνήθως περιλαμβάνουν τουλάχιστον μία εμφάνιση από κάθε γράμμα, αλλά είναι πιθανό να συμβεί για πολύ ειδικούς χαρακτήρες όπως τα emoji.

Αφού αναλύθηκε η εφαρμογή του αλγορίθμου BPE μπορεί να γίνει η σύνδεση με τον WordPiece αλγόριθμο. Κατά την εκτέλεση του WordPiece αρχικά αρχικοποιείται το λεξιλόγιο για να συμπεριλάβει κάθε χαρακτήρα που υπάρχει στα δεδομένα εκπαίδευσης και σταδιακά μαθαίνει έναν δεδομένο αριθμό κανόνων συγχώνευσης όπως προηγουμένως. Σε αντίθεση με το BPE, το WordPiece δεν επιλέγει το πιο συχνό ζεύγος συμβόλων, αλλά αυτό που μεγιστοποιεί την πιθανότητα των δεδομένων εκπαίδευσης μόλις προστεθούν στο λεξιλόγιο. Διαισθητικά, το WordPiece είναι ελαφρώς διαφορετικό από το BPE στο ότι αξιολογεί τι χάνει συγχωνεύοντας δύο σύμβολα για να διασφαλίσει ότι αξίζει τον κόπο [33].

2.3.4 Σύνολα δεδομένων για εργασίες επεξεργασίας φυσικής γλώσσας

Ήδη έχει γίνει αναφορά στον μεγάλο αριθμό συνόλων δεδομένων που είναι διαθέσιμος σήμερα. Γενικά ένα σύνολο δεδομένων χρησιμοποιείται τόσο στο στάδιο της εκπαίδευσης όσο και στο στάδιο του ελέγχου ενός μοντέλου. Τα δεδομένα που χρησιμοποιούνται σε ένα μοντέλο χωρίζονται σε τρεις κατηγορίες ανάλογα με το στάδιο ανάπτυξης του μοντέλου που χρησιμοποιούνται. Οι κατηγορίες αυτές είναι οι εξής:

1. **Σύνολο δεδομένων εκπαίδευσης (Training dataset):** Ένα σύνολο δεδομένων εκπαίδευσης είναι ένα σύνολο δεδομένων παραδειγμάτων που χρησιμοποιούνται κατά τη διάρκεια της εκπαιδευτικής διαδικασίας και χρησιμοποιείται για την προσαρμογή των παραμέτρων (π.χ. βάρη) του δικτύου.

2. **Σύνολο δεδομένων επικύρωσης (Validation dataset ή dev set):** Ένα σύνολο δεδομένων επικύρωσης είναι ένα σύνολο δεδομένων παραδειγμάτων που χρησιμοποιούνται για τον συντονισμό των υπερπαραμέτρων του δικτύου. Ένα παράδειγμα υπερπαραμέτρου για τεχνητά Νευρωνικά Δίκτυα περιλαμβάνει τον αριθμό των κρυφών μονάδων σε κάθε επίπεδο.

3. **Σύνολο δεδομένων ελέγχου (test set):** Ένα σύνολο δεδομένων ελέγχου (test set) είναι ένα σύνολο δεδομένων που είναι ανεξάρτητο από το σύνολο δεδομένων εκπαίδευσης (train set), αλλά που ακολουθεί την ίδια κατανομή πιθανοτήτων με αυτό [34].

Προκειμένου να γίνει η παρουσίαση των επιδόσεων διαφόρων μοντέλων σε εργασίες NLP γίνεται σύγκριση των επιδόσεων των μοντέλων σε συγκεκριμένα σύνολα δεδομένων. Σύνολα δεδομένων είναι διαθέσιμα στο κοινό τόσο για την εκπαίδευση (training) του μοντέλου όσο και

για τον έλεγχο (testing) του. Για αυτό το λόγο γίνεται σύντομη αναφορά των χαρακτηριστικών συνόλων δεδομένων που θα αναφερθούν παρακάτω προκειμένου να γίνει σύγκριση τους.

Συλλογή GLUE

Μια βασική μονάδα μέτρησης των δυνατοτήτων διαφόρων Νευρωνικών Δικτύων σε εργασίες NLP είναι η συλλογή συνόλων δεδομένων General Language Understanding Evaluation benchmark (GLUE) που χρησιμοποιούνται για εκπαίδευση, αξιολόγηση και ανάλυση μοντέλων NLP μεταξύ τους, με στόχο την προώθηση της έρευνας στην ανάπτυξη γενικών και ισχυρών συστημάτων κατανόησης φυσικής γλώσσας. Η συλλογή αποτελείται από εννέα δύσκολα και διαφορετικά σύνολα δεδομένων εργασιών που έχουν σχεδιαστεί για να ελέγχουν τη γλωσσική κατανόηση ενός μοντέλου και είναι ζωτικής σημασίας για την κατανόηση του τρόπου με τον οποίο αξιολογούνται τα μοντέλα μεταφοράς μάθησης. Τα μοντέλα που θα αναφερθούν λοιπόν συγκρίνονται σε αυτή την συλλογή δεδομένων προκειμένου να φανεί η διαφορά τους σε αυτή τη μαζική μονάδα [35].

SQUAD

Το σύνολο δεδομένων Stanford Question Answering (SQuAD) είναι ένα σύνολο δεδομένων κατανόησης ανάγνωσης, που αποτελείται από ερωτήσεις που τίθενται πάνω σε ένα σύνολο άρθρων της Wikipedia, όπου η απάντηση σε κάθε ερώτηση είναι ένα τμήμα κειμένου από το αντίστοιχο απόσπασμα ανάγνωσης. Η πρώτη έκδοση του συνόλου δεδομένων SQUAD είναι η SQUAD1.1 που περιέχει 100.000+ ζεύγη ερωτήσεων-απαντήσεων σε παραπάνω από 500 άρθρα. Στη συνέχεια, η επόμενη έκδοση του συνόλου δεδομένων. Το SQuAD2.0, συνδυάζει τις 100.000 ερωτήσεις του SQuAD1.1 με περισσότερες από 50.000 αναπάντητες ερωτήσεις που έχουν γραφτεί εναλλακτικά για να μοιάζουν με αυτές που μπορούν να απαντηθούν. Για να τα πάνε καλά στο SQuAD2.0, τα συστήματα πρέπει όχι μόνο να απαντούν σε ερωτήσεις όταν είναι δυνατόν, αλλά και να προσδιορίζουν πότε καμία απάντηση δεν υποστηρίζεται από την παράγραφο και να απέχουν από την απάντηση [36].

IMDb

Το σύνολο δεδομένων IMDB είναι ένα σύνολο δεδομένων που περιλαμβάνει 50.000 κριτικές ταινιών για επεξεργασία φυσικής γλώσσας ή ανάλυση κειμένου. Το σύνολο δεδομένων αυτό είναι ένα σύνολο δεδομένων για δυαδική ταξινόμηση συναισθημάτων που περιέχει σημαντικά περισσότερα δεδομένα από τα προηγούμενα σύνολα δεδομένων αναφοράς. Σε αυτό παρέχεται ένα σύνολο από 25.000 κριτικές διαφόρων συναισθημάτων για εκπαίδευση καθώς και 25.000 για έλεγχο. [37].

Bookcorpus

Το σύνολο δεδομένων Bookcorpus είναι μια μεγάλη συλλογή αδημοσίευτων δωρεάν μυθιστορηματικών βιβλίων, η οποία περιέχει 11.038 βιβλία (περίπου 74 εκατομμύρια προτάσεις και λέξεις) 16 διαφορετικών ειδών (π.χ. Ρομαντικό, Ιστορικό, Περιπέτεια, κ.λ.π.) [38].

Κεφάλαιο 3

Μέθοδοι και Τεχνολογίες

Στο κεφάλαιο αυτό παρουσιάζονται τεχνολογίες και μέσα που χρησιμοποιήθηκαν στην παρούσα ανάλυση. Θα αναφερθούν τεχνολογίες και τεχνικές που σχετίζονται με το κομμάτι της εργασίας που αφορά την ανάπτυξη της εφαρμογής καθώς, και τεχνικές για την ενσωμάτωση αλγορίθμων Μηχανικής Μάθησης σε αυτή.

3.1 Java

Η Java είναι μια γλώσσα προγραμματισμού και μια υπολογιστική πλατφόρμα που κυκλοφόρησε για πρώτη φορά από τη Sun Microsystems, το 1995 [39]. Η Java ως αντικειμενοστραφής γλώσσα προγραμματισμού βασίζεται σε κλάσεις και αντικείμενα. Μια κλάση (class) είναι μια φόρμα για τη δημιουργία αντικειμένων (objects ή instances). Αφού πρώτα δημιουργηθεί μια κλάση, στη συνέχεια μπορεί να χρησιμοποιηθεί για την παραγωγή αντικειμένων έτσι με αυτόν τον τρόπο δίνει την δυνατότητα επαναχρησιμοποίησης κώδικα, μειώνοντας έτσι το κόστος ανάπτυξης. Το προϊόν αυτό ανήκει στην Oracle και περισσότερες από 3 δισεκατομμύρια συσκευές τρέχουν Java. Επιπλέον η Java έχει μια τεράστια κοινότητα μελών υποστήριξης που ξεπερνούν τους 10 εκατομμύρια προγραμματιστές [40]. Η Java έχει τροφοδοτήσει ένα μεγάλο μερίδιο του σημερινού ψηφιακού κόσμου, παρέχοντας μια αξιόπιστη πλατφόρμα πάνω στην οποία χτίζονται πολλές υπηρεσίες και εφαρμογές. Νέα, καινοτόμα προϊόντα και ψηφιακές υπηρεσίες που έχουν σχεδιαστεί για το μέλλον συνεχίζουν να βασίζονται μέχρι και σήμερα στην Java. Υπάρχουν πολλές εφαρμογές, ακόμη και ορισμένες ιστοσελίδες που δεν θα λειτουργήσουν αν δεν έχει εγκατασταθεί η Java [39].

Η γλώσσα Java χρησιμοποιείται για εφαρμογές κινητών τηλεφώνων, ειδικά για το λειτουργικό σύστημα Android, εφαρμογές υπολογιστών, διακομιστές ιστού και εφαρμογών, παιχνίδια, βάσεις δεδομένων και πολλά άλλα. Η Java είναι μια από τις πιο ευρέως διαδεδομένες γλώσσες ακριβώς επειδή λειτουργεί σε διαφορετικές πλατφόρμες όπως Linux, Mac, Raspberry Pi κλπ. Είναι μια γλώσσα ανοιχτού κώδικα, ασφαλής, γρήγορη και ισχυρή. Η Java είναι δωρεάν για λήψη και προσωπική χρήση. Η Java είναι κοντά στις C++ και C#, γεγονός που διευκολύνει προγραμματιστές να μεταβούν από τη Java σε αυτές ή το αντίστροφο. Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος από την πλατφόρμα. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh χωρίς να χρειάζεται να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για

κάθε διαφορετικό λειτουργικό σύστημα [40].

Όσον αφορά στη χρήση της Java σε Android συσκευές υπάρχουν πολύ σοβαροί λόγοι που την καθιστούν κατάλληλη για αυτή τη πλατφόρμα. Συγκεκριμένα, δεδομένου ότι κύριος στόχος πίσω από την ανάπτυξη του Android ήταν να δημιουργηθεί ένα περιβάλλον εφαρμογών ανεξάρτητο από την εκάστοτε πλατφόρμα και έτσι το λογισμικό αυτό να μπορεί να τρέχει σε πλήθος συσκευών, η Java ήταν μια προφανής επιλογή αφού όπως γνωρίζουμε σαν γλώσσα έχει ήδη αυτή την ποιότητα. Επιπλέον πολύ σημαντικό για την επιλογή της, είναι ότι οι εφαρμογές Android τρέχουν σε μια ειδική εικονική μηχανή που ονομάζεται Dalvik VM που είναι άμεση έμπνευση από την εικονική μηχανή της Java που ονομάζεται JVM. Η εφαρμογή Android μπορεί να εκτελεστεί σε οποιαδήποτε συσκευή, όπου εφαρμόζεται το ειδικό Dalvik VM. Με αυτόν τον τρόπο οι εφαρμογές Android μεταγλωττίζονται και εκτελούνται σε περιβάλλον βέλτιστης απόδοσης με το χαρακτηριστικό της ανεξαρτησίας από την πλατφόρμα υλικού.

Εκτός όλων αυτών, η καλή προσέγγιση για την ανάπτυξη λογισμικού είναι η αντικειμενοστραφής προσέγγιση. Η Java βασίζεται στην έννοια του OOPs (Object-Oriented Programming System). Το Android βασίζεται σε μεγάλο βαθμό στις βασικές αρχές της Java, όπως οι κλάσεις και τα αντικείμενα. Εκτός αυτών η Java έχει ένα εκτεταμένο σύνολο βιβλιοθηκών. Το Android SDK περιλαμβάνει πολλές τυπικές βιβλιοθήκες Java. Οι βιβλιοθήκες αυτές παρέχουν λειτουργίες για δομές δεδομένων, μαθηματικές συναρτήσεις, γραφικά, λειτουργίες δικτύωσης και πολλά άλλα. Με αυτόν τον τρόπο η Java βοηθά στην ανάπτυξη εφαρμογών Android με γρήγορο και αναποτελεσματικό τρόπο και έτσι το Android αποτελεί λογισμικό που έχει σχεδιαστεί με αρχιτεκτονική ουδετερότητας υλικού. Τέλος, η Java όπως προαναφέρθηκε είναι μια πολύ δημοφιλής γλώσσα με μια τεράστια κοινότητα προγραμματιστών. Έτσι, είναι λογικό οι προγραμματιστές Android να επιλέξουν την Java καθώς ήδη υπάρχει μια καλή βάση προγραμματιστών διαθέσιμοι και μπορούν να συμβάλλουν στη γρήγορη ανάπτυξη πολλών εφαρμογών και την βελτιστοποίηση τους. Επιπλέον με τη σειρά τους πολλές βιβλιοθήκες και εργαλεία στη Java κάνουν τη ζωή των προγραμματιστών πιο εύκολη ενώ οι προγραμματιστές που δεν χρησιμοποιούν Java πρέπει να αντιμετωπίσουν σοβαρά προβλήματα, όπως η διαρροή μνήμης και η δυσκολία στην χρήση των δεικτών (pointers) που μπορούν να οδηγήσουν σε σφάλμα του προγράμματος. Μερικές φορές αυτά τα προβλήματα βλάπτουν σε υψηλότερο επίπεδο και οδηγούν σε καταστάσεις όπως η κατάρρευση της εφαρμογής ή χειρότερα την κατάρρευση του λειτουργικού συστήματος [41].

3.2 Android OS

Το Android OS είναι ένα λειτουργικό σύστημα για κινητές συσκευές το οποίο αναπτύχθηκε από την Google έτσι ώστε να χρησιμοποιείται κυρίως σε συσκευές με οθόνη αφής, κινητά τηλέφωνα και tablet. Η Google χρησιμοποιεί επίσης λογισμικό Android σε τηλεοράσεις, κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, αυτοκίνητα και ρολόγια χειρός, καθένα από τα οποία είναι εξοπλισμένο με μια μοναδική διεπαφή χρήστη [42, 43]. Το λειτουργικό σύστημα Android αναπτύχθηκε για πρώτη φορά από την Android, Inc., μια εταιρεία λογισμικού που βρίσκεται στη Silicon Valley πριν την αποκτήσει η Google το 2005. Αμέσως μετά, η Google ανακοίνωσε την επικείμενη κυκλοφορία της πρώτης εμπορικά διαθέσιμης

συσκευής με Android το 2007, αν και το προϊόν αυτό κυκλοφόρησε στην αγορά το 2008 [42].

Το Android OS είναι λειτουργικού σύστημα ανοιχτού κώδικα και είναι βασισμένο στο Linux [43, 44]. Το Android είναι ένα πλήρες λειτουργικό σύστημα, με προσαρμόσιμο πηγαίο κώδικα που μπορεί να μεταφερθεί σε σχεδόν οποιαδήποτε συσκευή. Διαθέτει δημόσια τεκμηρίωση και κανόνες που είναι διαθέσιμα σε όλους [44]. Από την ανάπτυξη του λογισμικού αυτού και έπειτα οι προγραμματιστές λογισμικού και εφαρμογών μπόρεσαν να χρησιμοποιούν την τεχνολογία Android για την ανάπτυξη εφαρμογών για κινητά τηλέφωνα, οι οποίες πωλούνται μέσω καταστημάτων εφαρμογών, όπως το Google Play. Επιπλέον, επειδή το λογισμικό αυτό έχει αναπτυχθεί ως προϊόν της Google, οι χρήστες Android έχουν την ευκαιρία να συνδέσουν τις φορητές συσκευές τους με άλλα προϊόντα Google, όπως προϊόντα αποθήκευσης cloud, πλατφόρμες ηλεκτρονικού ταχυδρομείου και υπηρεσίες βίντεο [42].

Το λογισμικό Android έχει κάνει πολλές αναβαθμίσεις από το 2008 με διορθώσεις σφαλμάτων προηγούμενων εκδόσεων καθώς και συνεχείς προσθήκες νέων λειτουργιών. Κάθε μία από τις πρώτες κύριες εκδόσεις Android ονομάζεται, με αλφαβητική σειρά, με το όνομα ενός γλυκού. Την αρχή κάνει το Cupcake (έκδοση 1.5), το 2008, ενώ ακολουθείται από το Donut (έκδοση 1.6) το 2009. Την ίδια χρονιά ακολουθεί το Eclair (έκδοση 2.0) και την επόμενη ακολουθεί το Froyo (έκδοση 2.2). Το 2011, ακολουθούν τα Gingerbread (έκδοση 2.3), Honeycomb (έκδοση 3.2) και Ice Cream Sandwich (έκδοση 4.0), ενώ το 2012 ακολουθεί το Jelly Bean με τρεις εκδόσεις μέσα στη διετία 2012-2013 (εκδόσεις 4.1 έως 4.3). Αργότερα την ίδια χρονιά, κυκλοφορεί το Lollipop (έκδοση 5.0) και η ανανεωμένη έκδοση αυτού κάποιους μήνες μετά (έκδοση 5.1). Επόμενα στη σειρά είναι το Marshmallow (έκδοση 6.0), το 2015, στη συνέχεια το Nougat (εκδόσεις 7.0 και 7.1), το 2016 και το Oreo (εκδόσεις 8.0 και 8.1), το 2017. Η τελευταία έκδοση Android που τηρεί την παράδοση ονοματολογίας από γλυκά είναι το Pie (έκδοση 9), το 2018. Μια νέα εποχή στο Android ξεκινάει με την έκδοση του Android 10, το 2019 και στη συνέχεια του Android 11, το 2020. Τέλος, η πιο πρόσφατη έκδοση μέχρι και σήμερα είναι το Android 12 το οποίο δημοσιεύτηκε τον Οκτώβρη του 2021. Το Android είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο, τρέχει σε συσκευές διαφόρων εταιριών ενώ και η ίδια η Google διαθέτει μια σειρά συσκευών με το όνομα Pixel από το 2013. Οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS X μαζί [43].

3.2.1 Αρχιτεκτονική μιας εφαρμογής Android

Μια τυπική εφαρμογή Android περιέχει πολλά δομικά στοιχεία, συμπεριλαμβανομένων των δραστηριοτήτων (activities), των τμημάτων (fragments), των υπηρεσιών (services), των παροχών περιεχομένου (content providers) και των δεκτών εκπομπής (broadcast receivers). Οι δραστηριότητες και τα τμήματα είναι κλάσεις που πρακτικά μεταφράζονται στον χρήστη ως οθόνες. Στις κλάσεις αυτές υλοποιούνται λειτουργίες που σχετίζονται με την σύνδεση λειτουργιών και διεπαφών χρήστη. Στη συνέχεια, υπάρχουν λειτουργίες που γίνονται στο παρασκήνιο της εφαρμογής (services), τις οποίες ο χρήστης δεν αντιλαμβάνεται (πχ κλήσεις σε απομακρυσμένους διακομιστές). Κάθε ένα από αυτά το δομικά στοιχεία, όπως οι λειτουργίες, έχει έναν κύκλο ζωής κάθε στάδιο του οποίου καθορίζουν μια κατάσταση της δομής

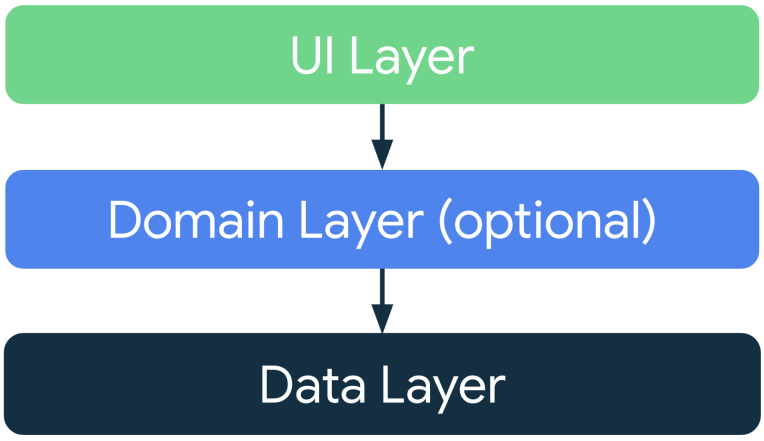
αυτής (έναρξη λειτουργίας, λειτουργία στο προσκήνιο, τερματισμός λειτουργία, προσωρινή παύση λειτουργίας κ.α.) Η σωστή αρχιτεκτονική μιας εφαρμογής Android απαιτεί την εφαρμογή του "separation of concerns" το οποίο απαιτεί τον διαχωρισμό της διεπαφής από τις λειτουργίες και τα δεδομένα της εφαρμογής. Έτσι η προτεινόμενη οργάνωση μιας εφαρμογής Android είναι ο διαχωρισμός της σε επίπεδα όπως φαίνεται και στην Εικόνα 3.1, τα οποία είναι:

1. **Το Επίπεδο διεπαφής χρήστη (UI Layer)**, το οποίο όπως λέει και ο τίτλος του είναι το επίπεδο που συνδέεται άμεσα με τον χρήστη και την αλληλεπίδραση του με την οθόνη.

2. **Το Επίπεδο τομέα (Domain Layer)**, το οποίο είναι υπεύθυνο για την ενθυλάκωση σύνθετης επιχειρηματικής λογικής ή απλής επιχειρηματικής λογικής που χρησιμοποιείται μεταξύ των δεδομένων και της διεπαφής. Αυτό το επίπεδο είναι προαιρετικό επειδή δεν θα έχουν όλες οι εφαρμογές αυτές τις απαιτήσεις.

3. **Το Επίπεδο δεδομένων (Data Layer)**, το οποίο περιέχει την επιχειρηματική λογική της εφαρμογής και εξάγει τα δεδομένα της εφαρμογής στο επίπεδο διεπαφής [6].

Ένα αρχιτεκτονικό μοτίβο λογισμικού που διευκολύνει τον διαχωρισμό της ανάπτυξης της διεπαφής χρήστη από την ανάπτυξη της επιχειρηματικής λογικής είναι το Model - View - View Model (MVVM). Στο μοτίβο αυτό υπάρχει μια κλάση με τίτλο View Model, η οποία είναι ουσιαστικά ο συνδετικός κρίκος μεταξύ των κλάσεων για τον χειρισμό των δεδομένων της εφαρμογής (model) και της διεπαφής χρήστη (view). Με αυτόν τον τρόπο όταν κάποιο μοντέλο της εφαρμογής ενημερώνεται τότε ενημερώνεται άμεσα εκείνη τη στιγμή και η διεπαφή που βλέπει ο χρήστης. Ένα View Model πρακτικά συνδέει τις δραστηριότητες και τα της εφαρμογής τόσο μεταξύ τους, για την ανταλλαγή πληροφοριών όσο και με τα μοντέλα δεδομένων [45].



Εικόνα 3.1: *Τυπική αρχιτεκτονική μιας εφαρμογής Android [6]*

3.2.2 Παραλληλοποίηση στο Android

Όταν μια εφαρμογή ξεκινάει την εκτέλεση της στο Android, δημιουργεί το πρώτο νήμα εκτέλεσης, γνωστό ως το «κύριο» νήμα (main thread). Το κύριο νήμα είναι υπεύθυνο για την αποστολή συμβάντων (events) στα κατάλληλα γραφικά στοιχεία διεπαφής χρήστη, καθώς και για την επικοινωνία με στοιχεία της διεπαφής χρήστη. Ως συμβάν ορίζεται κάποια αλληλεπίδρασή του χρήστη με την επαφή, όπως η πληκτρολόγηση ενός κειμένου ή το πάτημα ενός κουμπιού. Η σωστή πρακτική είναι το κύριο νήμα είναι να αποφεύγεται για την εκτέλεση λειτουργιών μιας εφαρμογής, όπως οι κλήσεις προς βάσεις δεδομένων ή προς διακομιστές. Για το λόγο αυτό, τέτοιες λειτουργίες εκτελούνται συνήθως σε ξεχωριστά νήματα, δηλαδή, εκτελούνται ασύγχρονα από το περιβάλλον χρήστη. Αυτό έχει ως αποτέλεσμα την προστασία και την απομόνωση των λειτουργιών της διεπαφής αποφεύγοντας έτσι τυχόν σφάλμα της εφαρμογής. Το Android παρέχει πολλούς τρόπους δημιουργίας και διαχείρισης νημάτων και υπάρχουν πολλές βιβλιοθήκες που κάνουν τη διαχείριση νημάτων πολύ πιο ευχάριστη. Γενικά, τα νήματα που χρησιμοποιούνται πέρα από το κύριο χωρίζονται στις εξής κατηγορίες:

1. **Νήματα που συνδέονται με τις λειτουργίες ή τα τμήματα:** τα οποία συνδέονται με τον κύκλο ζωής των δομών αυτών και με την καταστροφή αυτών καταστρέφονται και εκείνα προκειμένου να μην καταναλώνουν περιττούς πόρους.

2. **Νήματα που δεν συνδέονται με τις λειτουργίες ή τα τμήματα:** τα οποία συνεχίζουν να τρέχουν και μετά την καταστροφή των δομών αυτών [46].

3.3 Android Studio

Το Android Studio είναι το επίσημο Ολοκληρωμένο Περιβάλλον Ανάπτυξης (IDE: Integrated Development Environment) για την ανάπτυξη εφαρμογών Android, με βάση το IntelliJ IDEA. Το Android Studio ήταν διαθέσιμο σε πρώιμο στάδιο για προεπισκόπηση ξεκινώντας από την έκδοση 0.1 τον Μάιο του 2013, έπειτα ξεκίνησε το δοκιμαστικό στάδιο από την έκδοση 0.8 που βγήκε τον Ιούνιο του 2014. Η πρώτη σταθερή έκδοση βγήκε το Δεκέμβριο του 2014, με την έκδοση 1.0. Βασισμένο στο λογισμικό JetBrains της ΙντελλιΘ ΙΔΕΑ, το Android Studio σχεδιάστηκε αποκλειστικά για προγραμματισμό Android. Είναι διαθέσιμο για Windows, Mac OS X και Linux, και αντικατέστησε το Eclipse Android Development Tools (ADT) ως το κύριο IDE της Google για την ανάπτυξη εφαρμογών Android.

Το Android Studio περιέχει εργαλεία για ανάπτυξη, εντοπισμό σφαλμάτων, δοκιμές κ.α που καθιστούν ταχύτερη και ευκολότερη την ανάπτυξη εφαρμογών. Τα βασικά στοιχεία της διεπαφής και της λειτουργίας του Android Studio είναι τα εξής:

1. **Layout Editor**: ο Layout Editor είναι ένα εικονικό περιβάλλον για την σχεδίαση της διεπαφής της εφαρμογής. Ο editor αυτός δίνει τη δυνατότητα γρήγορης δημιουργίας διατάξεων. Η δημιουργία αυτή μπορεί να γίνει σέρνοντας στοιχεία διεπαφής χρήστη σε ένα πρόγραμμα επεξεργασίας οπτικού σχεδιασμού αντί του γραψίματος xml αρχείων προγραμματιστικά.

2. **Αναλυτής APK**: ο Αναλυτής APK παρέχει άμεσες πληροφορίες σχετικά με τη σύνθεση του APK ή του πακέτου εφαρμογών Android μετά την ολοκλήρωση της διαδικασίας κατασκευής. Η χρήση του APK Analyzer μπορεί να μειώσει τον χρόνο για τον εντοπισμό σφαλμάτων και να συμβάλει στη μείωση του μεγέθους του τελικού APK που παράγεται.

3. **Προσομοιωτές (Emulators):** Υπάρχει μια μεγάλη γκάμα προδιαμορφωμένων προσομοιωτών όπου μπορεί να δοκιμαστεί η εκάστοτε εφαρμογή. Με αυτόν τον τρόπο ο προγραμματιστής έχει τη δυνατότητα να πραγματοποιήσει δοκιμές σε συσκευές διαφορετικών προδιαγραφών. Επιπλέον, εκτός αυτού δίνεται η δυνατότητα δοκιμών σε πραγματικές συσκευές με δυνατότητα σύνδεσης με τον υπολογιστή τόσο μέσω Usb όσο και μέσω Wifi.

4. **Code Editor**: ο Code Editor είναι ένας έξυπνος επεξεργαστής κώδικα που παρέχει λειτουργία συμπλήρωσης κώδικα για γλώσσες Kotlin, Java και C/C++ αυξάνοντας έτσι την παραγωγικότητα και την ταχύτητα της διαδικασίας γραψίματος του κώδικα της εφαρμογής.

5. **Ευέλικτο build σύστημα:** το build σύστημα που παρέχεται είναι στηριζόμενο στο Gradle, ένα προηγμένο σύνολο εργαλείων, για την αυτοματοποίηση και τη διαχείριση της διαδικασίας κατασκευής της εφαρμογής.

6. **Εργαλεία για δοκιμές:** Εκτός όλων αυτών που προαναφέρθηκαν υπάρχουν εκτεταμένα εργαλεία για την διευκόλυνση των δοκιμών (Testing) των εφαρμογών Android με το JUnit 4 και λειτουργικά πλαίσια δοκιμής διεπαφής χρήστη. Με το Espresso Test Recorder, μπορεί κανείς να δημιουργήσει τον κώδικα δοκιμής διεπαφής χρήστη καταγράφοντας τις αλληλεπιδράσεις χρήστη με την εφαρμογή σε μια συσκευή ή προσομοιωτή.

7. **Android Profiler:** ο Android Profiler υπάρχει για εκδόσεις Android Studio 3.0. Ο Android Profiler παρέχει δεδομένα σε πραγματικό χρόνο τα οποία διευκολύνουν την κατανόηση του τρόπου λειτουργίας της εφαρμογής, σχετικά με το πώς η εφαρμογή χρησιμοποιεί πόρους επεξεργαστή, μνήμης, μπαταρίας και δικτύου [47].

3.4 Python

Η Python είναι μια διερμηνευόμενη (interpreted), διαδραστική (interactive), αντικειμενοστραφής (object-oriented) γλώσσα προγραμματισμού. Πιο αναλυτικά, η Python είναι μια scripting γλώσσα, μια γλώσσα δηλαδή που επιτρέπει τον συνδυασμό στοιχειωδών εργασιών ή κλήσεων διεπαφής προγραμματισμού εφαρμογών (API) [14]. Σαν γλώσσα ενσωματώνει λειτουργικές μονάδες (modules), εξαιρέσεις (exceptions), δυναμική πληκτρολόγηση, πολύ υψηλού επιπέδου δυναμικούς τύπους δεδομένων (data types) και κλάσεις (classes), ενώ υποστηρίζει πολλαπλά παραδείγματα προγραμματισμού πέρα από αντικειμενοστραφή προγραμματισμό, όπως διαδικαστικό (procedural) και λειτουργικό (functional) προγραμματισμό. Το χαρακτηριστικό της Python είναι ότι συνδυάζει προγραμματιστική δύναμη με πολύ

σαφή σύνταξη. Επιπλέον σαν γλώσσα έχει δυνατότητα διασύνδεσης με πολλές κλήσεις συστήματος και βιβλιοθήκες ενώ μπορεί επίσης να χρησιμοποιηθεί ως γλώσσα επέκτασης για εφαρμογές που χρειάζονται προγραμματιζόμενη διεπαφή. Τέλος, η Python είναι "portable" γλώσσα δηλαδή τρέχει σε πολλές παραλλαγές Unix, συμπεριλαμβανομένων Linux και MacOS, αλλά και σε Windows [48].

Για όλους αυτούς τους λόγους η Python έχει πολλές εφαρμογές. Η μεγάλη τυπική βιβλιοθήκη της Python, που συνήθως αναφέρεται ως ένα από τα μεγαλύτερα δυνατά της σημεία, παρέχει εργαλεία κατάλληλα για πολλές εργασίες. Για εφαρμογές που έχουν πρόσβαση στο Διαδίκτυο, υποστηρίζονται πολλές τυπικές μορφές και πρωτόκολλα όπως το MIME και το HTTP. Επιπλέον, περιλαμβάνει ενότητες για τη δημιουργία γραφικών διεπαφών χρήστη, σύνδεση με σχεσιακές βάσεις δεδομένων, δημιουργία ψευδοτυχαίων αριθμών, αριθμητική με αυθαίρετα δεκαδικά ψηφία, χειρισμό κανονικών εκφράσεων και άλλα. Από τον Σεπτέμβριο 2021, το Python Package Index (PyPI), το επίσημο αποθετήριο για λογισμικό τρίτων κατασκευαστών Python, περιέχει πάνω από 329.000 πακέτα με ένα ευρύ φάσμα λειτουργιών, όπως: Βάσεις Δεδομένων, Ανάλυση Δεδομένων, Μηχανική Μάθηση, Εφαρμογές κινητών, Δίκτυα Υπολογιστών κ.α [49].

Όσον αφορά στο πεδίο της Μηχανικής Μάθησης η Python παρέχει το Keras που είναι ένα API Βαθιάς Μάθησης, που τρέχει πάνω από την πλατφόρμα Μηχανικής Μάθησης TensorFlow. Αναπτύχθηκε με έμφαση στην παροχή δυνατότητας γρήγορου πειραματισμού. Το να μπορεί κάποιος να πηγαίνει από την ιδέα στο αποτέλεσμα όσο το δυνατόν γρηγορότερα είναι το κλειδί για την καλή έρευνα. Το Keras μειώνει τον γνωστικό φόρτο του προγραμματιστή ενώ υιοθετεί την αρχή της προοδευτικής αποκάλυψης της πολυπλοκότητας: οι απλές ροές εργασίας πρέπει να είναι γρήγορες και εύκολες, ενώ οι αυθαίρετα προηγμένες ροές εργασίας θα πρέπει να είναι δυνατές μέσω μιας σαφούς διαδρομής που βασίζεται σε ήδη υπάρχουσες γνώσεις. Τέλος, το Keras παρέχει απόδοση και επεκτασιμότητα στη βιομηχανία ενώ χρησιμοποιείται από οργανισμούς και εταιρείες όπως η NASA, το YouTube ή η Waymo [50].

3.5 Jupyter Notebook

Το Project Jupyter είναι ένα μη κερδοσκοπικό έργο ανοιχτού κώδικα, που γεννήθηκε από το IPython Project το 2014 και εξελίχθηκε για να υποστηρίξει τη διαδραστική επιστήμη δεδομένων και τον επιστημονικό υπολογισμό σε όλες τις γλώσσες προγραμματισμού.

Η Jupyter Notebook εφαρμογή είναι μια εφαρμογή διακομιστή-πελάτη (web app) που επιτρέπει την επεξεργασία και την εκτέλεση κώδικα μέσω ενός προγράμματος περιήγησης ιστού. Η εφαρμογή Notebook Jupyter μπορεί να εκτελεστεί σε υπολογιστή, τοπικά χωρίς να απαιτεί πρόσβαση στο Διαδίκτυο ή μπορεί να εγκατασταθεί σε έναν απομακρυσμένο διακομιστή και να έχει πρόσβαση σε αυτόν μέσω Διαδικτύου. Εκτός από την εμφάνιση, επεξεργασία και εκτέλεση κώδικα, η εφαρμογή Notebook Jupyter έχει έναν "Πίνακα εργαλείων" και έναν "Πίνακα ελέγχου" που εμφανίζουν τοπικά αρχεία και επιτρέπει το άνοιγμα κελιών όπου εκτελείται ο κώδικας.

Συγκεκριμένα, ο κώδικας που υπάρχει σε κάθε κελί εκτελείται μέσω του πυρήνα (kernel). Ένας πυρήνας είναι μια «υπολογιστική μηχανή». Ο πυρήνας ipython, εκτελεί κώδικα

Python. Υπάρχουν πυρήνες για πολλές άλλες γλώσσες. Όταν δημιουργείται ένα νέο έγγραφο Notebook, ο συσχετισμένος του με τον πυρήνα εκκινείται αυτόματα. Όταν εκτελείται το περιεχόμενο ενός κελιού ή όλα τα κελιά μαζί, ο πυρήνας εκτελεί τον υπολογισμό και παράγει τα αποτελέσματα. Ανάλογα με τον τύπο των υπολογισμών, ο πυρήνας μπορεί να καταναλώνει σημαντικούς πόρους από τον επεξεργαστή (CPU) και τη μνήμη RAM ενώ συγκεκριμένα η μνήμη RAM δεν αποδεσμεύεται μέχρι να κλείσει ο πυρήνας [51].

Το Jupyter Notebook επομένως αποτελεί ένα εργαλείο όπου μπορεί να εκπαιδευτεί ή να τρέξει ένα μοντέλο Μηχανικής Μάθησης γραμμένο σε Python χωρίς να χρειάζονται πολλές εγκαταστάσεις άλλων βοηθητικών προγραμμάτων ή συσκευή ιδιαίτερων προδιαγραφών. Για αυτόν ακριβώς το λόγο χρησιμοποιείται ευρέως για αυτό το σκοπό αφού ο προγραμματιστής αρκεί να γράψει τον κώδικα του και με το πάτημα ενός κουμπιού να δει απευθείας το αποτέλεσμα αυτού σε λογικό χρόνο. Έτσι, στην παρούσα διπλωματική εργασία στο Jupyter Notebook εκτελείται κώδικας που συνδέεται με την μετατροπή του μοντέλου σε μορφή που μπορεί να εισαχθεί στην Android συσκευή και να εκτελεστεί σε αυτή. Η μορφή είναι η TensorFlow Lite με χαρακτηριστικά που θα αναλυθούν στην συνέχεια.

3.6 TensorFlow και TensorFlow Lite

Γενικά το TensorFlow είναι μια από άκρο σε άκρο (end-to-end) πλατφόρμα ανοιχτού κώδικα για Μηχανική Μάθηση. Διαθέτει ένα ολοκληρωμένο, ευέλικτο σύστημα εργαλείων, βιβλιοθηκών και κοινοτικών πόρων που επιτρέπει στους ερευνητές να προωθήσουν την τελευταία λέξη της τεχνολογίας στην Μηχανική Μάθηση. Οι προγραμματιστές μπορούν μέσω της πλατφόρμας αυτής να δημιουργούν και να αναπτύξουν εύκολα εφαρμογές που υποστηρίζουν Μηχανική Μάθηση. Το TensorFlow προσφέρει πολλαπλά επίπεδα αφαίρεσης (levels of abstraction), ώστε να μπορεί κανείς να επιλέξει το κατάλληλο επίπεδο για τις ανάγκες της εκάστοτε εφαρμογής. Μέσω του TensorFlow μπορεί κανείς να δημιουργήσει και να εκπαιδεύσει μοντέλα χρησιμοποιώντας το υψηλού επιπέδου Keras API, το οποίο καθιστά εύκολο το ξεκίνημα με το TensorFlow και τη Μηχανική Μάθηση. Εκτός αυτού αν τυχόν κάπου χρειάζεται περισσότερη ευελιξία, ο τρόπος εκτέλεσης επιτρέπει την άμεση επανάληψη και τον διαισθητικό εντοπισμό σφαλμάτων. Για μεγάλες εργασίες εκπαίδευσης στη Μηχανική Μάθηση, υπάρχει το Distribution Strategy API για κατανεμημένη εκπαίδευση σε διαφορετικές διαμορφώσεις υλικού χωρίς να χρειάζεται αλλαγή του ορισμού του μοντέλου. Το TensorFlow μπορεί να παρουσιαστεί ως ένα επίπεδο υποδομής για διαφοροποιήσιμο προγραμματισμό. Συνδυάζει τέσσερις βασικές ικανότητες:

1. Ικανότητα αποτελεσματικής εκτέλεσης λειτουργιών τανυστή (tensor) χαμηλού επιπέδου σε CPU, GPU ή TPU.

2. Ικανότητα υπολογισμού της κλίσης αυθαίρετων διαφοροποιήσιμων παραστάσεων.

3. Ικανότητα κλιμάκωσης υπολογισμού σε πολλές συσκευές, όπως συμπλέγματα εκατοντάδων GPU.

4. Ικανότητα εξαγωγής προγραμμάτων (γραφήματα) σε εξωτερικούς χρόνους εκτέλεσης

όπως διακομιστές, προγράμματα περιήγησης, κινητές συσκευές και ενσωματωμένες συσκευές.

Το Keras όπως προαναφέρθηκε είναι το API υψηλού επιπέδου του TensorFlow 2. Παρέχει βασικά δομικά στοιχεία για την αποδοτική ανάπτυξη τεχνικών Μηχανικής Μάθησης. Το Keras δίνει τη δυνατότητα στους μηχανικούς και τους ερευνητές να επωφεληθούν πλήρως από τις δυνατότητες επεκτασιμότητας και πολλαπλών πλατφορμών του TensorFlow 2. Μέσω αυτής μπορεί κανείς να εξάγει τα μοντέλα του Keras για εκτέλεση στο πρόγραμμα περιήγησης ή σε μια κινητή συσκευή [14].

Η πλατφόρμα του TensorFlow προσφέρει και τις εφαρμογές του TensorFlow Lite. Το TensorFlow Lite είναι ένα σύνολο εργαλείων που επιτρέπει την εφαρμογή της Μηχανικής Μάθησης σε συσκευές, βοηθώντας έτσι τους προγραμματιστές να εκτελούν τα μοντέλα τους σε κινητές συσκευές, ενσωματωμένες συσκευές ακόμα και σε συσκευές IoT. Ένα μοντέλο TensorFlow Lite αναπαρίσταται σε μια ειδική αποτελεσματική φορητή μορφή γνωστή ως FlatBuffer (αναγνωρίζεται από την επέκταση αρχείου .tflite). Αυτό παρέχει πολλά πλεονεκτήματα σε σχέση με τη μορφή μοντέλου προσωρινής αποθήκευσης πρωτοκόλλου του TensorFlow, όπως μειωμένο μέγεθος και ταχύτερη εξαγωγή συμπερασμάτων. Έτσι λοιπόν κάθε προγραμματιστής μπορεί είτε να δημιουργήσει ένα μοντέλο απευθείας σε TensorFlow Lite μορφή είτε να μετατρέψει ένα TensorFlow μοντέλο σε μορφή TensorFlow Lite μέσω του TensorFlow Lite μετατροπέα (TensorFlow Lite Converter) [52].

3.6.1 Μετατροπέας TensorFlow Lite και μεταδεδομένα

Ο μετατροπέας του TensorFlow Lite, όπως προαναφέρθηκε είναι εργαλείο που παρέχεται από το TensorFlow Lite. Ο TensorFlow Lite μετατροπέας παίρνει ένα μοντέλο TensorFlow και δημιουργεί ένα μοντέλο TensorFlow Lite το οποίο αποτελεί την βελτιστοποιημένη μορφή FlatBuffer που προσδιορίζεται από την επέκταση αρχείου .tflite, όπως προαναφέρθηκε. Υπάρχουν δυο επιλογές για τη χρήση αυτού του μετατροπέα:

1. **Python API:** επιλογή η οποία και συνιστάται. Δηλαδή, τα συσχετισμένα αρχεία μπορούν να συνδυαστούν με κάποιο μοντέλο μέσω της βιβλιοθήκης μεταδεδομένων που παρέχεται από την Python. Αυτό διευκολύνει τη μετατροπή μοντέλων αφού η διαδικασία αυτή πραγματοποιείται ως μέρος της διαδικασίας ανάπτυξης μοντέλων, της εφαρμογής βελτιστοποιήσεων καθώς και της προσθήκης μεταδεδομένων (metadata).

2. **Μέσω της γραμμής εντολών (command line):** επιλογή η οποία υποστηρίζει μόνο τη βασική μετατροπή μοντέλου.

Με την χρήση του Python API μπορείς να μετατρέψεις μοντέλα που βρίσκονται στη μορφή SavedModel, ή μια συνάρτηση concrete, που είναι συναρτήσεις για δημιουργία γράφων από τα προγράμματα Python. Στην συγκεκριμένη περίπτωση ασχολούμαστε με την μετατροπή μοντέλων της βιβλιοθήκης Keras.

Στη συνέχεια, τα μεταδεδομένα TensorFlow Lite παρέχουν ένα πρότυπο για περιγραφές μοντέλων. Τα μεταδεδομένα του TensorFlow Lite είναι μια σημαντική πηγή γνώσης σχετικά με τη λειτουργία ενός μοντέλου και τις πληροφορίες εισόδου και εξόδου του. Τα

μεταδεδομένα αποτελούνται τόσο από αναγνώσιμα από τον άνθρωπο μέρη, που μεταφέρουν την καλύτερη πρακτική κατά τη χρήση του μοντέλου, αλλά και εξαρτήματα αναγνώσιμα από μηχανή. Όλα τα μοντέλα που φιλοξενούνται στο TensorFlow Lite και στο TensorFlow Hub έχουν συμπληρωθεί με μεταδεδομένα. Ορισμένα μοντέλα σε TensorFlow Lite μορφή ενδέχεται να συνοδεύονται από συσχετισμένα αρχεία, όπως αρχεία με ετικέτες (labels) ταξινόμησης που υποδεικνύουν κατηγορίες ή αρχεία λεξιλογίου που αντιστοιχίζουν κομμάτια λέξεων σε αναγνωριστικά λέξεων σε περιπτώσεις μοντέλων ανάλυσης της φυσικής γλώσσας. Αυτά τα αρχεία συνδέονται στο τέλος στο αρχικό αρχείο μοντέλου ως συμπιεσμένο αρχείο (zip). Χωρίς τα συσχετισμένα αρχεία (αν υπάρχουν), ένα μοντέλο δεν θα λειτουργήσει καλά. Τα συσχετισμένα αρχεία μπορούν τώρα να συνδυαστούν με το μοντέλο μέσω της βιβλιοθήκης μεταδεδομένων της Python. Το νέο μοντέλο μετατρέπεται ουσιαστικά σε ένα αρχείο tflite που σαν ένα συμπιεσμένο αρχείο περιέχει τόσο το μοντέλο όσο και τα σχετικά αρχεία. Το παραγόμενο αρχείο επομένως μπορεί να αποσυμπιεστεί με κοινά εργαλεία όπως ακριβώς ένα συμπιεσμένο αρχείο [15].

3.6.2 Λειτουργίες TensorFlow Lite

Το TensorFlow Lite υποστηρίζει έναν αριθμό λειτουργιών (operations) που χρησιμοποιούνται πάνω σε κοινά μοντέλα συμπερασματολογίας (inference models). Καθώς υποβάλλονται σε επεξεργασία από τον μετατροπέα TensorFlow Lite, αυτές οι λειτουργίες ενδέχεται να διαγραφούν ή να συγχωνευθούν, προτού οι υποστηριζόμενες λειτουργίες TensorFlow αντιστοιχιστούν με τις αντίστοιχες λειτουργίες TensorFlow Lite. Δεδομένου ότι η ενσωματωμένη βιβλιοθήκη TensorFlow Lite operators υποστηρίζει μόνο περιορισμένο αριθμό TensorFlow operators, δεν είναι όλα τα μοντέλα μετατρέψιμα από την μια μορφή στην άλλη. Ακόμη και για υποστηριζόμενες λειτουργίες, μερικές φορές αναμένονται πολύ συγκεκριμένα μοτίβα χρήσης, για λόγους απόδοσης. Το σύνολο των υποστηριζόμενων operators αναμένεται να επεκταθεί σε μελλοντικές εκδόσεις του TensorFlow Lite. Ο καλύτερος τρόπος για την κατανόηση του πώς γίνεται η δημιουργία ενός μοντέλου TensorFlow, το οποίο μπορεί να χρησιμοποιηθεί με το TensorFlow Lite, είναι η προσεκτική μελέτη του πώς μετατρέπονται και βελτιστοποιούνται οι operatos, μαζί με τους περιορισμούς που επιβάλλονται από αυτήν τη διαδικασία.

Οι περισσότεροι TensorFlow Lite operators στοχεύουν τόσο σε κινητής υποδιαστολής (float32) όσο και σε κβαντισμένη (uint8, int8) συμπερασματολογία αλλά πολλοί operators δεν κάνουν ακόμη για άλλους τύπους όπως tf.float16 και συμβολοσειρές (strings). Εκτός από τη χρήση διαφορετικής έκδοσης των λειτουργιών, η άλλη διαφορά μεταξύ μοντέλων κινητής υποδιαστολής και κβαντισμένων μοντέλων είναι ο τρόπος μετατροπής τους. Ένας αριθμός λειτουργιών TensorFlow μπορούν να υποβληθούν σε επεξεργασία από το TensorFlow Lite. Ακόμη και ορισμένες υποστηριζόμενες λειτουργίες μπορεί μερικές φορές να αφαιρεθούν μέσω μιας από αυτές τις διαδικασίες. Έτσι λοιπόν υπάρχει το TensorFlow Model Optimization Toolkit το οποίο παρέχει εργαλεία για την βελτιστοποίηση μοντέλων Μηχανικής Μάθησης σχετικά με την ανάπτυξη και την εκτέλεση. Περισσότερες περισσότερες πληροφορίες για τα εργαλεία αυτά θα δοθούν παρακάτω [53].

3.6.3 Βελτιστοποιήσεις TensorFlow Lite

Οι συσκευές άκρων (edge devices) έχουν συχνά περιορισμένη μνήμη ή υπολογιστική ισχύ. Μπορούν να εφαρμοστούν διάφορες βελτιστοποιήσεις σε μοντέλα, ώστε να μπορούν να εκτελεστούν εντός αυτών των περιορισμών. Επιπλέον, ορισμένες βελτιστοποιήσεις επιτρέπουν τη χρήση εξειδικευμένου υλικού για επιταχυνόμενη εξαγωγή συμπερασμών. Το TensorFlow Lite και το TensorFlow Model Optimization Toolkit παρέχουν εργαλεία για την ελαχιστοποίηση της πολυπλοκότητας της βελτιστοποίησης της συμπαρασματολογίας. Γενικά συνιστάται η βελτιστοποίηση ενός μοντέλου κατά τη διαδικασία ανάπτυξης μιας εφαρμογής. Το TensorFlow Lite υποστηρίζει επί του παρόντος τη βελτιστοποίηση μέσω κβαντοποίησης (quantization), κλαδέματος (pruning) και ομαδοποίησης (clustering). Αυτά αποτελούν μέρος του TensorFlow Model Optimization Toolkit, το οποίο παρέχει πόρους για τεχνικές βελτιστοποίησης μοντέλων που είναι συμβατές με το TensorFlow Lite. Βασικοί τρόποι με τους οποίους μπορεί να επιτευχθεί η βελτιστοποίηση του τελικού μοντέλου είναι:

- **Η μείωση του μεγέθους του**. Τα μικρότερα μοντέλα έχουν πολλά πλεονεκτήματα. Συγκεκριμένα, αρχικά έχουν μικρότερο μέγεθος αποθήκευσης στις συσκευές των χρηστών, μικρότερο χρόνο λήψης ενώ, τα μικρότερα μοντέλα χρησιμοποιούν λιγότερη RAM όταν εκτελούνται, γεγονός που ελευθερώνει τη μνήμη για χρήση άλλων τμημάτων της εφαρμογής κάτι που μπορεί να μεταφραστεί σε καλύτερη απόδοση και σταθερότητα.Η μείωση του μεγέθους ενός μοντέλου μπορεί να γίνει μέσω της διαδικασίας της κβαντοποίησης. Επί του παρόντος, η κβαντοποίηση μπορεί να χρησιμοποιηθεί για τη μείωση του χρόνου εκτέλεσης απλοποιώντας τους υπολογισμούς που πραγματοποιούνται κατά τη διάρκεια της εξαγωγής συμπερασμάτων.

- **Η μείωση της καθυστέρησης του μοντέλου**. Η καθυστέρηση είναι ο χρόνος που χρειάζεται για να εκτελεστεί η συμπερασματολογία (inference) με ένα δεδομένο μοντέλο. Ορισμένες μορφές βελτιστοποίησης μπορούν να οδηγήσουν σε μείωση του όγκου του υπολογισμού που απαιτείται για την εκτέλεση της συμπερασματολογίας χρησιμοποιώντας ένα μοντέλο και κατ' επέκταση στη μείωση του χρόνου εκτέλεσης. Η καθυστέρηση μπορεί επίσης να έχει αντίκτυπο στην κατανάλωση ενέργειας.

- **Οι επιταχυντές υλικού**. Επιταχυντές, όπως το Edge TPU, που μπορούν να εκτελέσουν εξαιρετικά γρήγορα συμπερασμούς με μοντέλα που έχουν βελτιστοποιηθεί σωστά. Γενικά, αυτοί οι τύποι συσκευών απαιτούν την κβαντοποίηση των μοντέλων με συγκεκριμένο τρόπο.

Οι βελτιστοποιήσεις μπορούν ενδεχομένως να οδηγήσουν σε αλλαγές στην ακρίβεια του μοντέλου, οι οποίες πρέπει να ληφθούν υπόψη κατά τη διαδικασία ανάπτυξης της εφαρμογής. Οι αλλαγές ακρίβειας εξαρτώνται από το μεμονωμένο μοντέλο που βελτιστοποιείται και είναι δύσκολο να προβλεφθούν εκ των προτέρων. Γενικά, τα μοντέλα που είναι βελτιστοποιημένα για μέγεθος ή καθυστέρηση θα χάσουν ένα μικρό ποσοστό ακρίβειας. Ανάλογα με την εφαρμογή, αυτό μπορεί να επηρεάσει ή να μην επηρεάσει την εμπειρία των χρηστών της. Σε σπάνιες περιπτώσεις, ορισμένα μοντέλα ενδέχεται να αποκτήσουν κάποια ακρίβεια ως αποτέλεσμα της διαδικασίας βελτιστοποίησης.

Η κβαντοποίηση είναι η πιο συνηθισμένη από τις παραπάνω τεχνικές. Συγκεκριμένα, με την κβαντοποίηση η ακρίβεια των αριθμών που χρησιμοποιούνται για την αναπαράσταση των παραμέτρων ενός μοντέλου μειώνεται. Οι παράμετροι αυτοί από προεπιλογή είναι αριθμοί κινητής υποδιαστολής των 32 bit. Η μείωση αυτή έχει ως αποτέλεσμα το μικρότερο μέγεθος μοντέλου και κατ' επέκταση ταχύτερους υπολογισμούς από αυτό. Υπάρχουν διάφοροι τύπου κβαντισμού. Στο TensorFlow Lite διαθέσιμες υπάρχουν οι εξής μέθοδοι:

Post-training float16 quantization

Το TensorFlow Lite υποστηρίζει τη μετατροπή βαρών σε τιμές κινητής υποδιαστολής (float) 16-bit κατά τη μετατροπή του μοντέλου από το TensorFlow στη μορφή επίπεδης προσωρινής μνήμης του TensorFlow Lite μέσω του μετατροπέα. Αυτό έχει ως αποτέλεσμα τη μείωση του μεγέθους του μοντέλου κατά δύο φορές. Σε ορισμένο υλικό, όπως στη GPU, το κβαντισμένο μοντέλο μπορεί να εκτελεστεί σε αυτήν την αριθμητική μειωμένης ακρίβειας, δίνοντας έτσι μια επιτάχυνση σε σχέση με την παραδοσιακή εκτέλεση κινητής υποδιαστολής.

Post-training dynamic range quantization

Το TensorFlow Lite υποστηρίζει τη μετατροπή βαρών σε ακρίβεια των 8 bits ως μέρος της μετατροπής μοντέλου από graphdefs TensorFlow στη μορφή επίπεδης προσωρινής μνήμης του TensorFlow Lite. Η κβαντοποίηση δυναμικού εύρους επιτυγχάνει τέσσερις φορές μείωση στο μέγεθος του μοντέλου. Επιπλέον, το TFLite υποστηρίζει on the fly κβαντοποίηση και αποκβάντωση των ενεργοποιήσεων για να επιτρέψει χρήση κβαντισμένων πυρήνων για ταχύτερη υλοποίηση όταν είναι διαθέσιμος καθώς και ανάμειξη πυρήνων κινητής υποδιαστολής με κβαντισμένους πυρήνες για διαφορετικά μέρη του γραφήματος. Οι ενεργοποιήσεις αποθηκεύονται πάντα σε μορφή κινητής υποδιαστολής. Για τις λειτουργίες που υποστηρίζουν κβαντισμένους πυρήνες, οι ενεργοποιήσεις κβαντίζονται δυναμικά σε ακρίβεια των 8 bit πριν από την επεξεργασία και απο-κβαντοποιούνται με ακρίβεια float μετά την επεξεργασία. Ανάλογα με το μοντέλο που μετατρέπεται, αυτό μπορεί να δώσει μια επιτάχυνση σε σχέση με τον καθαρό υπολογισμό κινητής υποδιαστολής. Επιπλέον, τα βάρη κβαντίζονται μετά την εκπαίδευση και οι ενεργοποιήσεις κβαντίζονται δυναμικά στο συμπέρασμα σε αυτή τη μέθοδο. Επομένως, τα βάρη του μοντέλου δεν επανεκπαιδεύονται για να αντισταθμίσουν τα σφάλματα που προκαλούνται από την κβαντοποίηση έτσι λοιπόν σε αυτή τη περίπτωση είναι σημαντικό να ελεγχθεί η ακρίβεια του κβαντισμένου μοντέλου για να διασφαλιστεί ότι η υποβάθμιση που προκύπτει είναι αποδεκτή.

Post-training integer quantization

Η κβαντοποίηση ακεραίων είναι μια στρατηγική βελτιστοποίησης που μετατρέπει αριθμούς κινητής υποδιαστολής 32 bit (όπως βάρη και έξοδοι ενεργοποίησης) στους πλησιέστερους αριθμούς σταθερού σημείου 8 bit. Αυτό έχει ως αποτέλεσμα ένα μικρότερο μοντέλο με αυξημένη ταχύτητα συμπερασμάτων, κάτι που είναι πολύτιμο για συσκευές χαμηλής κατανάλωσης όπως οι μικροελεγκτές. Αυτή η μορφή δεδομένων απαιτείται επίσης από επιταχυντές μόνο ακέραιου αριθμού όπως το Edge TPU.

Τεχνική	Είσοδος	Βάρη	Ενεργοποιήσεις	Έξοδος	Μέγεθος
Non-quantized model	fp32/int32/int64	fp32	fp32	fp32/int32/int64	-
Dynamic Range (DR)	fp32/int32/int64	int8	fp32	fp32/int32/int64	4x μικρότερο
Integer (INT)	fp32/int32/int64	int8	int8	fp32/int32/int64	4x μικρότερο
Full Integer (FULL)	int8	int8	int8	int8	4x μικρότερο
Float16 (FP16)	fp32/int32/int64	fp16	fp32	fp32/int32/int64	2x μικρότερο

Πίνακας 3.1: *Είδη κβαντοποίησης και χαρακτηριστά*

Quantization-aware training

Αυτό το είδος κβαντισμού είναι συχνά καλύτερο για την ακρίβεια του μοντέλου. Η εκπαίδευση με επίγνωση κβαντισμού μιμείται την κβαντοποίηση σε χρόνο συμπερασματολογίας, δημιουργώντας ένα μοντέλο που θα χρησιμοποιήσουν τα downstream tools για να παράγουν πραγματικά κβαντισμένα μοντέλα. Τα κβαντισμένα μοντέλα χρησιμοποιούν βάρη χαμηλότερης ακρίβειας (π.χ. 8-bit αντί για float 32-bit), που οδηγεί σε οφέλη κατά την ανάπτυξη. Η κβαντοποίηση φέρνει βελτιώσεις μέσω της συμπίεσης του μοντέλου και της μείωσης της καθυστέρησης. Με τις προεπιλογές του API, το μέγεθος του μοντέλου μικραίνει κατά 4 φορές και συνήθως βλέπουμε μεταξύ 1,5 έως 4 φορές βελτίωση στον χρόνο εκτέλεσης στη CPU. Τελικά, βελτιώσεις χρόνου μπορούν να φανούν σε συμβατούς επιταχυντές, όπως το EdgeTPU και το NNAPI. Η τεχνική χρησιμοποιείται στην παραγωγή σε περιπτώσεις χρήσης ομιλίας, όρασης, κειμένου και μετάφρασης [54].

Τα διάφορα είδη κβαντοποίησης, όπως προαναφέρθηκε χρησιμοποιούνται ανάλογα με το είδος του μοντέλου, του υλικού και γενικά της εργασίας που πρέπει να ικανοποιηθεί. Στον Πίνακα 3.1 φαίνονται συνολικά τα χαρακτηριστικά κάθε είδους κβαντοποίησης που είναι διαθέσιμες στο TensorFlow Lite προκειμένου να υπάρχει μια πιο ολοκληρωμένη εικόνα.

3.7 Hugging Face και Transformers

Η Hugging Face είναι μια εταιρία που έχει δημιουργήσει μια διαδικτυακή κοινότητα στην οποία παρέχονται NLP τεχνολογίες ανοιχτού κώδικα. Η κοινότητα αυτή το 2019 συγκέντρωσε 15 εκατομμύρια δολάρια για να δημιουργήσει μια οριστική βιβλιοθήκη NLP. Μέχρι σήμερα, η Hugging Face μπόρεσε να αναπτύξει γρήγορα τεχνογνωσία στην επεξεργασία φυσικής γλώσσας. Στόχος της εταιρείας είναι να προωθήσει το NLP και να το κάνει πιο προσιτό για χρήση από όλους. Σε μια προσπάθεια να διευκολυνθεί η επικοινωνία των ανθρώπων με τις μηχανές, τεχνολογίες όπως το NLP είναι ζωτικής σημασίας. Για παράδειγμα, με το NLP, είναι δυνατό για τους υπολογιστές να διαβάζουν κείμενο, να ακούν ομιλία, να το ερμηνεύουν, να μετρούν συναίσθημα ακόμη και να προσδιορίζουν ποια μέρη του κειμένου ή της ομιλίας είναι σημαντικότερα σε σχέση με άλλα, με στόχο την δημιουργία μιας περίληψης για παράδειγμα. Καθώς όλο και περισσότερες εταιρείες προσθέτουν τεχνολογίες NLP για βελτιωμένες αλληλεπιδράσεις μεταξύ ανθρώπου και μηχανής, καθίσταται επιτακτική ανάγκη

να υπάρχουν έτοιμες βιβλιοθήκες στις οποίες μπορούν εύκολα να εκπαιδευτούν μοντέλα επεξεργασίας φυσικής γλώσσας, εξοικονομώντας χρόνο και κόστος. Αυτός ήταν και ο κύριος στόχος της εταιρίας Hugging Face [55].

Έτσι λοιπόν, η Hugging Face δημιούργησε τη βιβλιοθήκη Transformers η οποία πλέον είναι μια εξαιρετικά δημοφιλής βιβλιοθήκη Python που παρέχει χιλιάδες προεκπαιδευμένα μοντέλα που είναι εξαιρετικά χρήσιμα για μια ποικιλία εργασιών NLP σε διαφορετικούς τρόπους λειτουργίας, όπως κείμενο, όραση και ήχος. Η βιβλιοθήκη Transformers υποστηρίζεται από τις τρεις πιο δημοφιλείς βιβλιοθήκες Βαθιάς Μάθησης τις Jax, PyTorch και TensorFlow, με απρόσκοπτη ενσωμάτωση μεταξύ τους. Προηγουμένως η βιβλιοθήκη Transformers υποστήριζε μόνο το PyTorch αλλά, από τα τέλη του 2019, υποστηρίζεται και το TensorFlow 2, το οποίο περιγράφηκε παραπάνω. Ενώ η βιβλιοθήκη Transformers μπορεί να χρησιμοποιηθεί για πολλές εργασίες, από το Natural Language Inference (NLI) έως την Question-Answering, η ταξινόμηση κειμένου παραμένει μια από τις πιο δημοφιλείς και πρακτικές περιπτώσεις χρήσης της.

Στη συνέχεια, η βιβλιοθήκη ktrain είναι ένα ελαφρύ "περιτύλιγμα" του tf.keras στο TensorFlow 2 η οποία έχει σχεδιαστεί για να κάνει τη Βαθιά Μάθηση και την Τεχνητή Νοημοσύνη πιο προσιτά και ευκολότερα τόσο για ειδικούς στο τομέα όσο και για αρχάριους. Από την έκδοση 0.8, και πάνω το ktrain περιλαμβάνει πλέον μια απλοποιημένη διεπαφή με τους Transformers του Hugging Face για ταξινόμηση κειμένου. Με αυτόν τον τρόπο λοιπόν ο καθένας έχει τη δυνατότητα να δημιουργήσει, να εκπαιδεύσει και να αναπτύξει ένα μοντέλο με την χρήση της βιβλιοθήκης Transformers από Hugging Face σε λίγες μόνο γραμμές κώδικα, ενώ μπορεί να παρουσιάσει την δουλειά του σε όλη την κοινότητα. Έτσι λοιπόν η δουλειά του καθενός μπορεί να χρησιμοποιηθεί ή να εμπλουτιστεί από άλλα μέλη της κοινότητας ή να χρησιμοποιηθεί ως έχει σε άλλες εφαρμογές, όπως στο Android, γεγονός που έχει οδηγήσει στην τόσο μεγάλη ανάπτυξη της κοινότητας και της ίδιας της βιβλιοθήκης με όλο και περισσότερα μοντέλα να προστίθενται συνεχώς.

Τα μοντέλα Transformers μπορούν επίσης να εκτελούν εργασίες με διάφορους τρόπους συνδυασμένους μεταξύ τους, όπως απάντηση σε ερωτήσεις, οπτική αναγνώριση χαρακτήρων, εξαγωγή πληροφοριών από σαρωμένα έγγραφα, ταξινόμηση βίντεο και οπτική απάντηση ερωτήσεων. Επίσης παρέχεται API για τη γρήγορη λήψη και χρήση αυτών των προεκπαιδευμένων μοντέλων σε ένα δεδομένο κείμενο, έτσι ώστε ο καθένας να μπορεί να τα ρυθμίζει με ακρίβεια στα δικά του σύνολα δεδομένων. Ταυτόχρονα, κάθε λειτουργική μονάδα Python που ορίζει μια αρχιτεκτονική είναι πλήρως αυτόνομη και μπορεί να τροποποιηθεί για να επιτρέψει γρήγορα ερευνητικά πειράματα [56, 57].

Κεφάλαιο 4

Διεργασίες, μοντέλα και τεχνικές στην παρούσα εφαρμογή

Στο κεφάλαιο αυτό θα γίνει αρχικά παρουσίαση βασικών τεχνικών προεπεξεργασίας κειμένου προκειμένου να προκύψει η είσοδος στη μορφή που μπορεί να δεχτεί το εκάστοτε Νευρωνικό Δίκτυο. Μια από αυτές τις τεχνικές είναι η διαδικασία τμηματοποίησης που χρησιμοποιείται προκειμένου το κείμενο να μετατραπεί σε είσοδο αριθμητικών διανυσμάτων, τα οποία είναι σε θέση να επεξεργαστεί το μοντέλο, έτσι ώστε να πραγματοποιηθεί η συμπερασματολογία. Στην παρούσα εφαρμογή πραγματοποιείται τμηματοποίηση για κάθε ένα από τα μοντέλα που χρησιμοποιούνται έτσι λοιπόν στο συγκεκριμένο κεφάλαιο γίνεται παρουσίαση όλων των διαφορετικών τεχνικών τμηματοποίησης που χρησιμοποιούνται. Επιπλέον, δίνονται γενικές πληροφορίες για τα μοντέλα που μελετώνται στην παρούσα εφαρμογή καθώς και των εργασιών που μελετώνται.

4.1 Γνωστές αρχιτεκτονικές

4.1.1 BERT

Το BERT (Bidirectional Encoder Representations from transformers) είναι ένα μοντέλο για NLP εργασίες, το οποίο δημοσιεύτηκε σε μία εργασία με τίτλο "BERT: Pre-training of Deep Bidirectional transformers for Language Understanding", από ερευνητές του Google AI Language, το 2019 [58]. Το συγκεκριμένο μοντέλο έχει προκαλέσει σάλο στην κοινότητα της Μηχανικής Μάθησης παρουσιάζοντας αποτελέσματα τελευταίας τεχνολογίας σε μια μεγάλη ποικιλία εργασιών NLP, συμπεριλαμβανομένων της εργασίας Απάντησης Ερωτήσεων (Question Answering) και του Συμπερασμού Φυσικής Γλώσσας (Natural Language Inference). Η βασική τεχνική καινοτομία του BERT είναι η εφαρμογή της αμφίδρομης εκπαίδευσης του transformer, ενός δημοφιλούς μοντέλου προσοχής στη μοντελοποίηση γλώσσας. Αυτό έρχεται σε αντίθεση με προηγούμενες προσπάθειες που εξέταζαν μια ακολουθία κειμένου, είτε από αριστερά προς τα δεξιά είτε συνδύαζαν εκπαίδευση από αριστερά προς τα δεξιά και από δεξιά προς τα αριστερά. Τα αποτελέσματα της εργασίας δείχνουν ότι ένα γλωσσικό μοντέλο που εκπαιδεύεται αμφίδρομα μπορεί να έχει μια βαθύτερη αίσθηση του γλωσσικού πλαισίου και της ροής από τα γλωσσικά μοντέλα μιας κατεύθυνσης.

Δεδομένου ότι ο στόχος του BERT είναι να δημιουργήσει ένα μοντέλο γλώσσας, μόνο ο

μηχανισμός κωδικοποιητή είναι απαραίτητος. Η δομή transformer που χρησιμοποιείται στο BERT θεωρείται αμφίδρομη, αν και θα ήταν πιο ακριβές να πούμε ότι είναι μη κατευθυντική. Αυτό το χαρακτηριστικό επιτρέπει στο μοντέλο να μάθει το πλαίσιο μιας λέξης με βάση όλο το περιβάλλον της (αριστερά και δεξιά της λέξης). Η είσοδος ενός transformer είναι μια ακολουθία από διακριτικά, τα οποία πρώτα ενσωματώνονται σε διανύσματα και στη συνέχεια υποβάλλονται σε επεξεργασία από το Νευρωνικό Δίκτυο. Η έξοδος είναι μια ακολουθία διανυσμάτων, στην οποία κάθε διάνυσμα αντιστοιχεί σε ένα διακριτικό εισόδου με τον ίδιο δείκτη. Κατά την εκπαίδευση μοντέλων γλώσσας, υπάρχει μια πρόκληση για τον καθορισμό ενός στόχου πρόβλεψης.

Πολλά μοντέλα προβλέπουν την επόμενη λέξη με μια σειρά, μια κατευθυντική προσέγγιση που περιορίζει εγγενώς τη μάθηση με βάση το πλαίσιο. Για να ξεπεράσει αυτή την πρόκληση, το BERT χρησιμοποιεί δύο στρατηγικές εκπαίδευσης: το Masked Language Holding (MLM) και το Next Sentence Prediction (NSP) τα οποία αναλύονται στη συνέχεια [7].

Masked Language Holding (MLM)

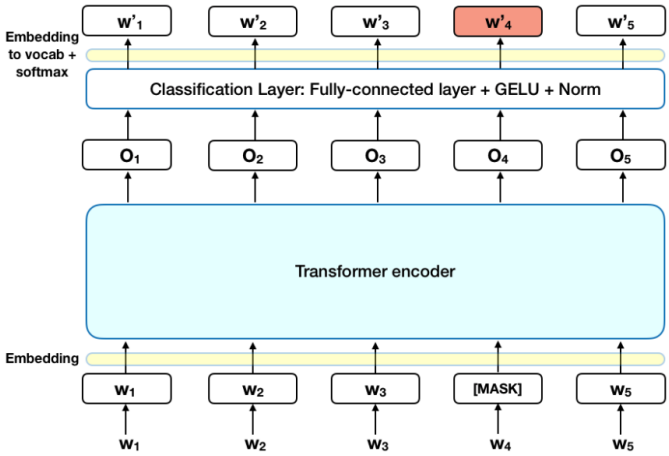
Πριν από την τροφοδοσία ακολουθιών λέξεων στο BERT, το 15% των λέξεων σε κάθε ακολουθία αντικαθίστανται με ένα διακριτικό [MASK], μια διαδικασία που ορίζεται ως Masking. Στη συνέχεια, το μοντέλο επιχειρεί να προβλέψει την αρχική τιμή των καλυμμένων λέξεων, με βάση το πλαίσιο που παρέχεται από τις άλλες, "μη καλυμμένες", λέξεις της ακολουθίας. Σε τεχνικούς όρους, η πρόβλεψη των λέξεων εξόδου απαιτεί:

1. Προσθήκη στρώματος ταξινόμησης πάνω από την έξοδο του κωδικοποιητή.
2. Πολλαπλασιασμός των διανυσμάτων εξόδου με τον πίνακα ενσωμάτωσης, μετατρέποντάς τα στη διάσταση του λεξιλογίου.
3. Υπολογισμός της πιθανότητας κάθε λέξης στο λεξιλόγιο με την συνάρτηση softmax.

Σε ένα γενικό πλαίσιο η συνάρτηση απώλειας του BERT λαμβάνει υπόψη μόνο την πρόβλεψη των masked τιμών και αγνοεί την πρόβλεψη των μη masked λέξεων. Κατά συνέπεια, το μοντέλο συγκλίνει πιο αργά από τα κατευθυντικά μοντέλα, ένα χαρακτηριστικό που αντισταθμίζεται από την αυξημένη επίγνωση του πλαισίου του κειμένου [59]. Στην Εικόνα 4.1 φαίνεται η δομή του MLM.

Next Sentence Prediction (NSP)

Στη διαδικασία εκπαίδευσης του BERT, το μοντέλο λαμβάνει ζεύγη προτάσεων ως είσοδο και μαθαίνει να προβλέπει εάν η δεύτερη πρόταση στο ζεύγος είναι η επόμενη πρόταση στο αρχικό κείμενο. Κατά τη διάρκεια της εκπαίδευσης, το 50% των εισροών είναι ένα ζευγάρι προτάσεων στο οποίο η δεύτερη πρόταση είναι η επόμενη πρόταση στο αρχικό κείμενο, ενώ στο άλλο 50% μια τυχαία πρόταση από το σώμα επιλέγεται ως δεύτερη πρόταση. Η υπόθεση είναι ότι η τυχαία πρόταση θα αποσυνδεθεί από την πρώτη πρόταση. Προκειμένου να βοηθηθεί το μοντέλο στο να διακρίνει μεταξύ των δύο προτάσεων στην εκπαίδευση, η εισαγωγή υποβάλλεται σε επεξεργασία με τον ακόλουθο τρόπο πριν εισέλθει στο μοντέλο.



Εικόνα 4.1: *Διαδικασία MLM [7]*

- Ένα διακριτικό [CLS] εισάγεται στην αρχή της πρώτης πρότασης και ένα διακριτικό [SEP] στο τέλος κάθε πρότασης.
- Οι προτάσεις διαχωρίζονται μεταξύ τους από αντίστοιχα διανύσματα αναπαράστασης προτάσεων (sentence embeddings) τα οποία προστίθενται για να υποδείξουν τις θέσεις των διακριτικών κάθε πρότασης.
- Τέλος, προστίθεται ένα διάνυσμα αναπαράστασης θέσης (positional embedding) σε κάθε διακριτικό για να υποδείξει τη θέση του στην ακολουθία.

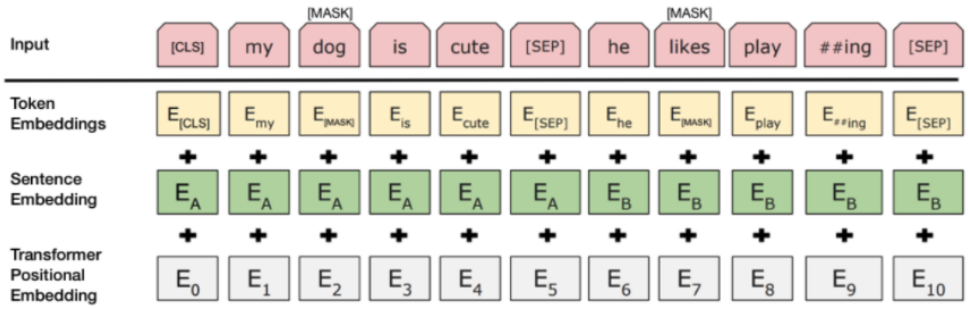
Για να προβλεφθεί εάν η δεύτερη πρόταση είναι όντως συνδεδεμένη με την πρώτη, εκτελούνται τα ακόλουθα βήματα:

- Ολόκληρη η ακολουθία εισόδου περνά από το μοντέλο του transformer.
- Η έξοδος του διακριτικού [CLS] μετατρέπεται σε ένα διάνυσμα σχήματος 2×1, χρησιμοποιώντας ένα απλό στρώμα ταξινόμησης (μαθημένες μήτρες βαρών και προκαταλήψεων).
- Υπολογισμός της πιθανότητας IsNextSequence με τη συνάρτηση softmax [7].

Κατά την εκπαίδευση του μοντέλου BERT, το Masked LM και το Next Sentence Prediction εκπαιδεύονται μαζί, με στόχο την ελαχιστοποίηση της συνδυασμένης συνάρτησης απώλειας των δύο στρατηγικών [59].

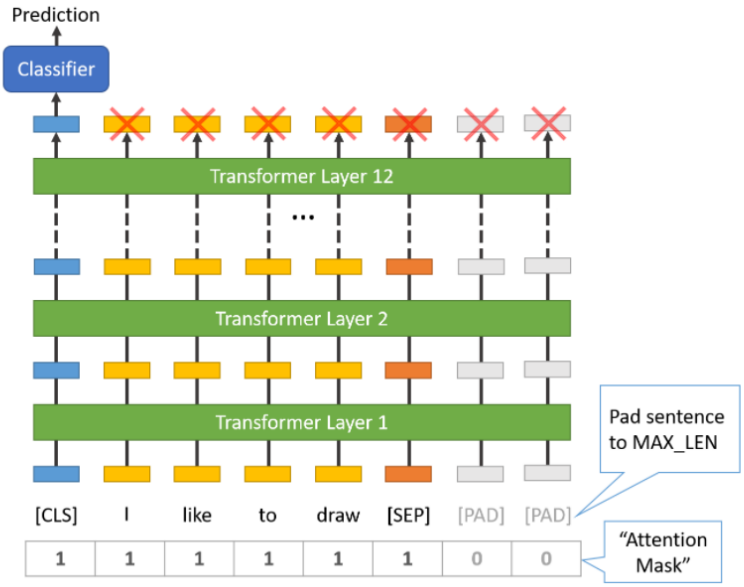
Προκειμένου να γίνει μια οπτικοποίηση, παρουσιάζεται η βασική δομή του BERT, η οποία είναι η εξής (4.3):

- **Ένα επίπεδο εισόδου**, όπου λαμβάνει τα διανύσματα εισόδου (embeddings).
- **Τα επίπεδα transformers**, με τα κεφάλια αυτοπροσοχής (self attention heads). Κάθε transformer παίρνει μια λίστα με διανυσματικές αναπαραστάσεις από tokens και παράγει τον ίδιο αριθμό διανυσμάτων στην έξοδο, προφανώς με αλλαγμένες τιμές των χαρακτηριστικών.



Εικόνα 4.2: *Είσοδος στο BERT μοντέλο [7]*

- **Το επίπεδο εξόδου**, παράγεται μια λίστα από διανύσματα αναπαράστασης tokens του ίδιου μέγεθος με την είσοδο. Συγκεκριμένα, η έξοδος του BERT είναι το διάνυσμα κρυφής κατάστασης προκαθορισμένου κρυφού μεγέθους (αυτό της εισόδου) [60]. Το πρώτο token κάθε ακολουθίας εισόδου είναι πάντα ένα ειδικό token ταξινόμησης ([CLS]). Η τελική κρυφή κατάσταση που αντιστοιχεί σε αυτό το token χρησιμοποιείται ως η αθροιστική αναπαράσταση ακολουθίας για εργασίες ταξινόμησης [8].



Εικόνα 4.3: *Βασική δομή του BERT [8]*

Με βάση το βάθος της αρχιτεκτονικής του μοντέλου, εισάγονται δύο τύποι μοντέλων BERT, το BERT_{BASE} και το BERT_{LARGE}. Το μοντέλο BERT_{BASE} χρησιμοποιεί 12 στρώματα από transformers με κρυφό μέγεθος 768 και αριθμό κεφαλών αυτοπροσοχής (self-attention heads) 12 και έχει περίπου 110 εκατομμύρια παραμέτρους που μπορούν να εκπαιδευτούν. Από την άλλη πλευρά, το BERT_{LARGE} χρησιμοποιεί 24 στρώματα από transformers με κρυφό μέγεθος 1024 και αριθμό κεφαλών αυτοπροσοχής 16 και έχει περίπου 340 εκατομμύρια παραμέτρους που μπορούν να εκπαιδευτούν. Το BERT χρησιμοποιεί την ίδια αρχιτεκτονική μοντέλου για όλες τις εργασίες με ελάχιστη αλλαγή, όπως η προσθήκη επιπέδου εξόδου για ταξινόμηση όπως φαίνεται και στην Εικόνα 4.3 [60]. Τα μοντέλα που περιγράφονται

και χρησιμοποιούνται σε NLP εργασίες έχουν δυστυχώς πολύ μεγάλο μέγεθος, γεγονός που τα καθιστά ακατάλληλα για χρήση σε κινητές συσκευές που έχουν περιορισμένους πόρους. Για αυτό το λόγο στην περίπτωση του BERT δημιουργήθηκε μια άλλη εκδοχή του η οποία προτάθηκε για χρήση σε κινητές συσκευές, το MobileBERT, το οποίο αναλύεται στη συνέχεια [61].

4.1.2 MobileBERT

Το MobileBERT παρουσιάστηκε στην εργασία με τίτλο "MobileBERT: a Compact Task – Agnostic BERT for Resource - Limited Devices", από τους Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, και Denny Zhou, το 2020 [9]. Συγκεκριμένα, το MobileBERT είναι ένα αμφίδρομο μοντέλο από την βιβλιοθήκη transformers, που προφανώς βασίζεται στο BERT αλλά είναι ελαφρύτερο και πιο γρήγορο με την χρήση διαφόρων τεχνικών και προσεγγίσεων [9]. Συγκεκριμένα, το MobileBERT είναι μια λεπτή εκδοχή του BERT_{LARGE}, ενώ είναι εξοπλισμένο με bottleneck δομές και μια προσεκτικά σχεδιασμένη ισορροπία μεταξύ των δικτύων αυτοπροσοχής και τροφοδοσίας. Για να γίνει η εκπαίδευση του MobileBERT, πρώτα εκπαιδεύεται ένα ειδικά σχεδιασμένο μοντέλο δασκάλου και συγκεκριμένα ένα μοντέλο BERT_{LARGE} με αντίστροφη συμφόρηση. Ο όρος "δάσκαλος" προέρχεται από την διαδικασία της απόσταξης γνώσης. Συγκεκριμένα, απόσταξη γνώσης είναι μέθοδος συμπίεσης ενός μοντέλου κατά την οποία ένα μικρό μοντέλο εκπαιδεύεται να μιμείται ένα προεκπαιδευμένο, μεγαλύτερο μοντέλο (ή σύνολο μοντέλων). Αυτή η ρύθμιση κατάρτισης αναφέρεται μερικές φορές ως "δάσκαλος-μαθητής", όπου το μεγάλο μοντέλο είναι ο δάσκαλος και το μικρό μοντέλο είναι ο μαθητής. Στην απόσταξη, η γνώση μεταφέρεται από το μοντέλο του δασκάλου στον μαθητή ελαχιστοποιώντας μια συνάρτηση απώλειας στην οποία ο στόχος είναι η κατανομή των πιθανοτήτων τάξης που προβλέπεται από το μοντέλο δασκάλου [61].

Στην παρούσα περίπτωση πραγματοποιείται μεταφορά γνώσης από τον δάσκαλο, δηλαδή το BERT, στον μαθητή, δηλαδή το MobileBERT. Εμπειρικές μελέτες δείχνουν ότι το MobileBERT είναι 4,3 φορές μικρότερο και 5,5 φορές ταχύτερο από το BERT_{BASE} επιτυγχάνοντας ανταγωνιστικά αποτελέσματα σε γνωστά σημεία αναφοράς. Τόσο το BERT όσο και το MobileBERT είναι "task-agnostic" μοντέλα, το οποίο σημαίνει ότι με μικρές αλλαγές μπορούν να εφαρμοστούν σε οποιοδήποτε NLP εργασία.

Σχετικά με τις εργασίες συμπερασμάτων φυσικής γλώσσας του GLUE, το MobileBERT επιτυγχάνει συνολικά σε αυτές μια GLUE βαθμολογία ίση με 77.7 (0,6 χαμηλότερη από το BERT_{BASE}) [9]. Προκειμένου να υπολογιστεί η GLUE βαθμολογία το μοντέλο εκπαιδεύεται και στις 9 εργασίες που απαρτίζουν το GLUE και στην συνέχεια υπολογίζεται ο μέσος όρος των βαθμολογιών του σε κάθε μια από αυτές [62]. Στο SquAD v1.1/v2.0 σύνολο δεδομένων για την εργασία της Απάντησης Ερώτησης το MobileBERT επιτυγχάνει, βαθμολογία F1 score 90,0/79,2 (1,5/2,1 υψηλότερο από το BERT_{BASE}) στο dev σύνολο δεδομένων. Το F1-score, είναι ένα μέτρο της ακρίβειας ενός μοντέλου σε ένα κάποιο δεδομένων. Συγκεκριμένα, είναι ένας τρόπος συνδυασμού της ακρίβειας και της ανάκλησης του μοντέλου και ορίζεται ως ο αρμονικός μέσος όρος της ακρίβειας και της ανάκλησης του μοντέλου. Από εδώ και πέρα θα το δούμε να χρησιμοποιείται στη παρούσα μελέτη ως μέσο σύγκρισης των μοντέλων [63]. Τα αποτελέσματα του MobileBERT σε σχέση με άλλα μοντέλα στο GLUE φαίνονται στην 4.4.

Τα αποτελέσματα αυτά δίνονται στην εργασίας παρουσίασης του μοντέλου. Ο αριθμός κάτω από κάθε εργασία υποδηλώνει τον αριθμό εισόδων κατά την εκπαίδευση. Το σύμβολο ¨†¨ υποδηλώνει ότι μπορεί να είναι άδικο να συγκρίνουμε απευθείας το MobileBERT με αυτά τα μοντέλα αφού το MobileBERT είναι task-agnostic συμπιεσμένο μοντέλο ενώ αυτά τα μοντέλα χρησιμοποιούν το μοντέλο του δασκάλου στο στάδιο του fine-tuning [9].

	#Params	#FLOPS	Latency	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m/mm	QNLI	RTE	GLUE
				8.5k	67k	3.7k	5.7k	364k	393k	108k	2.5k	
ELMo-BiLSTM-Attn	-	-	-	33.6	90.4	84.4	72.3	63.1	74.1/74.5	79.8	58.9	70.0
OpenAI GPT	109M	-	-	47.2	93.1	87.7	84.8	70.1	80.7/80.6	87.2	69.1	76.9
$\text{BERT}_{\text{BASE}}$	109M	22.5B	342 ms	**52.1**	**93.5**	**88.9**	**85.8**	71.2	**84.6/83.4**	90.5	66.4	78.3
$\text{BERT}_{\text{BASE}}$-6L-PKD*	66.5M	11.3B	-	-	92.0	85.0	-	70.7	81.5/81.0	89.0	65.5	-
$\text{BERT}_{\text{BASE}}$-4L-PKD†*	52.2M	7.6B	-	24.8	89.4	82.6	79.8	70.2	79.9/79.3	85.1	62.3	-
$\text{BERT}_{\text{BASE}}$-3L-PKD*	45.3M	5.7B	-	-	87.5	80.7	-	68.1	76.7/76.3	84.7	58.2	-
$\text{DistilBERT}_{\text{BASE}}$-6L†	62.2M	11.3B	-	-	92.0	85.0		70.7	81.5/81.0	89.0	65.5	-
$\text{DistilBERT}_{\text{BASE}}$-4L†	52.2M	7.6B	-	32.8	91.4	82.4	76.1	68.5	78.9/78.0	85.2	54.1	-
TinyBERT*	14.5M	1.2B	-	43.3	92.6	86.4	79.9	**71.3**	82.5/81.8	87.7	62.9	75.4
$\text{MobileBERT}_{\text{TINY}}$	15.1M	3.1B	40 ms	46.7	91.7	87.9	80.1	68.9	81.5/81.6	89.5	65.1	75.8
MobileBERT	25.3M	5.7B	62 ms	50.5	92.8	88.8	84.4	70.2	83.3/82.6	90.6	66.2	77.7
MobileBERT w/o OPT	25.3M	5.7B	192 ms	51.1	92.6	88.8	84.8	70.5	84.3/**83.4**	**91.6**	**70.4**	**78.5**

Εικόνα 4.4: *Το MobileBERT στο σημείο αναφοράς GLUE [9]*

4.1.3 RoBERTa

Στη συνέχεια, θα παρουσιαστεί το μοντέλο RoBERTa. Το συγκεκριμένο μοντέλο, προτάθηκε στην εργασία με τίτλο "RoBERTa: A Robustly Optimized BERT Pretraining Approach", από τους Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer και Veselin Stoyanov, το 2019 [10]. Η συγκεκριμένη εργασία απέδειξε ότι το μοντέλο BERT είναι υποεκπαιδευμένο. Έως τότε θεωρούνταν ότι τα σύνολα δεδομένων των BooksCorpus και της Αγγλικής Wikipedia (συνολικά 16 GB) για την προεκπαίδευση του μοντέλου ήταν αρκετά για ένα γλωσσικό μοντέλο προκειμένου να αποκτήσει τη βασική κατανόηση της γλώσσας, αλλά δεν ήταν έτσι [64]. Η βελτιστοποιημένη μέθοδος στο RoBERTa, παράγει αποτελέσματα τελευταίας τεχνολογίας στο ευρέως χρησιμοποιούμενο σημείο αναφοράς NLP, το GLUE. Συγκεκριμένα, αποτελέσματα του RoBERTa στο GLUE σε σχέση με το BERT καθώς και άλλα μοντέλα που όπως δίνονται στην εργασία παρουσίασγς του μοντέλου δίνονται στην Εικόνα 4.5. Όλα τα αποτελέσματα βασίζονται σε μια αρχιτεκτονική 24 επιπέδων (large). Τα αποτελέσματα του RoBERTa στο dev set είναι ο (Median) μεταξύ πέντε εκτελέσεων και στο test set είναι σύνολα μοντέλων μίας εργασίας [10].

Πιο συγκεκριμένα, το RoBERTa μοντέλο είναι επίσης πανομοιότυπο με το BERT_{BASE} μοντέλο όσον αφορά την μορφή του, έχει δηλαδή 12 επίπεδα μετασχηματιστών με κρυφό μέγεθος 768 και αριθμό κεφαλών αυτοπροσοχής 12 και έχει περίπου 110 εκατομμύρια εκπαιδεύσιμες παραμέτρους [10]. Η ουσιαστική διαφορά αυτού του μοντέλου λοιπόν είναι η εκπαίδευση του. Σχετικά με την εκπαίδευση του ερευνητές του Facebook και του Πανεπιστημίου της Ουάσιγκτον, συγγραφείς της εργασίας, εκπαίδευσαν τον RoBERTa σε ένα σύνολο 160 GB μη συμπιεσμένων αγγλικών δεδομένων [64]. Το RoBERTa, το οποίο υλοποιήθηκε

	MNLI	QNLI	QQP	RTE	SST	MRPC	CoLA	STS	WNLI	Avg
Single-task single models on dev										
$\text{BERT}_{\text{LARGE}}$	86.6/-	92.3	91.3	70.4	93.2	88.0	60.6	90.0	-	-
$\text{XLNet}_{\text{LARGE}}$	89.8/-	93.9	91.8	83.8	95.6	89.2	63.6	91.8	-	-
RoBERTa	**90.2/90.2**	**94.7**	**92.2**	**86.6**	**96.4**	**90.9**	**68.0**	**92.4**	**91.3**	-
Ensembles on test (from leaderboard as of July 25, 2019)										
ALICE	88.2/87.9	95.7	**90.7**	83.5	95.2	92.6	**68.6**	91.1	80.8	86.3
MT-DNN	87.9/87.4	96.0	89.9	86.3	96.5	92.7	68.4	91.1	89.0	87.6
XLNet	90.2/89.8	98.6	90.3	86.3	**96.8**	**93.0**	67.8	91.6	**90.4**	88.4
RoBERTa	**90.8/90.2**	**98.9**	90.2	**88.2**	96.7	92.3	67.8	**92.2**	89.0	**88.5**

Εικόνα 4.5: *Το RoBERTa στο σημείο αναφοράς GLUE [10]*

στο PyTorch, τροποποιεί βασικές υπερπαραμέτρους στο BERT, συμπεριλαμβανομένης της κατάργησης του NSP, ενώ γίνεται εκπαίδευση με πολύ μεγαλύτερο mini-batch και μεγαλύτερους ρυθμούς εκμάθησης (learning rates). Αυτό επιτρέπει στο RoBERTa να βελτιώσει το μοντέλο μάσκας (Masking). Επιπλέον η εκπαίδευση του RoBERTa γίνεται με μια τάξη μεγέθους περισσότερα δεδομένα από το BERT, και σε μεγαλύτερο χρονικό διάστημα. Επίσης για την εκπαίδευση χρησιμοποιήθηκε υπάρχοντα σύνολα δεδομένων χωρίς ετικέτες για NLP καθώς και CC-News, ένα νέο σύνολο δεδομένων που προέρχεται από δημόσια άρθρα ειδήσεων.

Μετά την εφαρμογή αυτών των αλλαγών σχεδιασμού, το μοντέλο αυτό παρείχε κορυφαίες επιδόσεις στα σύνολα δεδομένων MNLI, QNLI, RTE, STS-B και RACE και σημαντική βελτίωση της απόδοσης στο σημείο αναφοράς GLUE. Με GLUE βαθμολογία 88,5 το RoBERTa έφτασε στην πρώτη θέση στον πίνακα κορυφαίων μοντέλων στο GLUE, ταιριάζοντας με την απόδοση του προηγούμενου ηγέτη, XLNet-Large. Αυτά τα αποτελέσματα υπογραμμίζουν τη σημασία των ανεξερεύνητων επιλογών σχεδίασης στην εκπαίδευση του BERT και βοηθούν στην αποσύνδεση των σχετικών συνεισφορών του μεγέθους των δεδομένων, του χρόνου εκπαίδευσης και των στόχων προεκπαίδευσης.

Τα αποτελέσματα της συγκεκριμένης εργασίας μας δείχνουν ότι ο συντονισμός της εκπαιδευτικής διαδικασίας BERT μπορεί να βελτιώσει σημαντικά την απόδοσή της σε μια ποικιλία εργασιών NLP, ενώ υποδεικνύει επίσης ότι αυτή η συνολική προσέγγιση παραμένει ανταγωνιστική με εναλλακτικές προσεγγίσεις. Ευρύτερα, αυτή η έρευνα καταδεικνύει περαιτέρω τη δυνατότητα τεχνικών αυτο-επίβλεψης εκπαίδευσης να ταιριάζουν ή να υπερβαίνουν την απόδοση πιο παραδοσιακών, εποπτευόμενων προσεγγίσεων. Το RoBERTa αποτελεί μέρος της συνεχούς δέσμευσης του Facebook για την προώθηση της τελευταίας τεχνολογίας σε αυτοεποπτευόμενα συστήματα που μπορούν να αναπτυχθούν με λιγότερη εξάρτηση από την επισήμανση δεδομένων με ένταση χρόνου και πόρων [65]. Τεχνικές του RoBERTa χρησιμοποιήθηκαν σε πολλά επόμενα μοντέλα όπως το DistilBERT.

4.1.4 DistilBERT

Το DistilBERT είναι ένα μοντέλο που δημοσιεύτηκε στην εργασία με τίτλο "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", από τους Victor Sanh,

Lysandre Debut, Julien Chaumond και Thomas Wolf, το 2019 [11]. Σε εργασία αυτή προτείνεται μια μέθοδος για την προεκπαίδευση ενός μικρότερου μοντέλου αναπαράστασης γλώσσας γενικής χρήσης, που ονομάζεται DistilBERT, το οποίο μπορεί στη συνέχεια να βελτιωθεί με καλές επιδόσεις σε ένα ευρύ φάσμα εργασιών, όπως τα μεγαλύτερα μοντέλα αντίστοιχα. Ενώ οι περισσότερες προηγούμενες εργασίες διερεύνησαν τη χρήση της απόσταξης (distillation) για την κατασκευή μοντέλων ειδικών εργασιών, στην συγκεκριμένη περίπτωση γίνεται αξιοποίηση της απόσταξη γνώσης κατά τη φάση προεκπαίδευσης και φαίνεται έτσι από αυτή ότι είναι δυνατό να μειωθεί το μέγεθος ενός μοντέλου BERT κατά 40%, διατηρώντας το 97% της ικανότητες κατανόησης γλώσσας ενώ είναι 60% ταχύτερο. Στην συγκεκριμένη εργασία λοιπόν, φαίνεται ότι είναι δυνατό να επιτευχθούν παρόμοιες επιδόσεις σε πολλές downstream εργασίες, δηλαδή εργασίες επιβλεπόμενης μάθησης που χρησιμοποιούν ένα προεκπαιδευμένο μοντέλο ή έτοιμα components. Αυτό γίνεται μέσω πολύ μικρότερων μοντέλων με την χρήση της απόσταξης γνώσης με αποτέλεσμα μοντέλα που είναι ελαφρύτερα και πιο γρήγορα στο χρόνο συμπερασματολογίας, ενώ απαιτούν επίσης μικρότερο υπολογιστικό εκπαιδευτικό προϋπολογισμό [11].

Έτσι λοιπόν στο DistilBERT πραγματοποιείται η διαδικασία της απόσταξης γνώσης και έτσι παράγεται ένα μικρότερο και γρηγορότερο μοντέλο μαθητής με δάσκαλο το BERT μοντέλο [61]. Συγκεκριμένα, προκειμένου να αξιοποιηθούν οι επαγωγικές προκαταλήψεις που μαθαίνουν τα μεγαλύτερα μοντέλα κατά τη διάρκεια της προεκπαίδευσης, γίνεται εισαγωγή μιας τριπλής απώλειας που συνδυάζει τη μοντελοποίηση γλώσσας, την απόσταξη και τις απώλειες συνημιτονικής απόστασης [66]. Ο μαθητής που προκύπτει στην συγκεκριμένη περίπτωση λοιπόν είναι μια μικρή έκδοση του BERT στην οποία έχουν αφαιρεθεί τα διανύσματα αναπαράστασης γνώσης στηριζόμενα σε tokens, καθώς και το στάδιο εφαρμογής της γραμμικής συνάρτησης (στάδιο NSP), ενώ η υπόλοιπη αρχιτεκτονική είναι πανομοιότυπη με βασικές διαφορές ότι μειώθηκε ο αριθμός των παραμέτρων κατά δύο φορές από το BERT_{BASE} καθώς και ότι διατηρεί το 95% των επιδόσεων του BERT στο σημείο αναφοράς (benchmark) για την κατανόηση γλώσσας στο GLUE [66].

Μερικά από τα αποτελέσματα στα οποία φαίνονται οι διαφορές στην επίδοση του DistilBERT σε σχέση με το BERT, στη συλλογή συνόλων δεδομένων GLUE φαίνονται στην Εικόνα 4.6. Τα αποτελέσματα του ELMo δίνονται όπως αναφέρθηκαν από τους συγγραφείς. Το ELMo είναι ένας νέος τρόπος αναπαράστασης λέξεων σε διανύσματα ELMo(vectors) ή embeddings [67]. Αυτές οι ενσωματώσεις λέξεων είναι χρήσιμες για την επίτευξη αποτελεσμάτων τελευταίας τεχνολογίας σε διάφορες εργασίες NLP. Τα αποτελέσματα των BERT και DistilBERT είναι ο Median 5 εκτελέσεων των μοντέλων με 5 διαφορετικές εισόδους [11, 67].

Model	**Score**	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SST-2	STS-B	WNLI
ELMo	68.7	44.1	68.6	76.6	71.1	86.2	53.4	91.5	70.4	56.3
BERT-base	79.5	56.3	86.7	88.6	91.8	89.6	69.3	92.7	89.0	53.5
DistilBERT	77.0	51.3	82.2	87.5	89.2	88.5	59.9	91.3	86.9	56.3

Εικόνα 4.6: *Το DistilBERT στο dev set του σημείου αναφοράς GLUE [11]*

Η εκπαίδευση ενός υποδικτύου δεν αφορά μόνο την αρχιτεκτονική, αφορά επίσης στη εύρεση της σωστής αρχικοποίησης για τη σύγκλιση του υποδικτύου. Προκειμένου να επι-

τευχθεί σύγκλιση, ο μαθητή μας, DistilBERT, αρχικοποιήθηκε από τον δάσκαλό του, BERT, παίρνοντας ένα στρώμα από τα δύο του, αξιοποιώντας το κοινό κρυφό μέγεθος μεταξύ μαθητή και δασκάλου. Επιπλέον κατά την διαδικασία εκπαίδευσης του DistilBERT χρησιμοποιήθηκαν επίσης τεχνικές που παρουσιάστηκαν στο μοντέλο RoBERTa που έδειξαν ότι ο τρόπος που εκπαιδεύεται το BERT είναι καθοριστικός για την τελική του απόδοση. Συγκεκριμένα, ακολουθώντας το RoBERTa, έγινε εκπαίδευση του DistilBERT με μεγάλο αριθμό δειγμάτων που μπορούν να διαδοθούν μέσω του δικτύου (batch size), που αξιοποιούν τη συσσώρευση διαβάθμισης (gradient accumulation), με δυναμικό Masking και αφαίρεση του NSP [66].

4.1.5 ELECTRA

Το τελευταίο μοντέλο που θα ασχοληθούμε στη παρούσα διπλωματική εργασία είναι το ELECTRA. Το μοντέλο ELECTRA προτάθηκε στην εργασία με τίτλο "ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators", από τους Kevin Clark, Minh-Thang Luong, Quoc V. L και τον Christopher D. Manning, το 2020 [12]. Η προσέγγιση του ELECTRA είναι ουσιαστικά μια νέα προσέγγιση προεκπαίδευσης στην οποία το εκπαιδεύονται δύο μοντέλα transformers: η γεννήτρια και ο διαχωριστής. Ο ρόλος της γεννήτριας είναι να αντικαθιστά τα διακριτικά σε μια ακολουθία με τα διακριτικά τύπου [MASK], ότι γίνεται δηλαδή στην εκπαίδευση με Masking. Ο διαχωριστής, που είναι το μοντέλο για το οποίο ενδιαφερόμαστε, και για αυτό πραγματοποιούμε την εκπαιδευτική διαδικασία, στη συνέχεια προσπαθεί να προσδιορίσει ποια διακριτικά αντικαταστάθηκαν από τη γεννήτρια στην ακολουθία. Με λίγα λόγια δεν είναι το μοντέλο που μας ενδιαφέρει αυτό που εκτελεί το Masking αλλά η γεννήτρια.

Πιο αναλυτικά, και στην περίπτωση του ELECTRA χρησιμοποιείται το ίδιο μέγεθος μοντέλου καθώς και το ίδιο μέγεθος δεδομένων, όπως και το BERT [12]. Ωστόσο, μέθοδοι προεκπαίδευσης όπως το MLM, το οποίο ,όπως αναφέρθηκε, εφαρμόζεται στο BERT, δεν χρησιμοποιούνται αφού παρατηρήθηκε ότι καταστρέφουν την είσοδο αντικαθιστώντας ορισμένα διακριτικά με το διακριτικό [MASK], γεγονός που στη συνέχεια οδηγεί στην εκπαίδευση ενός μοντέλου για την ανακατασκευή των αρχικών διακριτικών. Ενώ το Masking οδηγεί σε καλά αποτελέσματα όταν μεταφέρεται σε downstream εργασίες NLP, γενικά απαιτεί μεγάλες ποσότητες υπολογισμού για να είναι αποτελεσματικό. Ως εναλλακτική, στην συγκεκριμένη εργασία προτείνεται μια πιο αποδοτική ως προς το δείγμα εργασία προεκπαίδευσης που ονομάζεται ανίχνευση αντικατασταθέντος διακριτικού.

Συγκεκριμένα, αντί να γίνεται Masking στην είσοδο, η προσέγγισή γίνεται αντικαθιστώντας ορισμένα διακριτικά με εύλογες εναλλακτικές λύσεις που έχουν ληφθεί από ένα μικρό δίκτυο παραγωγής. Στη συνέχεια, αντί να γίνεται εκπαίδευση ενός μοντέλου που προβλέπει τις αρχικές ταυτότητες των κατεστραμμένων διακριτικών, γίνεται εκπαίδευση ενός διακριτικού μοντέλου που προβλέπει εάν κάθε διακριτικό στην κατεστραμμένη είσοδο αντικαταστάθηκε από ένα δείγμα γεννήτριας ή όχι. Ενδελεχή πειράματα αποδεικνύουν ότι αυτή η νέα εργασία προεκπαίδευσης είναι πιο αποτελεσματική από το MLM επειδή η εργασία ορίζεται σε όλα τα διακριτικά εισόδου και όχι μόνο στο μικρό υποσύνολο που καλύφθηκε. Ως αποτέλεσμα, οι αναπαραστάσεις με βάση τα συμφραζόμενα που μαθαίνουμε από την προσέγγισή μας ξεπερνούν σημαντικά αυτές που μαθαίνει ο BERT, δεδομένου του ίδιου μεγέθους

μοντέλου, δεδομένων και υπολογισμού. Τέλος, το μοντέλο αυτό έχει συγκρίσιμες επιδόσεις με το RoBERTa και το XLNet, ενώ χρησιμοποιεί λιγότερο από το 1/4 του υπολογισμού τους και τα ξεπερνά όταν χρησιμοποιούν τον ίδιο όγκο υπολογισμού [12]. Στην Εικόνα 4.7 φαίνονται μετρήσεις του ELECTRA σε σχέση με των άλλων μοντέλων που προαναφέρθηκαν όπως παρουσιάστηκαν στην εργασία παρουσίασης του μοντέλου [12].

Model	Train / Infer FLOPs	Speedup	Params	Train Time + Hardware	GLUE
ELMo	3.3e18 / 2.6e10	19x / 1.2x	96M	14d on 3 GTX 1080 GPUs	71.2
GPT	4.0e19 / 3.0e10	1.6x / 0.97x	117M	25d on 8 P6000 GPUs	78.8
BERT-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	75.1
BERT-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	82.2
ELECTRA-Small	1.4e18 / 3.7e9	45x / 8x	14M	4d on 1 V100 GPU	79.9
50% trained	7.1e17 / 3.7e9	90x / 8x	14M	2d on 1 V100 GPU	79.0
25% trained	3.6e17 / 3.7e9	181x / 8x	14M	1d on 1 V100 GPU	77.7
12.5% trained	1.8e17 / 3.7e9	361x / 8x	14M	12h on 1 V100 GPU	76.0
6.25% trained	8.9e16 / 3.7e9	722x / 8x	14M	6h on 1 V100 GPU	74.1
ELECTRA-Base	6.4e19 / 2.9e10	1x / 1x	110M	4d on 16 TPUv3s	85.1

Εικόνα 4.7: *Σύγκριση ELECTRA στο GLUE dev set [12]*

4.2 Διεργασίες στην παρούσα εφαρμογή

4.2.1 Απάντηση Ερωτήσεων

Η απάντηση ερωτήσεων είναι η εργασία κατά την οποία ένα Νευρωνικό Δίκτυο Επεξεργασίας Φυσικής Γλώσσας απαντά σε ερωτήσεις (συνήθως ερωτήσεις κατανόησης ανάγνωσης) με βάση κάποιο πλαίσιο που μπορεί να είναι μια πρόταση ή κάποιο κείμενο, ενώ απέχει όταν παρουσιάζεται μια ερώτηση που δεν μπορεί να απαντηθεί με βάση το παρεχόμενο του πλαισίου αυτού. Τα δημοφιλή σύνολα δεδομένων αναφοράς για την εργασία απάντησης ερωτήσεων είναι τα SQuAD, HotPotQA, bAbI, TriviaQA, WikiQA και πολλά άλλα. Τα μοντέλα για απάντηση ερωτήσεων συνήθως αξιολογούνται με μετρικές όπως το EM και το F1. Μερικά πρόσφατα μοντέλα με κορυφαίες επιδόσεις είναι τα T5 και XLNet [68].

4.2.2 Ταξινόμηση Κειμένου

Η ταξινόμηση κειμένου είναι η εργασία κατά την οποία ένα Νευρωνικό Δίκτυο Επεξεργασίας Φυσικής Γλώσσας αναθέτει σε μία πρόταση ή ένα κείμενο μια κατάλληλη κατηγορία (class). Οι κατηγορίες εξαρτώνται από το επιλεγμένο σύνολο δεδομένων και μπορεί να ποικίλλουν από θέματα. Τα προβλήματα Ταξινόμηση Κειμένου περιλαμβάνουν, μεταξύ άλλων ταξινόμηση συναισθημάτων, ταξινόμηση ειδήσεων, ταξινόμηση πρόθεσης παραπομπών. Τα σύνολα δεδομένων αναφοράς για την αξιολόγηση των δυνατοτήτων Ταξινόμηση Κείμενου περιλαμβάνουν, μεταξύ άλλων τα GLUE και AGNews. Τα τελευταία χρόνια, μοντέλα Βαθιάς Μάθησης, όπως το XLNet και το RoBERTa έχουν επιτύχει μερικά από τα καλύτερα αποτελέσματα απόδοσης για προβλήματα Ταξινόμηση Κειμένου [69].

4.3 Μοντέλα Εφαρμογής

4.3.1 Μοντέλα για την Απάντηση Ερωτήσεων

1. MobileBERT

Στην παρούσα εφαρμογή, αρχικά γίνεται δοκιμή διαφόρων μοντέλων στην εργασία της Απάντησης Ερωτήσεων ή αλλιώς "Question and Answering". Στην συγκεκριμένη εφαρμογή, η όλη ανάπτυξη στηρίζεται σε εφαρμογή που δημοσιεύτηκε το 2019 από συγγραφείς της πλατφόρμας του TensorFlow, η οποία, όπως αναφέρθηκε, υποστηρίζει εργαλεία και τεχνικές προκειμένου μοντέλα Μηχανικής Μάθησης να εκτελούνται τοπικά στη συσκευή και να είναι μέρος αυτής. Στην εφαρμογή φαίνεται η λειτουργία ενός MobileBERT μοντέλου πάνω στην εργασία Απάντησης Ερώτησης [70].

Σχετικά με την λειτουργία της εφαρμογής, αρχικά δίνεται μια λίστα από δεδομένα κειμενα από τα οποία ο χρήστης μπορεί να επιλέξει ένα. Στη συνέχεια, ο χρήστης μπορεί να επιλέξει είτε κάποια από τις προτεινόμενες ερωτήσεις είτε να πληκτρολογήσει μια δική του ερώτηση σχετικά με το κείμενο. Μετά την εκτέλεση του αλγορίθμου η απάντηση υπογραμμίζεται πάνω στο κείμενο σε πολύ μικρό χρόνο. Το MobileBERT μοντέλο αυτό έχει εκπαιδευτεί πάνω στο σύνολο δεδομένων SQuAD 1.1. Το μοντέλο παίρνει ένα απόσπασμα του κειμένου και μια ερώτηση ως είσοδο και, στη συνέχεια, επιστρέφει ένα τμήμα του αποσπάσματος που πιθανότατα απαντά στην ερώτηση που έχει δοθεί. Το συγκεκριμένο μοντέλο απαιτεί ημισύνθετη προεπεξεργασία, συμπεριλαμβανομένων των βημάτων δημιουργίας διακριτικών και μεταεπεξεργασίας.

Πιο αναλυτικά αυτό το MobileBERT μοντέλο τρέχει τέσσερις φορές πιο γρήγορα ενώ ταυτόχρονα έχει τέσσερις φορές πιο μικρό μέγεθος από το BERT. Στην εφαρμογή το μοντέλο αυτό εισάγεται με την μορφή ενός tflite αρχείου το οποίο περιέχει στα μεταδεδομένα του στοιχεία για την είσοδο, την έξοδο καθώς και το αρχείο λεξικού το οποίο έχει δημιουργηθεί κατά την εκπαίδευση του. Πιο συγκεκριμένα, όπως αναφέρθηκε προηγουμένως ένα tflite μοντέλο είναι σαν ένα συμπιεσμένο αρχείο (zip file). Εκτελώντας απλά μια εντολή αποσυμπίεσης αρχείου ή κάνοντας εξαγωγή των metadata αρχείων μέσω της Python βλέπουμε ότι στο tflite αρχείο περιέχεται ένα αρχείο vocab.txt το οποίο και είναι το λεξικό του μοντέλου.

Στη συνέχεια, το μοντέλο αυτό δέχεται ως είσοδο πίνακες (arrays) τύπου int32[1,384], δηλαδή πίνακες από ακέραιους, μη προσημασμένους αριθμούς που αναπαρίστανται με 32bits, των 384 θέσεων. Πιο συγκεκριμένα, παρατηρούμε ότι ως είσοδο στο μοντέλο δίνονται τρεις πίνακες οι οποίοι είναι οι εξής:

1. **Ένας πίνακας input_ids**, στον οποίο εμπεριέχονται τα ids των tokens όπως παράγονται από τον tokenizer.

2. **Ένας πίνακας input_mask**, στον οποίο εμπεριέχονται τιμές 1 ή 0 σε κάθε θέση που αντιστοιχούν σε πραγματικά ή padding tokens αντίστοιχα.

3. **Ένας πίνακας segment_ids**, στον οποίο εμπεριέχονται τιμές 0 ή 1 που αντιστοιχούν σε θέσεις της πρώτης πρότασης (ερώτησης) και της δεύτερης (κείμενο) αντίστοιχα.

Σχετικά με την έξοδο του μοντέλου είναι πίνακες τύπου float32[1,384], δηλαδή πίνακες, των 384 θέσεων, που αποτελούνται από μη προσημασμένους αριθμούς κινητής υποδιαστολής και απλής ακρίβειας που αναπαρίστανται με 32bits. Οι πίνακες εισόδου και εξόδου είναι οι εξής:

1. **Ένας πίνακας end_logits**, στον οποίο περιέχονται τα logits της ακολουθίας τα οποία υποδεικνύουν την τελική θέση της απάντησης.
2. **Ένας πίνακας start_logits**, στον οποίο περιέχονται τα logits της ακολουθίας τα οποία υποδεικνύουν την αρχική θέση της απάντησης [70].

Τέλος, το παραπάνω προκειμένου να εισαχθεί στο κινητό τηλέφωνο έχει υποστεί την διαδικασία του κβαντισμού προκειμένου να περιοριστεί το αρχικό μέγεθός του. Για αυτόν τον λόγο το μοντέλο έχει υποστεί κβαντισμό Post-training dynamic range quantization. Έτσι το αρχείο που προκύπτει έχει τελικό μέγεθος ίσο με 96MB.

2. DistilBERT

Με βάση την παραπάνω εφαρμογή λοιπόν προστέθηκε στην εφαρμογή άλλο ένα μοντέλο για την εργασία Απάντησης Ερωτήσεων, το οποίο είναι προεκπαιδευμένο και προέρχεται από την κοινότητα Hugging Face [71]. Συγκεκριμένα, το μοντέλο είναι ένα DistilBERT μοντέλο για το οποίο ισχύει:

1. Είναι uncased, όρος που αναφέρεται στο ότι το μοντέλο βλέπει τους κεφαλαίους και τους πεζούς χαρακτήρες ως ίδιους.
2. Είναι η BASE εκδοχή του, όσον αφορά το μέγεθος του.
3. Είναι εκπαιδευμένο στο SQuAD v1.1 σύνολο δεδομένων.
4. Επιτυγχάνει ακρίβεια F1 87,1 στο dev σύνολο δεδομένων, ενώ ταυτόχρονα η έκδοση *BERT_BASE* cased φτάνει σε βαθμολογία F1 88,7. Σημειώνεται ότι, στην cased εκδοχή, το μοντέλο βλέπει τους κεφαλαίους και τους πεζούς χαρακτήρες ως διαφορετικούς.

Εύκολα συμπεραίνουμε ότι το ότι το μοντέλο βλέπει τα κεφαλαία και τα πεζά γράμματα ως διαφορετικά είναι κάτι που ένα πλεονέκτημα αφού με αυτό τον τρόπο αν για παράδειγμα εμπεριέχεται στο λεξικό του μοντέλου το token “this" και στο μοντέλο δοθεί η λέξη "THIS" μέσα στην είσοδο τότε το μοντέλο θα κάνει την αντιστοίχιση στο id του token “this" που έχει ξανά συναντήσει κατά την εκπαίδευση του και το token δεν θα καταγραφεί ως άγνωστο σε αυτό.

Στην περίπτωση αυτού του μοντέλου η είσοδος είναι ένας πίνακας τύπου int32[1,384] ενώ η έξοδος είναι δυο πίνακες τύπου float32[1,384]. Πιο συγκεκριμένα, παρατηρούμε ότι η είσοδος και οι έξοδος του μοντέλο είναι οι εξής:

1. **Ένας πίνακας εισόδου**, ο οποίος αντιστοιχεί στον πίνακα input_ids του MobileBERT.
2. **Ένας πίνακας εξόδου**, ο οποίος αντιστοιχεί στο πίνακα "end_logits" του MobileBERT.

3. **Ένας πίνακας εξόδου**, ο οποίος αντιστοιχεί στο πίνακα "start_logits" του MobileBERT.

Έτσι λοιπόν με αυτόν τον τρόπο δίνεται η δυνατότητα σύγκρισης της εκτέλεσης των δυο μοντέλων σε άγνωστες για αυτά εισόδους. Στο Κεφάλαιο 6 τα μοντέλα αυτά θα συγκριθούν ανάλογα με συγκεκριμένες μετρικές. Περισσότερες λεπτομέρειες θα δοθούν στο κεφάλαιο της ανάπτυξης της εφαρμογής και των αποτελεσμάτων.

4.3.2 Μοντέλα για την Ταξινόμηση Κειμένου

1. TensorFlow Model

Στη συνέχεια, στην παρούσα εφαρμογή, γίνεται δοκιμή διαφόρων μοντέλων στην εργασία της Ταξινόμηση Κειμένου ή αλλιώς "Text Classification". Στην συγκεκριμένη εφαρμογή, οι ετικέτες της ταξινόμησης είναι το "Positive" και "Negative", έτσι δηλαδή το μοντέλο παράγει ένα ποσοστό για την "Θετικότητα" ή την "Αρνητικότητα" του δοθέντος κειμένου όσον αφορά την χροιά του. Ένα από τα μοντέλα που έχουν εισαχθεί για αυτή την εργασία μάλιστα έχει και ετικέτα "Neutral" που αντιστοιχεί στο ποσοστό "Ουδετερότητας" ενός κειμένου. Για καλύτερη κατανόηση αυτού, μια θετική κρητική προφανώς έχει κατά μεγαλύτερο ποσοστό θετική χροιά, όπως αντίστοιχα μια αρνητική έχει κατά μεγαλύτερο ποσοστό αρνητική χροιά, σχετικά με την ουδέτερη χροιά μπορούμε να πούμε ότι ένα κείμενο, όπως μια περιγραφή χωρίς σχόλια, έχει κατά μεγαλύτερο ποσοστό ουδέτερη χροιά. Και σε αυτή την περίπτωση η όλη ανάπτυξη στηρίζεται σε εφαρμογή που δημοσιεύτηκε επίσης το 2019 από τους συγγραφείς της πλατφόρμας του TensorFlow [72]. Στην εφαρμογή φαίνεται η λειτουργία ενός μοντέλου πάνω στην εργασία Ταξινόμησης Κειμένου.

Το μοντέλο που δίνεται στην εφαρμογή αυτή δεν είναι κάποιας γνωστής αρχιτεκτονικής αλλά είναι ένα μοντέλο πέντε επιπέδων. Το μοντέλο αυτό δέχεται ως είσοδο ένα πίνακα τύπου int32[1,256] ενώ ως έξοδος προκύπτει ένας πίνακα τύπου float32[1,2]. Οι πληροφορίες των εισόδων και εξόδων φαίνονται παρακάτω:

1. **Ένας πίνακας εισόδου**, που περιέχει το σύνολο των ids όλων των tokens του κειμένου.

2. **Ένας πίνακας εξόδου**, που περιέχει δύο δείκτες, έναν για κάθε κλάση που μπορεί να ταξινομήσει το μοντέλο (positive και negative).

Τέλος, το μοντέλο αυτό δεν είναι κβαντισμένο αφού δεδομένου του ότι έχει μόνο τέσσερα επίπεδα δεν χρειάζεται, αφού το μέγεθος του είναι ήδη πολύ μικρό. Στην περίπτωση των άλλων δυο μοντέλων η συνάρτηση αυτή εκτελείται στο τελικό επίπεδο του μοντέλου για αυτό και το αποτέλεσμα είναι κανονικοποιημένο

3. ELECTRA

Τέλος, το επόμενο μοντέλο που προστίθεται είναι ένα προεκπαιδευμένο μοντέλο ELECTRA το οποίο επίσης προέρχεται από την κοινότητα Hugging Face Το μοντέλο αυτό βρέθηκε

έτοιμο σε tlite μορφή και έχει μέγεθος ίσο με 13.3 MB [73]. Συγκεκριμένα, εισάγεται το αρχείο με όνομα με το μοντέλο κβαντισμένο με Post-training dynamic ("imdb_small_8bits.tflite). Η είσοδος που δέχεται το συγκεκριμένο μοντέλο είναι ένας πίνακας τύπου int32[1,40] ενώ η έξοδος που προκύπτει είναι ένας πίνακας τύπου float32[1,2]. Οι πληροφορίες των εισόδων και εξόδων φαίνεται παρακάτω:

1. **Ένας πίνακας εισόδου**, που περιέχει το σύνολο των ids όλων των tokens του κειμένου.

2. **Ένας πίνακας εξόδου**, που περιέχει δύο δείκτες, έναν για κάθε κλάση που μπορεί να ταξινομήσει το μοντέλο (positive και negative).

2. RoBERTa

Με βάση αυτή την εφαρμογή λοιπόν και σε αυτήν την περίπτωση προστίθενται στην κινητή συσκευή και άλλα μοντέλα τα οποία εκτελούν την αντίστοιχη εργασία. Το επόμενο μοντέλο που προστίθεται είναι ένα προεκπαιδευμένο μοντέλο RoBERTa το οποίο επίσης προέρχεται από την κοινότητα Hugging Face [74]. Συγκεκριμένα, το μοντέλο είναι ένα RoBERTa μοντέλο για το οποίο ισχύει ότι:

1. Είναι η BASE εκδοχή του, όσον αφορά το μέγεθος του.

2. Είναι εκπαιδευμένο εκπαιδευμένο σε περίπου 58 εκατομμύρια tweets, ενώ βελτιστοποιήθηκε (finetuned) για ανάλυση συναισθήματος στο σημείο αναφοράς TweetEval.

Όσον αφορά τις εισόδους και εξόδους του μοντέλου είναι ίδιες με του προηγούμενο μοντέλου με μια διαφοροποίηση στην έξοδο. Συγκεκριμένα, η είσοδος είναι ένας πίνακας τύπου int32[1,256] ενώ η έξοδος που προκύπτει είναι ένας πίνακας τύπου float32[1,3]. Οι πληροφορίες των εισόδων και εξόδων φαίνονται παρακάτω:

1. **Ένας πίνακας εισόδου**, που περιέχει το σύνολο των ids όλων των tokens του κειμένου.

2. **Ένας πίνακας εξόδου**, που περιέχει τρεις δείκτες, έναν για κάθε κλάση που μπορεί να ταξινομήσει το μοντέλο (positive, negative και netrual).

Στην περίπτωση του RoBERTa παρατηρούμε ότι τα αποτελέσματα της εξόδου δεν έχουν λογική αφού αν τα προσθέσω δεν έχουν άθροισμα ίσο με 1 και δεν μπορούν να αναπαρασταθούν με ποσοστά. Για αυτό το λόγο η συνάρτηση softmax εκτελείται απευθείας στην έξοδο του μοντέλου προκειμένου να κανονικοποιηθεί. Πιο αναλυτικά, η συνάρτηση softmax εκχωρεί δεκαδικές πιθανότητες σε κάθε τάξη σε ένα πρόβλημα πολλαπλών κλάσεων. Αυτές οι δεκαδικές πιθανότητες πρέπει να αθροίζονται και να ισούται με το 1,0. Αυτή η πρακτική χρησιμοποιείται ευρέως στα Νευρωνικά Δίκτυα αφού ο περιορισμός αυτός βοηθά την εκπαίδευση να συγκλίνει πιο γρήγορα από ό,τι θα συνέβαινε διαφορετικά [75].

4.4 Προεπεξεργασία εισόδου

Στην παρούσα εφαρμογή χρησιμοποιούνται διάφορα είδη τμηματοποίησης. Στην εργασία Απάντησης Ερώτησης ο BERT tokenizer που χρησιμοποιείται στην περίπτωση του MobileBERT χρησιμοποιείται και στον DistilBERT, αφού το μοντέλο αυτό αποτελεί μαθητή του BERT. Στη συνέχεια, στην εφαρμογή του Tensorflow για το Text Classification εφαρμόζεται ένας απλός tokenizer. Στην περίπτωση, του ELECTRA εφαρμόζεται και πάλι ο BERT tokenizer, αφού η τμηματοποίηση που εφαρμόζεται στο ELECTRA είναι πανομοιότυπη με αυτή που εφαρμόζεται στο BERT. Τέλος, στο RoBERTa εφαρμόζεται ο ROBERTA tokenizer ο οποίος έχει κάποιες διαφορές από τους υπόλοιπους. Χάριν κατανόησης όλοι οι tokenizers αναφέρονται παρακάτω.

4.4.1 BERT Tokenizer

Στην περίπτωση του BERT χρησιμοποιήθηκε η έξυπνη ιδέα της χρήσης της έννοιας του WordPiece tokenizer που παρουσιάστηκε στο Υποκεφάλαιο 2.3.3 [7]. Με βάση τα παραπάνω ο WordPiece αλγόριθμος λειτουργεί χωρίζοντας τις λέξεις είτε σε πλήρεις μορφές (π.χ., μια λέξη γίνεται ένα διακριτικό) είτε σε κομμάτια λέξεων, όπου μια λέξη μπορεί να χωριστεί σε πολλαπλά tokens. Ένα παράδειγμα όπου αυτό μπορεί να είναι χρήσιμο, είναι στην περίπτωση που έχουμε πολλαπλές μορφές λέξεων όπως φαίνεται στην Εικόνα 4.8.

Word	Token(s)
surf	['surf']
surfing	['surf', '##ing']
surfboarding	['surf', '##board', '##ing']
surfboard	['surf', '##board']
snowboard	['snow', '##board']
snowboarding	['snow', '##board', '##ing']
snow	['snow']
snowing	['snow', '##ing']

Εικόνα 4.8: *Αποτέλεσμα του WordPiece tokenizer σε διάφορα παράγωγα λέξεων [13]*

Διαχωρίζοντας τις λέξεις σε κομμάτια λέξεων, έχουμε ήδη προσδιορίσει ότι οι λέξεις "surfboard" και "snowboard" μοιράζονται νόημα μέσω του λεκτικού κομματιού "##board". Αυτό γίνεται χωρίς καν να κωδικοποιήσουμε τα διακριτικά μας ή να τα επεξεργαστούμε με οποιονδήποτε τρόπο μέσω του BERT. Η χρήση κομματιών λέξεων επιτρέπει στον BERT να αναγνωρίζει εύκολα σχετικές λέξεις, καθώς συνήθως μοιράζονται μερικά από τα ίδια διακριτικά εισόδου, τα οποία στη συνέχεια τροφοδοτούνται στα πρώτα στρώματα του BERT. Επομένως το λεξιλόγιο που προκύπτει από την εκπαίδευση ενός BERT μοντέλου περιέχει

τόσο λέξεις όσο και κομμάτια λέξεων [13].

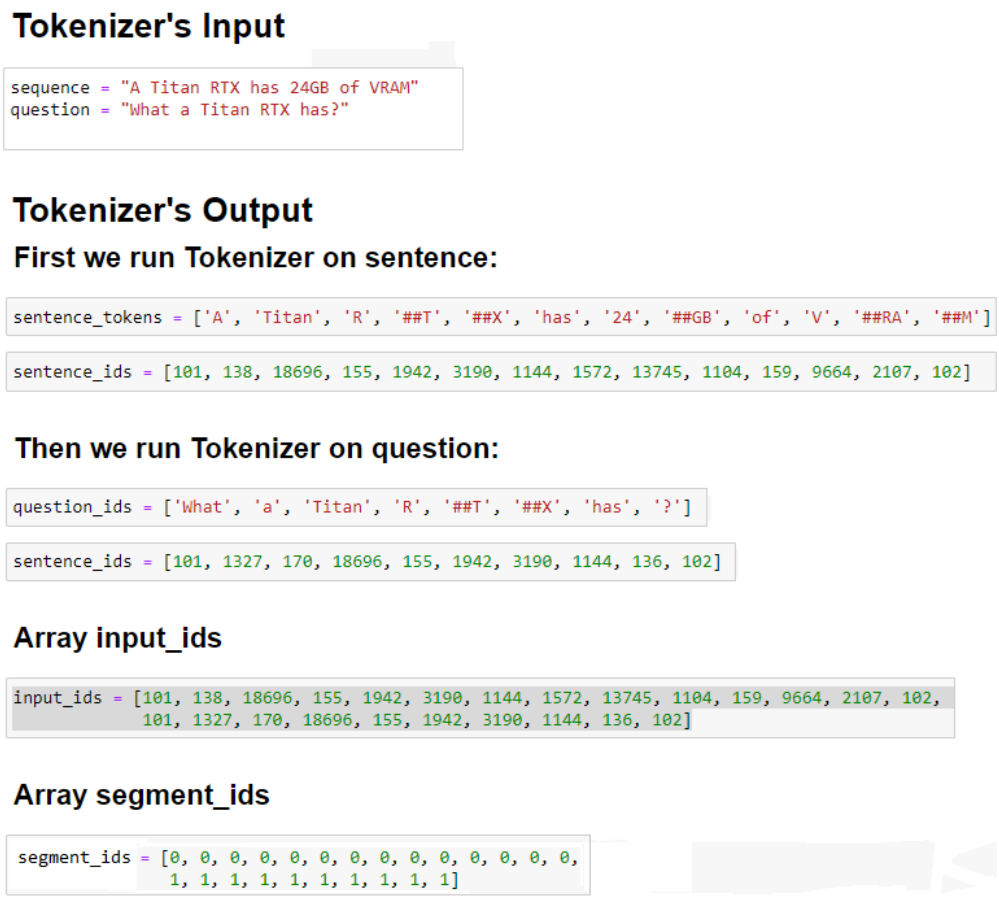
Στον BERT tokenizer πέρα από όλα τα tokens που προκύπτουν από την διαδικασία προστίθενται στα συνολικά tokens άλλα δύο ειδικά, τα [CLS] και [SEP]. Πιο συγκεκριμένα, το μοντέλο BERT έχει σχεδιαστεί με τέτοιο τρόπο ώστε η πρόταση να ξεκινά με το διακριτικό [CLS] και να τελειώνει με το διακριτικό [SEP]. Εάν εργαζόμαστε σε εργασίες όπως απάντηση ερωτήσεων ή μετάφραση γλώσσας, τότε πρέπει να χρησιμοποιήσουμε το διακριτικό [SEP] ανάμεσα στις δύο προτάσεις για τις διαχωρίσουμε. Στη περίπτωση της απάντησης ερωτήσεων, η είσοδος κατά την τμηματοποίηση μετατρέπεται σε ένα σύνολο tokens με τα tokens της ερώτησης και του κειμένου να χωρίζονται μεταξύ τους με ένα token τύπου [SEP]. Αυτά τα παραγόμενα tokens λοιπόν αντιστοιχίζονται σε ids μέσω του αρχείου λεξιλογίου με τη διαδικασία που έχει αναφερθεί παραπάνω. Έτσι λοιπόν προκύπτει το πρώτο κομμάτι της εξόδου της τμηματοποίησης που είναι τα input_ids. Επιπλέον προκύπτει το δεύτερο, τα segment_ids τα οποία έχουν τις τιμές 0 και 1 ανάλογα με το αν αντιστοιχούν στην ερώτηση ή στο κείμενο αντίστοιχα. Από αυτό είναι προφανές ότι αν η είσοδος είναι για άλλη εργασία όπως Text Classification τότε τα segment_ids δεν χρειάζονται αφού η είσοδος είναι ένα μόνο κείμενο και αυτά θα είναι όλα 0. Τέλος, προκύπτει το τρίτο κομμάτι της εξόδου το attention_mask ή αλλιώς τα input_mask. Για να κατανοήσουμε το input_mask πρέπει να επεξεργαστούμε δεδομένα σε παρτίδες (batches). Σε μια παρτίδα, μπορεί να έχουμε διαφορετικά μήκη εισόδων. Το μοντέλο απαιτούσε πάντα δεδομένα εισαγωγής σε ορθογώνια μορφή όταν εισάγουμε δεδομένα σε παρτίδες.

Έτσι λοιπόν προκειμένου τα δεδομένα να έχουν ορθογώνια μορφή κατά τη διάρκεια της διαδικασίας γίνεται συμπλήρωση τιμών στη δεξιά πλευρά των διακριτικών σε περιπτώσεις μικρότερων προτάσεων έτσι ώστε να διασφαλιστεί ότι το μοντέλο δεν θα εξετάσει αυτές τις τιμές με επένδυση χρησιμοποιείται το input_mask. Έστω ότι έχουμε ένα πραγματικό πρόβλημα δεδομένων για εργασία απάντησης μιας ερώτησης. Έτσι λοιπόν φτιάχνονται δύο λίστες, η πρώτη λίστα περιέχει όλα τα tokens της ερώτησης και η δεύτερη λίστα περιέχει όλα τα tokens του κειμένου. Σε περίπτωση εισόδου μικρότερου μεγέθους από το απαιτούμενο για το εκάστοτε μοντέλο στα ids της εξόδου του tokenizer προστίθενται ids ως padding προκειμένου να συμπληρωθεί το απαιτούμενο μέγεθος. Σε αυτό το σημείο φαίνεται η σημασία του input_mask, αφού το μοντέλο θα εστιάζει μόνο στα ids που αντιστοιχούν σε input_mask ίση με 1 [7].

Προκειμένου να γίνει καλύτερη κατανόηση του BERT Tokenizer στην Εικόνα 4.9 δίνεται ένα παράδειγμα εισόδου και η παραγόμενη από αυτή έξοδος. Συγκεκριμένα, στην εικόνα αυτή φαίνεται το πως λειτουργεί ο Tokenizer στην περίπτωση Ερώτησης Απάντησης όπου γίνεται τόσο η τμηματοποίηση του κειμένου και της ερώτησης αλλά και η σύνδεση τους. Σε αυτό το παράδειγμα προκειμένου να απλοποιηθεί η διαδικασία δεν θα ληφθεί υπόψιν η διαδικασία του padding. Όπως καταλαβαίνουμε αφού δεν αναφερόμαστε στην διαδικασία του padding δεν θα αναφερθούμε στην τιμή του πίνακα input_mask.

4.4.2 RoBERTa Tokenizer

Στην περίπτωση του μοντέλου RoBERTa εφαρμόζονται διαφορετικές τεχνικές τμηματοποίησης. Πιο συγκεκριμένα, αυτός ο tokenizer έχει εκπαιδευτεί να χειρίζεται τα κενά σαν

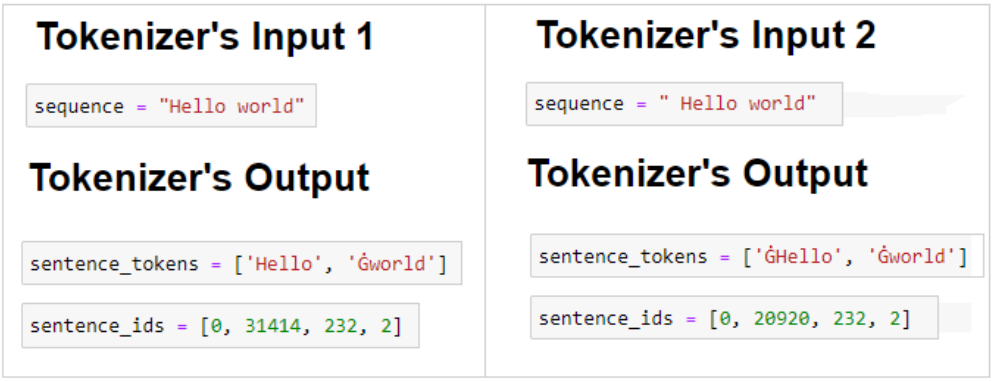


Εικόνα 4.9: *Αποτέλεσμα του BERT tokenizer*

μέρη των διακριτικών, επομένως μια λέξη θα κωδικοποιείται διαφορετικά ανάλογα με το αν βρίσκεται στην αρχή της πρότασης (χωρίς διάστημα) ή όχι. Συγκεκριμένα, κάθε κενό αντιστοιχίζεται στο παραγόμενο token με τον χαρακτήρα "Ġ". Επιπλέον, όπως και στην περίπτωση του BERT tokenizer έτσι και στην περίπτωση του RoBERTa tokenizer εκτός από όλα τα tokens που προκύπτουν από την διαδικασία τμηματοποίησης στο δοσμένο κείμενο έχουμε και τα ειδικά tokens τα οποία σηματοδοτούν την αρχή και την λήξη του κειμένου. Αυτά τα tokens είναι τα "$< s >$" και "$< /s >$" για τα tokens έναρξης και την λήξη αντίστοιχα. Στα tokens αυτά αποδίδονται τα ids 0 και 2 αντίστοιχα, κάτι που έχει προσυμφωνηθεί λόγω του αρχείου του λεξιλογίου. Στην περίπτωση που το μέγεθος του κειμένου εισόδου δεν αντιστοιχεί στο μήκος εισόδου που δέχεται το μοντέλο τότε ο tokenizer προσθέτει ειδικά tokens "pad" που είναι τα padding tokens.

Για περισσότερη κατανόηση στην Εικόνα 4.10 φαίνεται η έξοδος του RoBERTa Tokenizer για δύο διαφορετικές εισόδους. Σε αυτό το παράδειγμα προκειμένου να απλοποιηθεί η διαδικασία δεν θα ληφθεί υπόψιν η διαδικασία του padding.

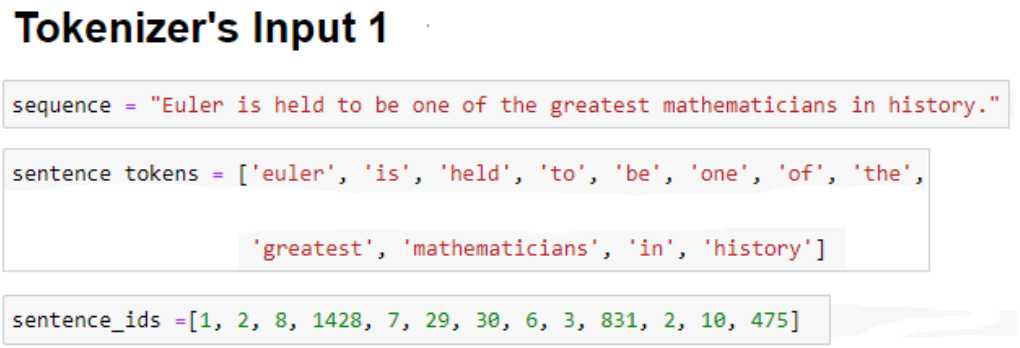
Αρχικά παρατηρούμε ότι στα παραγόμενα ids έχουν προστεθεί τα ειδικά ids αρχής και λήξης. Στη συνέχεια, παρατηρούμε ότι η έξοδος του tokenizer είναι ίδια και στις δύο περιπτώσεις όσον αφορά την λέξη World, αλλά όχι την λέξη Hello. Αυτό συμβαίνει επειδή στην πρώτη περίπτωση η λέξη Hello υπολογίζεται μόνη της, ενώ στη δεύτερη περίπτωση εφόσον υπάρχει κενό συμπεριλαμβάνεται στο token για αυτό και του αποδίδεται διαφορετικό id.



Εικόνα 4.10: *Αποτέλεσμα του RoBERTa tokenizer*

4.4.3 TensorFlow Tokenizer

Στο μοντέλο του TensorFlow εφαρμόζεται μια απλή διαδικασία τμηματοποίησης. Συγκεκριμένα, στο μοντέλο αυτό η είσοδος προκύπτει διαχωρισμό των λέξεων του κειμένου σε tokens, όπου κάθελέξη αποτελεί ένα token, ενώ αγνοεί τα σημεία στίξης. Στη συνέχεια, εισάγονται τα ειδικά tokens όπως το "START" που σημαίνει την αρχή, το "UNKNOWN" για τα άγνωστα tokens και τέλος το "PAD" που είναι τα padding tokens, δηλαδή τα tokens που προστίθενται σε περίπτωση που η είσοδος είναι μικρότερη από την είσοδο που χρειάζεται το μοντέλο. Στην Εικόνα 4.11 φαίνεται ένα παράδειγμα εξόδου του TensorFlow tokenizer. Σε αυτό το παράδειγμα προκειμένου να απλοποιηθεί η διαδικασία δεν θα ληφθεί υπόψιν η διαδικασία του padding.



Εικόνα 4.11: *Αποτέλεσμα του TensorFlow tokenizer*

4.5 Μετατροπή Μοντέλων

Μετατροπή του DistilBERT

Προκειμένου το DistilBERT μοντέλο που αναλύσαμε να μετατραπεί σε tflite μορφή χρησιμοποιείται ο μετατροπέας TensorFlow Lite. Στην Εικόνα Α΄.1 του Παραρτήματος Α, βλέπουμε τις εντολές που εκτελέστηκαν στην Python προκειμένου να γίνει η μετατροπή του μοντέλου στην μορφή που μας εξυπηρετεί στην συγκεκριμένη περίπτωση. Τα βήματα είναι τα εξής:

- **Βήμα 1**: το μοντέλο εξάγεται από την βιβλιοθήκη των transformers για την εργασία που μας ενδιαφέρει.
- **Βήμα 2** :γίνεται αλλαγή του μεγέθους της εισόδου έτσι ώστε να είναι ίδια με του BERT ,το οποίο υπάρχει ήδη, χάριν ομοιομορφίας ([1,384]).
- **Βήμα 3**: γίνεται βελτιστοποίηση του μοντέλου μέσω των λειτουργιών ops και γίνεται Post-training dynamic range κβαντοποίηση δυναμικού εύρους.
- **Βήμα 4**: γίνεται η μετατροπή του μοντέλου και η αποθήκευση του σε αρχείο μορφής tflite.
- **Βήμα 5**: γίνεται εισαγωγή του λεξιλογίου στα metadata του μοντέλου.

Το τελικό μέγεθος του tflite αρχείου που προκύπτει μετά την εισαγωγή του λεξιλογίου και τη κβαντοποίηση είναι 80.145 KB, ενώ στην αρχή είχε μέγεθος 314.457 KB, κάτι που θα χαρακτηρίζαμε ως μεγάλη σε σχέση με το αρχικό μείωση, πολύτιμη για τον σκοπό μας.

Μετατροπή του RoBERTa

Και σε αυτήν την περίπτωση προκειμένου το RoBERTa μοντέλο που αναλύσαμε να μετατραπεί σε tflite μορφή χρησιμοποιείται ο μετατροπέας TensorFlow Lite. Η διαδικασία μετατροπής είναι αντίστοιχη με την διαδικασία που εφαρμόστηκε προηγουμένως στο DistilBERT με τι ανάλογες διαφορές όπως βλέπουμε στην Εικόνα Α΄.2 του Παραρτήματος Α. Για την μετατροπή του μοντέλου αυτού σε TensorFlow lite μορφή εκτελέστηκαν οι εξής εντολές, αντίστοιχα με πριν:

- **Βήμα 1**: το μοντέλο εξάγεται από την βιβλιοθήκη των transformers για την εργασία που μας ενδιαφέρει.
- **Βήμα 2**: γίνεται αλλαγή του μεγέθους της εισόδου έτσι ώστε να είναι ίδια με το μοντέλο του TensorFlow ,το οποίο υπάρχει ήδη, χάριν ομοιομορφίας ([1,256]).
- **Βήμα 3**: γίνεται βελτιστοποίηση του μοντέλου μέσω των λειτουργιών ops και γίνεται Post-training dynamic range κβαντοποίηση δυναμικού εύρους.
- **Βήμα 4**: γίνεται η μετατροπή του μοντέλου και η αποθήκευση του σε αρχείο μορφής tflite.
- **Βήμα 5**: γίνεται εισαγωγή των αρχείων λεξιλογίου στα metadata του μοντέλου.
- **Βήμα 6**: γίνεται εισαγωγή των αρχείων των ετικετών στα metadata του μοντέλου.

Κατ' αυτό τον τρόπο μετά το κβαντισμό το tflite αρχείο που παράγεται προκύπτει να έχει μέγεθος ίσο με 147 MB από 586.114 KB που έχει προηγουμένως. Για καλύτερη κατανόηση όλων των παραπάνω παρατίθεται ο Πίνακας 4.1 με τις πληροφορίες όλων των μοντέλων που αναλύθηκαν παραπάνω ενώ πληροφορίες για την ακρίβεια των εισόδων, των βαρών, των ενεργοποιήσεων και εξόδων δίνονται στον Πίνακα 3.1.

Μοντέλο	Εργασία	Tokenizer	Κβαντισμός	Μέγεθος
MobileBERT	Question & Answering	BERT tokenizer	Post-training dynamic range	4x μικρότερο
DistilBERT	Question & Answering	BERT tokenizer	Post-training dynamic range	4x μικρότερο
TensorFlow	Text Classification	TensorFlow tokenizer	Non-quantized	-
RoBERTa	Text Classification	RoBERTa tokenizer	Post-training dynamic range	4x μικρότερο
ELECTRA	Text Classification	BERT tokenizer	Post-training dynamic range	4x μικρότερο

Πίνακας 4.1: *Σύνοψη στοιχείων μοντέλων*

Κεφάλαιο 5

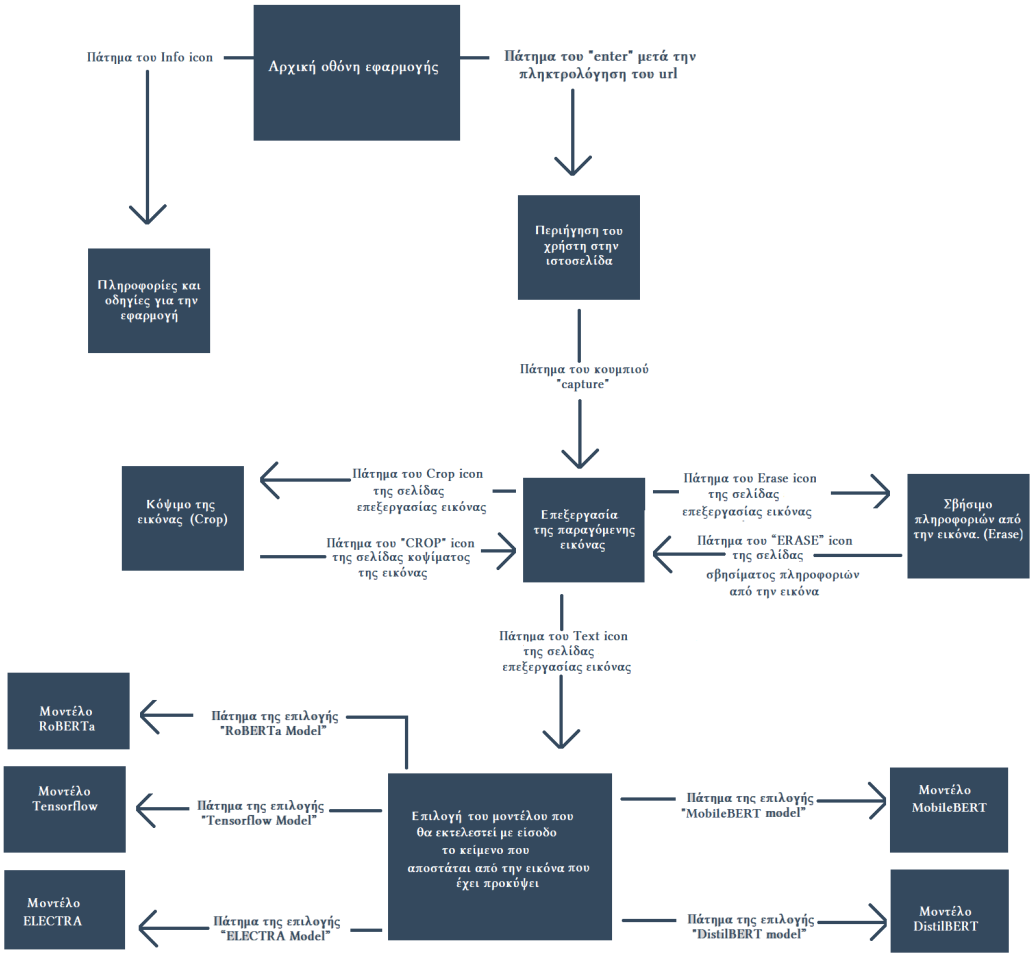
Η εφαρμογή

Στο κεφάλαιο αυτό γίνεται η παρουσίαση της εφαρμογής. Πιο συγκεκριμένα, θα αναλυθούν οι μέθοδοι που χρησιμοποιήθηκαν προκειμένου να ικανοποιηθούν οι ανάγκες και έτσι να οδηγηθούμε στο επιθυμητό αποτέλεσμα. Εκτός αυτού, δίνονται στιγμιότυπα οθόνης έτσι ώστε να παρουσιαστεί η διεπαφή του χρήστη (user interace), το σύνολο δηλαδή των συστατικών ενός συστήματος το οποίο επιτρέπει αμφίδρομη επικοινωνία μεταξύ συστήματος και χρήστη.

5.1 Ροή Εργασιών

Η εφαρμογή αυτή δημιουργήθηκε ουσιαστικά ως ένα εργαλείο με σκοπό την μελέτη των αποδόσεων αλγορίθμων Βαθιάς Μάθησης με εφαρμογή σε εργασίες NLP που εκτελούνται σε κινητές συσκευές. Οι εργασίες αυτές προαναφέρθηκαν σε προηγούμενα κεφάλαια και είναι η απάντηση ερωτήσεων και η ταξινόμηση κειμένου. Βασικός στόχος της εφαρμογής είναι, κείμενα από όλο το διαδίκτυο να μπορούν να δοθούν ως είσοδοι στα μοντέλα που μελετώνται. Κατά την ανάπτυξη μιας εφαρμογής τα πρώτα βασικά βήματα για την σωστή δημιουργία της είναι αρχικά η κατανόηση των αναγκών που επιθυμούμε να καλύψουμε μέσω της εφαρμογής αυτής και στη συνέχεια ο σωστός σχεδιασμός τόσο της διεπαφής του χρήστη όσο και των εσωτερικών ροών της. Στην Εικόνα 5.1 λοιπόν παρατίθεται το διάγραμμα ροής των οθονών της παρούσας εφαρμογή.

Όπως φαίνεται και στο διάγραμμα ροής αρχικά ο χρήστης ανοίγοντας την εφαρμογή βρίσκεται στην αρχική οθόνη, όπου μπορεί να κάνει επικόλληση ένα Url (Uniform Resource Locator) ή να το πληκτρολογήσει εκεί απευθείας. Αμέσως μετά, ο χρήστης βλέπει τη σελίδα να φορτώνεται μπροστά του, όπως σε έναν διακομιστή ιστού (browser). Στη συνέχεια, με την επιλογή "Capture" που φαίνεται μπροστά του ύστερα από την εμφάνιση της διαδικτυακής σελίδας, μπορεί να να γίνει καταγραφή του περιεχομένου της. Σε αυτό το σημείο η εικόνα που έχει παραχθεί φαίνεται στον χρήστη σε μια νέα οθόνη μαζί με κάποιες επιλογές επεξεργασίας. Εκεί, ο χρήστης μπορεί είτε να κόψει την εικόνα, είτε να σβήσει μέρη από την εικόνα που δεν χρειάζεται, είτε να πατήσει την επεξεργασία της εικόνας ξεκινώντας μια ενέργεια, όπου από την εικόνα που έχει δημιουργηθεί τελικά εξάγεται το κείμενο. Στη συνέχεια, ο χρήστης μπορεί να επιλέξει ένα από τα μοντέλα που έχουν εισαχθεί στην εφαρμογή και να δει τα αποτελέσματα του καθενός από αυτά ανάλογα με την είσοδο που έχει εισάγει σε αυτά.



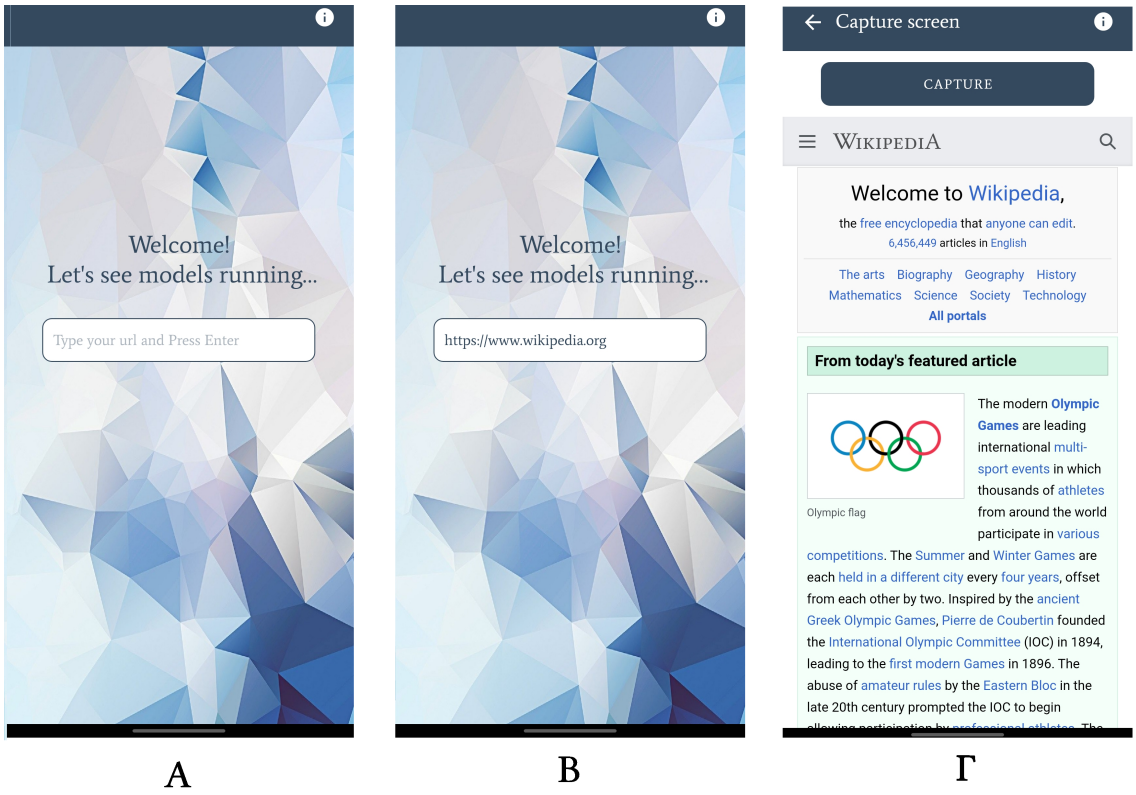
Εικόνα 5.1: *Διάγραμμα ροής της εφαρμογής [14]*

5.2 Ανάπτυξη εφαρμογής

Στη προηγούμενη Ενότητα έγινε μια οπτική αναπαράσταση των λειτουργιών της εφαρμογής μέσω του διαγράμματος ροής. Σε αυτό το κεφάλαιο θα αναλυθούν περαιτέρω οι οθόνες της εφαρμογής και οι λειτουργικότητες τους. Όταν ο χρήστης εισέλθει στην εφαρμογή εκτελεί τέσσερα βασικά βήματα προκειμένου να φτάσει στο σημείο εκτέλεσης του μοντέλου Βαθιάς Μάθησης με την είσοδο που έχει επιλέξει. Τα βήματα αυτά είναι ουσιαστικά η ροή της πληροφορίας μέσω των διαφόρων δραστηριοτήτων (activities) που απαρτίζουν την εφαρμογή. Οι δραστηριότητες αυτές είναι κλάσεις μέσα στις οποίες εκτελείται ο κώδικας που συνδέει την διεπαφή χρήστη με τις εσωτερικές λειτουργίες της εφαρμογής. Πρακτικά μια δραστηριότητα αποτελεί μια σελίδα της εφαρμογής. Οι δραστηριότητες της εφαρμογής είναι οι εξής:

Βήμα 1: MainActivity

Η πρώτη και αρχική δραστηριότητα της εφαρμογής είναι η MainActivity. Στη Εικόνα 5.2 φαίνεται η διεπαφή αυτής. Αυτή είναι ουσιαστικά η πρώτη βασική οθόνη που εμφανίζεται



Εικόνα 5.2: *Σύνολο καταστάσεων της MainActivity*

στον χρήστη κατά την έναρξη της εφαρμογής (Εικόνα 5.2, Α). Αρχικά κατά την έναρξη της εφαρμογής ο χρήστης βλέπει εκτός από ένα κείμενο χαιρετισμού, ένα εικονίδιο μέσω του οποίου μπορεί να δει πληροφορίες για την εφαρμογή και ένα πλαίσιο πληκτρολόγησης. Στη συγκεκριμένη περίπτωση, όπως αναγράφεται και στο βοηθητικό μήνυμα, μπορεί να εισάγει το Url της σελίδας όπου εμπεριέχεται το κείμενο που θέλει να επεξεργαστεί (Εικόνα 5.2, Β). Αφού πληκτρολογήσει το Url και πατώντας το “Enter” η σελίδα φορτώνεται μπροστά του, ενώ πλέον είναι ορατό στην οθόνη ένα κουμπί με τίτλο “Capture”, το οποίο πατάει ο χρήστης εφόσον έχει εντοπίσει το κείμενο που θέλει και είναι έτοιμος να συνεχίσει με την καταγραφή της οθόνης (Screenshot). Με το πάτημα του "Enter" η σελίδα που εμφανίζεται στον χρήστη είναι η δεύτερη σελίδα της ροής της εφαρμογής και είναι ουσιαστικά μια άλλη κατάσταση της MainActivity (Εικόνα 5.2, Γ). Στην περίπτωση που ο χρήστης πατήσει το εικονίδιο για τις πληροφορίες τότε βρίσκεται σε μία σελίδα όπου περιγράφεται η λειτουργία της εφαρμογής και εξηγούνται όλες οι επιλογές αυτής. Τέλος υπάρχει ένα εικονίδιο βέλους προς τα αριστερά με το οποίο ο χρήστης μπορεί να επιστρέψει στην προηγούμενη σελίδα δηλαδή, ουσιαστικά στην προηγούμενη μορφή της MainActivity.

Σχετικά με την ανάπτυξη της λειτουργίας του περιηγητή έχει φτιαχτεί μια κλάση στηριζόμενη στην κλάση WebViewClient η οποία εκτελεί την βασική λειτουργία ενός περιηγητή [76]. Όσον αφορά στην καταγραφή της οθόνης στην περίπτωση αυτή δεν γίνεται όπως στην προκαθορισμένη του κινητού. Ουσιαστικά στην περίπτωση μας δεν γίνεται καταγραφή της οθόνης όπως την βλέπει ο χρήστης, αλλά ουσιαστικά γίνεται καταγραφή όλης της σελίδας, δηλαδή του μέρους που βλέπει ήδη ο χρήστης αλλά και του μέρους που θα δει αν μετακινήσει το δάκτυλο του προς τα πάνω (scrolling) κατά την περιήγηση. Πιο αναλυτικά, η εικόνα

που παράγεται έχει ύψος και μήκος ανάλογα με αυτά της σελίδας και είναι ένα αρχείο εικόνας τύπου Bitmap, όπου ουσιαστικά σε αυτό σχεδιάζεται η διαδικτυακή σελίδα με την βοήθεια της κλάσης Canvas [77].

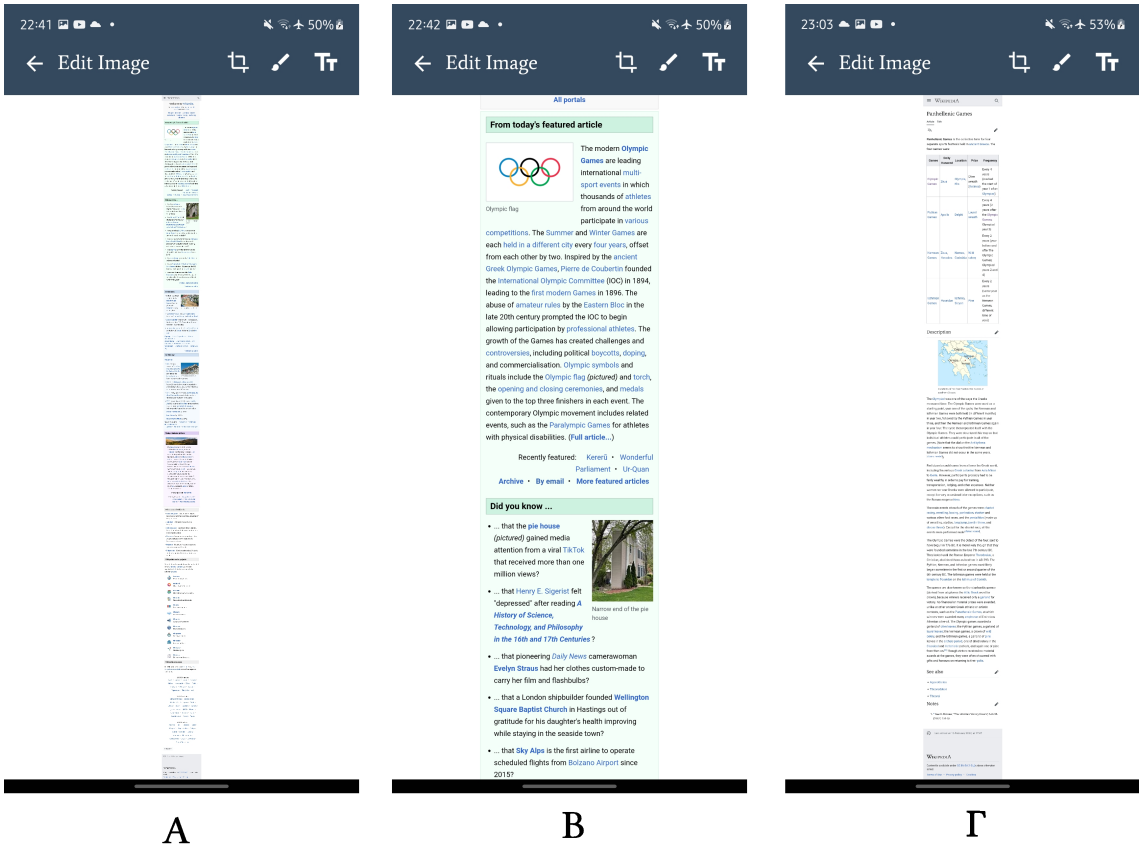
Η μορφή της εικόνας που παράγεται φαίνεται στην Εικόνα 5.3, όπου παρατίθενται διάφορα παραδείγματα κειμένου στις οθόνες Α και Γ. Προφανώς, εύκολα καταλαβαίνουμε ότι η προσπάθεια καταγραφής μιας μεγάλης σελίδας στο διαδίκτυο μπορεί να οδηγήσει σε σφάλμα της εφαρμογής αν δεν ληφθεί υπόψιν κάποιος περιορισμός. Έτσι λοιπόν, προκειμένου να αποφευχθεί αυτό, κατά τη διάρκεια της καταγραφής οθόνης πραγματοποιείται ένας έλεγχος σχετικά με το μέγεθος της εικόνας και μάλιστα ένας έλεγχος για την τιμή του ύψους της οθόνης που είναι και αυτό που μας περιορίζει. Συγκεκριμένα, αν αυτό είναι πάνω από έναν συγκεκριμένο αριθμό τότε η καταγραφή οθόνης πραγματοποιείται από την αρχή μέχρι το μέγιστο αυτό σημείο. Το σημείο αυτό υπολογίστηκε ανάλογα, τόσο με το ποιο είναι το μέγιστο ύψος ενός Bitmap αρχείου, όσο και το μέγιστο ύψος εικόνας που μπορεί να αποθηκευτεί προσωρινά στη RAM της συσκευής, ώστε να μπορεί ο χρήστης να την δει στην οθόνη και στη συνέχεια να την επεξεργαστεί. Τέλος, επίσης σημαντικό κριτήριο για την επιλογή του μέγιστου ύψους της παραγόμενης εικόνας είναι το μέγιστο ύψος που μπορεί να διαχειριστεί η λειτουργία κοψίματος (Cropper) που χρησιμοποιείται για το κόψιμο της εικόνας, στη συνέχεια. Εκτός όλων αυτών, λαμβάνεται υπόψιν και το οπτικό κομμάτι αφού θα πρέπει η εικόνα να είναι ορατή στον χρήστη και να μην είναι απλά μια λεπτή γραμμή, αφού δεδομένου του μεγάλου ύψους η εικόνα θα φαίνεται τόσο μικρή που το περιεχόμενο της δεν θα διακρίνεται σε πρώτη όψη έτσι ώστε να γίνει η επεξεργασία της. Αυτό είναι κάτι που μπορούμε να δούμε και στην Εικόνα 5.3. όπου στα Α και Γ βλέπουμε ότι στην περίπτωση Α που το ύψος της εικόνας είναι πιο μεγάλο το μήκος της γίνεται πιο μικρό όσον αφορά την παρουσίαση της.

Τέλος, σχετικά με το εικονίδιο για τις πληροφορίες, με το πάτημα αυτού ο χρήστης βρίσκεται σε μια νέα οθόνη όπου μπορεί να πληροφορηθεί για τον στόχο της παρούσας εφαρμογής καθώς και για το ποια μοντέλα μπορεί να δοκιμάσει μέσω αυτής. Στη συνέχεια, δίνονται πληροφορίες για την ροή της εφαρμογής έτσι ώστε ο χρήστης να μπορεί να κινηθεί μέσα σε αυτή, καθώς δίνεται και η επεξήγηση όλων των εικονιδίων και λειτουργιών που υπάρχουν μέσα σε αυτή.

Βήμα 2: ImageEditActivity

Η δραστηριότητα ImageEditActivity είναι ουσιαστικά η τρίτη βασική οθόνη που συναντάει ο χρήστη κατά την περιήγηση του στην εφαρμογή και η διεπαφή της φαίνεται στην Εικόνα 5.3. Φτάνοντας ο χρήστης σε αυτή την σελίδα μπορεί πλέον να δει την εικόνα που παράχθηκε από την καταγραφή του περιεχομένου της διαδικτυακής σελίδας. Εκεί αρχικά ο χρήστης μπορεί να κάνει μεγέθυνση στην εικόνα που παρουσιάζεται προκειμένου να τη μελετήσει περαιτέρω και να εντοπίσει το κομμάτι κειμένου που τον ενδιαφέρει. Στην Εικόνα 5.3, Β παρουσιάζεται η λειτουργία της μεγέθυνσης της εικόνας που φαίνεται στην Εικόνας 5.3 ,Α. Έκτος αυτού βλέπουμε ότι υπάρχει μια μπάρα η οποία περιέχει, όπως και πριν, ένα βέλος για την επιστροφή στη προηγούμενη σελίδα της εφαρμογής, δηλαδή στη MainActivity καθώς και τρία εικονίδια στα δεξιά της μπάρας. Τα εικονίδια αυτά όπως φαίνεται στην Εικόνα 5.3

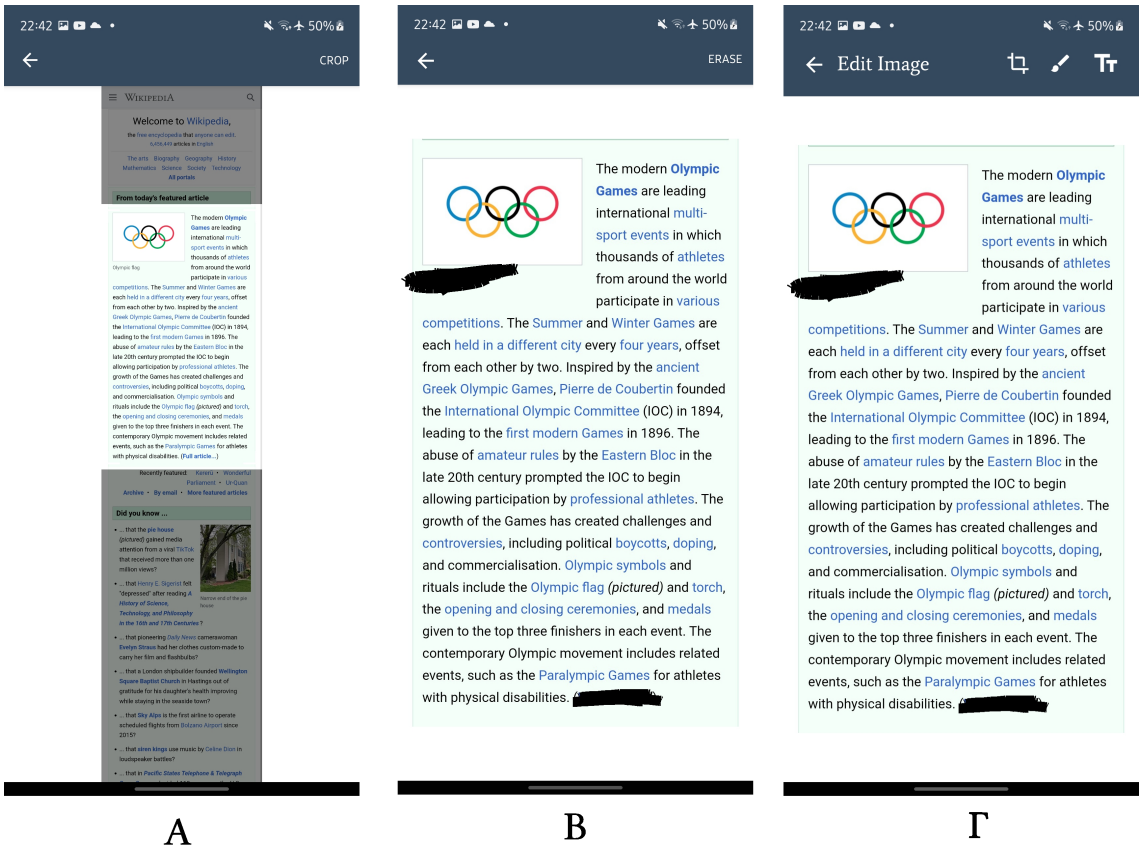
είναι με σειρά εμφάνισης από αριστερά προς τα δεξιά ένα εικονίδιο για το κόψιμο της εικόνας (Cropper), ένα εικονίδιο για σβήσιμο πληροφοριών (Eraser) και ένα εικονίδιο για την εξαγωγή κειμένου απ'π την εικόνα (Text). Έτσι ο χρήστης μπορεί να εκτελέσει όσες φορές θέλει τις ενέργειες κοψίματος και σβησίματος, ενώ κάθε φορά μετά από κάποια αλλαγή επιστρέφει στην οθόνη της ImageEditActivity με τη νέα εικόνα να παρουσιάζεται σε αυτή. Το αποτέλεσμα που θα προκύψει είναι το κείμενο που στη συνέχεια θα είναι η εισόδους στους αλγορίθμους που δοκιμάζουμε για αυτό και πρέπει να γίνει προσεκτικά η επιλογή του. Περισσότερες πληροφορίες για τα εικονίδια αυτά δίνονται παρακάτω.



Εικόνα 5.3: *Διεπαφή της ImageEditActivity*

1. **Εικονίδιο κοψίματος:** το οποίο μπορεί να χρησιμοποιηθεί από τον χρήστη στην περίπτωση που θέλει να αλλάξει το μέγεθος της εικόνας που παράγεται και να την κόψει στο σημείο που εκείνος επιθυμεί. Για το κόψιμο της εικόνας χρησιμοποιείται η βιβλιοθήκη ανοιχτού κώδικα ImageCropper, του Arthur Hub η οποία είναι διαθέσιμη στη σελίδα του Github [78]. Έτσι με το πάτημα του εικονιδίου αυτού ξεκινάει δραστηριότητα που είναι διαθέσιμη από τον ImageCropper, όπως φαίνεται στην Εικόνα 5.4, Α. Συγκεκριμένα, ο χρήστης μπορεί να κόψει την σελίδα στο μέγεθος που αυτός επιθυμεί και στην συνέχεια πατώντας το εικονίδιο με τίτλο "CROP" της οθόνης να επιστρέψει στην ImageEditActivity με την νέα εικόνα.

2. **Εικονίδιο σβησίματος:** το οποίο μπορεί να χρησιμοποιηθεί από τον χρήστη στην περίπτωση που θέλει να σβήσει περιττές πληροφορίες από το κείμενο, λεζάντες από ει-



Εικόνα 5.4: *Διεπαφές των Cropper και Eraser*

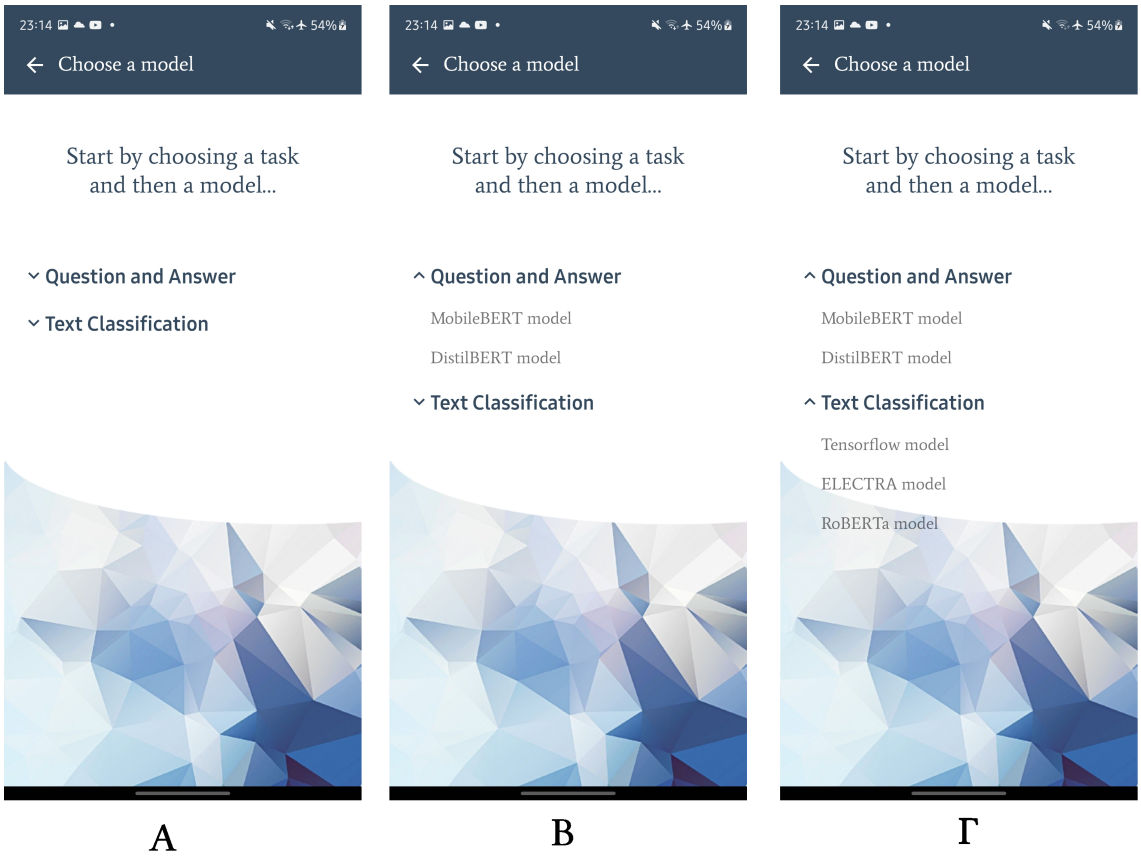
κόνες ή κείμενο μέσα σε εικόνες που τυχόν υπάρχουν στην ιστοσελίδα. Συγκεκριμένα, εκεί ο χρήστης χρησιμοποιώντας το δάκτυλο του μπορεί να σβήσει ότι πληροφορία θέλει σαν να γράφει με τη χρήση του δάκτυλου του, και το αποτέλεσμα μοιάζει όπως φαίνεται στην Εικόνα 5.4, Β. Ο λόγος που χρειαζόμασταν την ενέργεια σβησίματος ε-ίναι το ότι πολλές φορές όταν υπάρχει κείμενο σε σημεία όπου δεν μπορούν να κοπούν με τον ImageCropper η συνάρτηση που κάνει την εξαγωγή κειμένου λάμβανε υπόψιν και αυτά τα σημεία εκτός του βασικού κειμένου, γεγονός που οδηγούσε σε λανθα-σμένο κείμενο χωρίς συνοχή. Πρακτικά με το πάτημα του εικονιδίου για το σβήσιμο ο χρήστης οδηγείται σε μια νέα οθόνη που ουσιαστικά είναι η δραστηριότητα Eraser-activity, όπου η εικόνα εμφανίζεται σε μια νέα οθόνη που έχει δημιουργηθεί, όπου δίνεται δυνατότητα στο χρήστη να αλληλεπιδράσει με την οθόνη και έτσι να πραγματο-ποιήσει ενέργειες ζωγραφικής. Η δραστηριότητα της ζωγραφικής πραγματοποιούνται μέσω των βιβλιοθηκών Canvas και Paint [77, 79]. Ο χρήστης μπορεί μέσα στη δραστη-ριότητα Eraseractivity να κάνει τις αλλαγές που επιθυμεί και στη συνέχεια πατώντας το εικονίδιο με τίτλο “ERASE” της οθόνης να επιστρέψει στην ImageEditActivity με τη νέα εικόνα (Εικόνα 5.4 Γ).

3. **Εικονίδιο εξαγωγής κειμένου:** το οποίο μπορεί να χρησιμοποιηθεί από τον χρήστη στην περίπτωση που θέλει να προχωρήσει με την εξαγωγή κειμένου από την εικόνα προκειμένου το παραγόμενο κείμενο το οποίο στη συνέχεια θα δοθεί σαν είσοδος σε κάποιο από τα διαθέσιμα μοντέλα. Η εξαγωγή του κειμένου γίνεται με το χώρισμα της

εικόνας σε κομμάτια (frames). Από το καθένα από αυτά τα κομμάτια εξάγεται κείμενο χαρακτήρας προς χαρακτήρα μέσω αναγνώρισης συμβόλων από εικόνα και έτσι στη συνέχεια εξάγεται το κείμενο διαδοχικά σε μορφή συμβολοσειράς (string). Με το πάτημα του εικονιδίου αυτού λοιπόν αφού γίνει η εξαγωγή κειμένου και πρακτικά ο χρήστης μεταφέρεται στην επόμενη δραστηριότητα της εφαρμογής, την ChooseModelActivity.

Βήμα 3: ChooseModelActivity

Η δραστηριότητα ChooseModelActivity εμφανίζεται μετά τις οθόνες επεξεργασίας της εικόνας. Η διεπαφή της δραστηριότητας αυτής φαίνεται στην Εικόνα 5.5. Σε αυτή τη δραστηριότητα ο χρήστης είναι σε θέση να επιλέξει ποιο μοντέλο θα εκτελεστεί για την εργασία που επιθυμεί, με είσοδο το κείμενο που έχει εξαχθεί προηγουμένως. Οι εργασίες από τις οποίες μπορεί να επιλέξει είναι η Απάντηση Ερωτήσεων (Question and Answering) ή η Ταξινόμηση Κειμένου (Text Classification). Πρακτικά στην οθόνη ο χρήστης αρχικά βλέπει δύο επιλογές, τις “Question and Answering” και “Text Classification” (Εικόνα 5.5, Α). Ανάλογα με το ποια επιλογή θα πατήσει ξεκινάει ένα μενού με φορά προς τα κάτω, όπου στην συνέχεια ο χρήστης μπορεί να επιλέξει το ποιο μοντέλο θέλει να δει να εκτελείται στην εργασία που επέλεξε (Εικόνες 5.5, Β και Γ). Ο χρήστης πατώντας ένα από τα μοντέλα στη λίστα μετακινείται σε αντίστοιχη δραστηριότητα. Ανάλογα με το τι επιλέγει ο χρήστης μπορεί να βρεθεί στη MobileBertActivity, στη DistilBertActivity, στην TensorFlowActivity, στη RobertaActivity ή στην ElectraActivity αντίστοιχα όπου μπορεί να δει το εκάστοτε μοντέλο να εκτελείται.



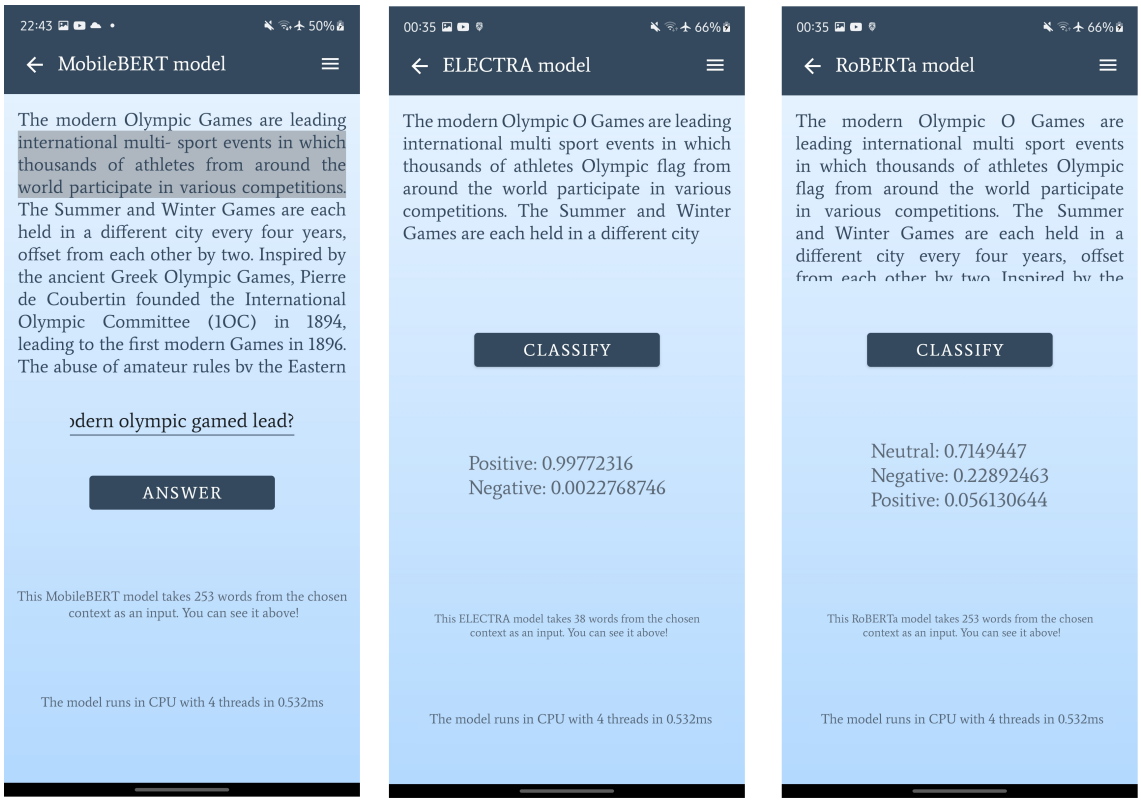
Εικόνα 5.5: *Διεπαφή της ChooseModelActivity*

Βήμα 4: Δραστηριότητες σχετικές με τα μοντέλα

Αρχικά οι δραστηριότητες MobileBertActivity και DistilBertActivity παρουσιάζονται μαζί καθώς η διεπαφές τους είναι ίδιες και φαίνονται στην Εικόνα 5.6, Α (περίπτωση του μοντέλου MobileBERT). Οι διαφορές όσον αφορά τη λειτουργία των δύο οθονών υπάγονται στις διαφορές εκτέλεσης και προεπεξεργασίας των δυο μοντέλων και στην διεπαφή αφορά τα στοιχεία κάθε μοντέλου. Δεδομένου ότι αυτά τα μοντέλα είναι εκπαιδευμένα στην εργασία της Απάντησης Ερωτήσεων, σε κάθε μια από αυτές τις διεπαφές βλέπουμε το κείμενο που έχει παραχθεί από τα προηγούμενα στάδια, καθώς και ένα πλαίσιο, όπου ο χρήστης μπορεί να εισάγει την ερώτηση του, ένα κουμπί με τίτλο “ANSWER", ένα κείμενο με πληροφορίες για το μέγεθος της εισόδου του μοντέλου, καθώς και ένα κείμενο με πληροφορίες για το πως εκτελείται το μοντέλο και σε πόσο χρόνο. Όσον αφορά το κείμενο που εμφανίζεται, σε αυτό φαίνονται οι n πρώτες λέξεις από το κείμενο που έχει προκύψει από την εικόνα και όχι όλο το κείμενο της εικόνας. Συγκεκριμένα, n είναι ο αριθμός των λέξεων που μπορεί να δεχτεί κάθε μοντέλο, αν λάβουμε προφανώς υπόψιν και τα ειδικά tokens. Με λίγα λόγια ο χρήστης μπορεί να δει ακριβώς ποιες λέξεις μπορεί να επεξεργαστεί το μοντέλο από την είσοδο που θέλει και να κάνει την ερώτηση του ανάλογα με αυτό. Με το πάτημα του κουμπιού με τίτλο “ANSWER" η απάντηση του μοντέλου υπογραμμίζεται πάνω στο κείμενο. Ο χρήστης μπορεί να πληκτρολογήσει όσες ερωτήσεις θέλει σχετικά με το κείμενο και να βλέπει κάθε φορά την απάντηση να εμφανίζεται μπροστά του.

Στη συνέχεια, οι δραστηριότητες TensorFlowActivity, RobertaActivity και ElectraActivity παρουσιάζονται μαζί καθώς η διεπαφές τους είναι αντίστοιχες και φαίνονται στις Εικόνες 5.6 Β και Γ. Οι τρεις αυτές δραστηριότητες, ξεκινούν εάν ο χρήστης επιλέξει από τα μοντέλα το TensorFlow, το RoBERTa ή τέλος το ELECTRA, τα οποία εκτελούν την εργασία της Ταξινόμησης Κειμένου. Σε κάθε μια από αυτές τις διεπαφές βλέπουμε το κείμενο το οποίο θα εισαχθεί ως είσοδος στο μοντέλο, ένα κουμπί με τίτλο “CLASSIFY”, ένα κείμενο με πληροφορίες για το μέγεθος της εισόδου που μπορεί να δεχθεί το μοντέλο, καθώς και ένα κείμενο με πληροφορίες για το που τρέχει το μοντέλο και σε πόσο χρόνο. Με το πάτημα του κουμπιού με τίτλο “CLASSIFY” η απάντηση του μοντέλου εμφανίζεται μεταξύ του κειμένου με τις πληροφορίες εισόδου. Η απάντηση του μοντέλου είναι η αντίστοιχη πιθανότητα θετικότητας, αρνητικότητας ή και ουδετερότητας της χροιάς του δοθέντος κειμένου, όπως έχει ήδη περιγραφεί προηγουμένως. Στην περίπτωση των TensorFlow και ELECTRA το μοντέλο αξιολογεί το κείμενο ως προς την θετικότητα και την αρνητικότητα του (Εικόνα 5.6, Β), ενώ στο RoBERTa και ως προς την ουδετερότητα του (Εικόνα 5.6, Γ).

Σε κάθε μια από τις προαναφερθείσες δραστηριότητες υπάρχει πάνω δεξιά στην μπάρα της εφαρμογής, όπως φαίνεται και στις διεπαφές τους, ένα εικονίδιο μενού. Πατώντας το ο χρήστης προκαλεί την εμφάνιση ενός μενού που εμφανίζεται από το κάτω μέρος της οθόνης στο οποίο μπορεί να επιλέξει την παράμετρο εκτέλεσης του μοντέλου. Στην συγκεκριμένη εφαρμογή, ασχολούμαστε με πέντε διαφορετικές παραμέτρους εκτέλεσης (configurations) για την εκτέλεση του μοντέλου οι οποίες φαίνονται στην Εικόνα 5.7, όπου φαίνεται η διεπαφή χρήστη που σχετίζεται με το μενού που προαναφέρθηκε. Στη συνέχεια αφού ο χρήστης επιλέξει την παράμετρο εκτέλεσης που επιθυμεί, πατάει το κουμπί με τίτλο "OK" προκειμένου να επιστρέψει στην σελίδα εκτέλεσης του μοντέλου. Ο χρήστης μέσω αυτής της επιλογής



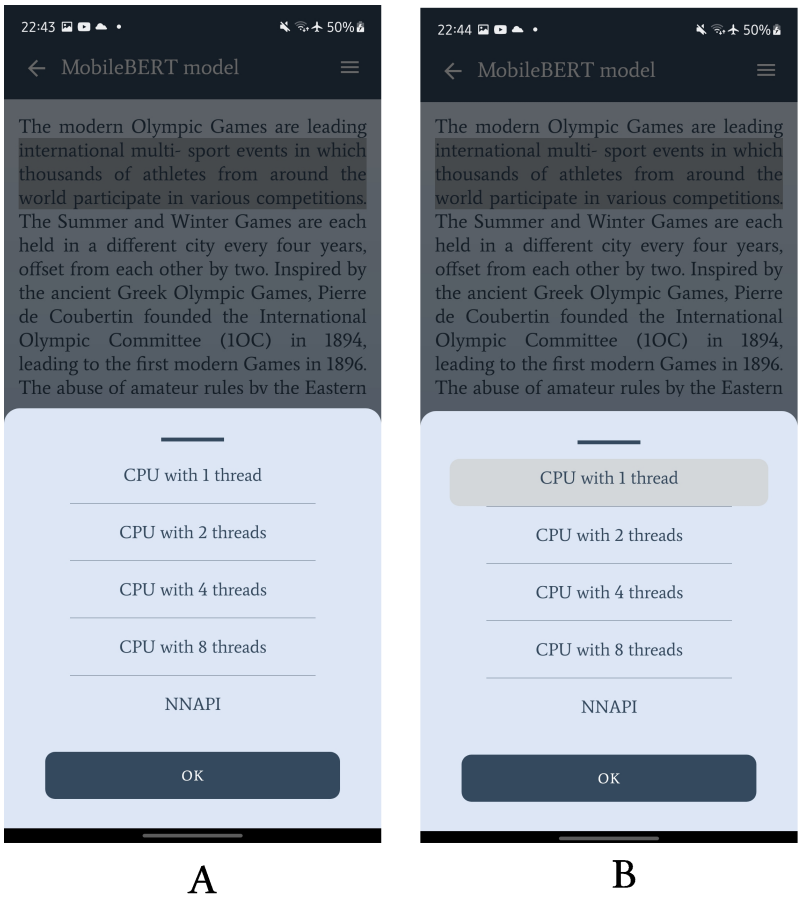
Εικόνα 5.6: *Διεπαφές των δραστηριοτήτων που αφορούν τα μοντέλα*

έχει την δυνατότητα να δει το μοντέλο να εκτελείται, όσες φορές θέλει, με όποια παράμετρο εκτέλεσης επιλέγει κάθε φορά. Αυτή η επιλογή είναι σημαντική καθώς μπορούμε να μελετήσουμε ταυτόχρονα όλες τις διαθέσιμες παραμέτρους και να δούμε τυχόν διαφορές τους στον χρόνο εκτέλεσης.

5.2.1 Εκτέλεση των μοντέλων στην εφαρμογή

Αφού έγινε η ανάλυση των διεπαφών του χρήστη ήρθε η ώρα να γίνει η ανάλυση των λειτουργιών των δραστηριοτήτων των μοντέλων. Τα μοντέλα εκτελούνται σε μια κλάση (client) και μάλιστα η εκτέλεση τους γίνεται σε νήματα, τα handler threads, στο παρασκήνιο της εφαρμογής [46]. Μετά τη δημιουργία του νήματος, το μοντέλο το οποίο βρίσκεται στη μορφή ενός TensorFlow Lite αρχείου σε ένα φάκελο της εφαρμογής φορτώνεται στη προσωρινή μνήμη της εφαρμογής, ενώ τα επιπλέον αρχεία που μπορεί να συνοδεύουν το εκάστοτε μοντέλο και βρίσκονται στα Metadata του αρχείου φορτώνονται επίσης σε τοπικές μεταβλητές. Στη συνέχεια, αρχικοποιείται ένας TensorFlow Lite Interpreter μέσω του οποίου τρέχει το μοντέλο. Ο Interpreter χρησιμοποιεί μια στατική σειρά γραφημάτων και έναν προσαρμοσμένο (λιγότερο δυναμικό) εκχωρητή μνήμης για να εξασφαλίσει ελάχιστο φόρτο, αρχικοποίηση και καθυστέρηση εκτέλεσης. Μέσω του Interpreter ορίζουμε την επιλογή υλικού όπου θα τρέξει το μοντέλο [80]. Οι επιλογές που δίνονται μέσω αυτού και δοκιμάζονται στην παρούσα εφαρμογή είναι οι εξής:

1. Το μοντέλο να τρέχει στην μονάδα CPU με εφαρμογή παράλληλης εκτέλεσης μέσω των



Εικόνα 5.7: *Μενού για την επιλογή παραμέτρου εκτέλεσης για το μοντέλο*

νημάτων (Threads).

2. Το μοντέλο να τρέχει στην μονάδα GPU. Η επιλογή αυτή απαιτεί τα βάρη των μοντέλων να είναι κινητής που χρησιμοποιούνται να αναπαρίστανται ως αριθμοί κινητής υποδιαστολής floating point numbers.

3. Το μοντέλο μπορεί να εκτελείται μέσω του NNAPI (Neural Network API) της Google. Το NNAPI είναι ένα API που υποστηρίζεται από συσκευές που τρέχουν από Android 8.1 και παρέχει επιτάχυνση για μοντέλα TensorFlow Lite σε συσκευές Android με υποστηριζόμενους επιταχυντές υλικού, όπως οι GPU, DSP και NPU. Η απόδοση του NNAPI ποικίλλει ανάλογα με το συγκεκριμένο υλικό που είναι διαθέσιμο στη συσκευή [80, 81].

Στη συγκεκριμένη περίπτωση, οι συσκευές που χρησιμοποιούνται για τις μετρήσεις και τη δοκιμή της εφαρμογής έχουν και οι δύο οκταπύρηνο επεξεργαστή για αυτό το λόγο πραγματοποιούνται εκτελέσεις στη CPU με 1,2,4 και 8 νήματα. Στη συνέχεια, πραγματοποιούνται και εκτελέσεις στο NNAPI.

Έτσι λοιπόν αφού γίνει η αρχικοποίηση όλων των παραπάνω ξεκινάει η διαδικασία της προεπεξεργασίας εισόδου. Συγκεκριμένα, για κάθε μοντέλο πραγματοποιείται η διαδικασία τμηματοποίησης η οποία αναπτύχθηκε στην Java. Στην περίπτωση των μοντέλων που

χρησιμοποιούν τον BERT Tokenizer, αυτός αναπτύσσεται μέσω μια κλάσης Full Tokenizer, όπου εκτελούνται διαδοχικά ένας Basic Tokenizer ο οποίος κάνει τον διαχωρισμό των λέξεων και την αφαίρεση βοηθητικών χαρακτήρων και σημείων στίξης και στη συνέχεια εκτελείται ένας Wordpiece Tokenizer. Σε άλλες περιπτώσεις όπου γίνεται η εφαρμογή απλού Basic Tokenizer, στον οποίο γίνεται διαχωρισμός των λέξεων μιας πρότασης και απόδοση ids σε κάθε μια από αυτές, ή του RoBERTa Tokenizer εκτελούνται οι αντίστοιχες συναρτήσεις για την δημιουργία της απαιτούμενης εξόδου για κάθε μια από τις περιπτώσεις.

5.2.2 Μεταφορά δεδομένων

Προηγουμένως παρουσιάστηκαν οι δραστηριότητες της εφαρμογής καθώς και η σειρά με την οποία ο χρήστης περιηγείται μέσα σε αυτές. Σε αυτό το σημείο ήρθε η ώρα να αναφερθεί το πως γίνεται η μεταφορά πληροφοριών μεταξύ των δραστηριοτήτων. Πιο συγκεκριμένα, αρχικά στην MainActivity γίνεται η καταγραφή του κειμένου του διαδικασία από την οποία προκύπτει ένα Bitmap, όπως έχει παρουσιαστεί παραπάνω. Το Bitmap αυτό προκειμένου να παρουσιαστεί και να γίνει η επεξεργασία του στην ImageEditActivity πρέπει να μεταφερθεί από την πρώτη στη δεύτερη. Ένας κοινός τρόπος μεταφοράς στοιχείων, όπως συμβολοσειρών (strings) και bitmaps από μια δραστηριότητα σε μία άλλη είναι το Intent. Πιο αναλυτικά, το Intent είναι μια παθητική δομή δεδομένων που περιέχει μια αφηρημένη περιγραφή μιας ενέργειας που πρέπει να εκτελεστεί. Η πιο σημαντική χρήση του είναι στην έναρξη Activities, όπου μπορεί να θεωρηθεί ως ο συνδετικός μεταξύ των δραστηριοτήτων τόσο διότι μέσω αυτού γίνεται η μεταφορά από μια activity σε μια άλλη όσο και επειδή μέσα σε ένα Intent μπορεί να αποθηκευτεί πληροφορία διαφόρων τύπων δεδομένων [82].

Στην συγκεκριμένη περίπτωση ωστόσο, τα πράγματα δεν ήταν τόσο απλά. Οι πληροφορίες που μπορούν να αποθηκευτούν σε ένα Bitmap αρχείο έχουν όριο όσον αφορά το μέγεθος τους. Έτσι λοιπόν το Bitmap δεν μπορεί να μεταφερθεί αυτούσιο μέσω ενός Intent. Στη συνέχεια, δοκιμάστηκε μια άλλη κοινή πρακτική για την μεταφορά του Bitmap που είναι η μετατροπή του σε πίνακα από Bytes δεδομένων (ByteArray). Δυστυχώς και σε αυτή την περίπτωση το μέγεθος των Bytes που μπορούμε να μεταφέρουμε δεν είναι αρκετό έτσι ώστε να μεταφερθεί η εικόνα μας αφού αυτή στις περισσότερες περιπτώσεις έχει μεγάλο ύψος. Προκειμένου λοιπόν να ξεπεραστεί αυτό το πρόβλημα και δεδομένου ότι δεν θέλουμε να αποθηκεύουμε τίποτα στο κινητό έτσι ώστε να μην γεμίζουμε την εσωτερική (Internal) ή την εξωτερική (External) μνήμη του, χρησιμοποιήθηκε μια κλάση η οποία αναπτύσσεται στηριζόμενη στην έννοια του View Model, στοιχείο της αρχιτεκτονικής του MVVM, το οποίο αναφέρεται στο Κεφάλαιο 3. Έτσι, λοιπόν εκμεταλλευόμενοι την ικανότητα σύνδεσης του View Model δημιουργούμε μια δομή η οποία θα μεταφέρει πληροφορίες μεταξύ των δραστηριοτήτων της παρούσας εφαρμογής. Συγκεκριμένα, μέσω της κλάσης NewInstanceFactory της ViewModelProvider δημιουργούμε σε κάθε μια δραστηριότητα μια εκδοχή (instance) του View Model. Μέσα στο View Model λοιπόν αποθηκεύεται η εικόνα που θέλουμε να κοινοποιήσουμε μεταξύ των δραστηριοτήτων και έτσι μεταφέρεται από την μια στην άλλη [45].

Κεφάλαιο 6

Μετρήσεις και Αποτελέσματα

Στο κεφάλαιο αυτό θα γίνουν μετρήσεις σχετικά με τον χρόνο εκτέλεσης συγκεκριμένων λειτουργιών της εφαρμογής. Οι ενέργειες αυτές σχετίζονται τόσο με την εφαρμογή αυτή καθ' αυτή όσο και με τους αλγορίθμους Βαθιάς Μάθησης που έχουν εισαχθεί και εκτελούνται σε αυτή. Στη συνέχεια, θα γίνει αξιολόγηση των αποτελεσμάτων που προέκυψαν από τις μετρήσεις, σε δύο συσκευές μέσω συγκεκριμένων μετρικών.

6.1 Μετρήσεις

Στην παρούσα μελέτη, προκειμένου να ληφθεί ένα συμπέρασμα για την απόδοση του κάθε μοντέλου στην κινητή συσκευή, λαμβάνεται μια σειρά μετρήσεων χρόνου. Οι μετρήσεις αυτές πάρθηκαν με την χρήση της συνάρτησης System.currentTimeMillis() η οποία υπολογίζει την χρονική στιγμή που εκτελείται η εντολή σε Milliseconds. Με αυτόν τον τρόπο προκειμένου να μετρήσουμε το χρόνο που διαρκεί μια συγκεκριμένη λειτουργία μετράμε την χρονική στιγμή που ξεκινάει το κομμάτι του κώδικα της λειτουργίας αυτής, καθώς και το σημείου όπου τελειώνει. Τελικώς, αφαιρώντας τις δυο αυτές τιμές μπορούμε να βρούμε τον συνολικό χρόνο που διήρκεσε η λειτουργία αυτή. Συγκεκριμένα, οι μετρήσεις που έγιναν αφορούν:

- Τον χρόνο που απαιτείται για την καταγραφή της οθόνης και την δημιουργία της εικόνας (Screenshot Time).
- Τον χρόνο που απαιτείται για την εξαγωγή του κειμένου από την εικόνα που έχει δημιουργηθεί (Detect Time).
- Τον χρόνο που απαιτείται για την προεπεξεργασία των εισόδων των μοντέλων, δηλαδή τις διαδικασίες τμηματοποίησης (Tokenization Time).
- Τον χρόνο που απαιτείται για το στάδιο συμπερασματολογίας κάθε μοντέλου σε κάθε παράμετρο εκτέλεσης (Inference Time).

Στις τρεις πρώτες περιπτώσεις, η κάθε μέτρηση απαιτούσε την διαδικασία επιλογής και επεξεργασίας της εικόνας από τον χρήστη. Με λίγα λόγια, σε κάθε μέτρηση ο χρήστης έπρεπε να πληκτρολογήσει την ηλεκτρονική διεύθυνση της σελίδας, όπου εμπεριέχεται κείμενο

που θέλει να χρησιμοποιήσει ως είσοδο στα μοντέλα, στη συνέχεια να επεξεργαστεί την εικόνα που έχει προκύψει και τέλος να ξεκινήσει την διαδικασία εξαγωγής κειμένου από την τελική μορφή της εικόνας. Προκειμένου να ληφθούν οι απαραίτητες πληροφορίες χρόνου, η διαδικασία εύρεσης και εξαγωγής κειμένου πραγματοποιήθηκε 20 φορές για διαφορετικά μεγέθη σελίδων και κειμένων. Οι χρόνοι αυτοί ουσιαστικά αφορούν την αρχιτεκτονική της εφαρμογής και δεν συνδέονται με τα μοντέλα.

Στην περίπτωση του χρόνου συμπερασματολογίας θέλαμε οι μετρήσεις να είναι πιο ακριβείς σχετικά με την πραγματικότητα και δεδομένου ότι καταφέραμε να μην απαιτούσαν μεγάλη διεργασία από τον χρήστη, πάρθηκαν 100 μετρήσεις για κάθε παράμετρο εκτέλεσης, σε καθένα από τα 5 μοντέλα που υπάρχουν στην εφαρμογή. Προκειμένου να κάνουμε πιο εύκολη την διαδικασία συλλογής των μετρήσεων, έγιναν κάποιες ενέργειες, έτσι ώστε να πραγματοποιηθεί ένα είδος αυτοματισμού της διαδικασίας. Πιο αναλυτικά, δημιουργήθηκε μια συνάρτηση όπου γίνεται υπολογισμός τυχαίων εισόδων για το κάθε μοντέλο. Δεδομένου ότι οι είσοδοι των μοντέλων είναι διανύσματα αριθμών, αυτό καθιστά ευκολότερη τη διαδικασία, αφού κάθε φορά μέσω της συνάρτησης Random παράγονται τυχαίοι αριθμοί που αντιστοιχούν σε καθένα από τα ids των παραγόμενων tokens από το υποτιθέμενο κείμενο εισόδου. Έτσι λοιπόν, με την εισαγωγή αυτών, καθώς και την εισαγωγή των ids που αντιστοιχούν στα ειδικά tokens, δημιουργείται η είσοδος στη μορφή που την χρειάζεται κάθε μοντέλο. Πρακτικά κάθε φορά που πατιέται το κουμπί εκτέλεσης ενός μοντέλου, εκτελείται η συμπερασματολογία με διαφορετική είσοδο. Η διαδικασία αυτή επαναλαμβάνεται για όλες τις παραμέτρους εκτέλεσης και για κάθε μοντέλο. Οι μετρήσεις συλλέγονται προκειμένου να υπολογιστούν συγκεκριμένες μετρικές.

6.2 Μετρικές

Η αξιολόγηση των αποτελεσμάτων των μετρήσεων χρόνου έγιναν με βάση τις παρακάτω μετρικές:

1. **Average:** Η τιμή που αναπαριστά τον μέσο όρο όλων των μετρήσεων.
2. **Median:** Η τιμή που χωρίζει το ταξινομημένο σύνολο των μετρήσεων στην μέση.
3. **Minimum:** Η ελάχιστη τιμή όλων των μετρήσεων.
4. **Maximum:** Η μέγιστη τιμή όλων των μετρήσεων.
5. **90th Percentile:** Η τιμή κάτω από την οποία βρίσκεται το 90% των μετρήσεων.

Αφού γίνει ο υπολογισμός των μετρικών για κάθε ομάδα τιμών για τα διάφορα στάδια προεπεξεργασίας και για τις διάφορες παραμέτρους εκτέλεσης, είμαστε σε θέση να έχουμε μια εικόνα για την χρονική απόδοση, τόσο της εφαρμογής όσο και των μοντέλων που τρέχουν σε αυτή. Συνολικά, από τις τιμές αυτές θα έχουμε μια εικόνα σχετικά με το που κυμαίνονται οι μετρήσεις σε κάθε περίπτωση. Με αυτόν τον τρόπο, θα είμαστε σε θέση να συγκρίνουμε τις αποδόσεις της εφαρμογής συνολικά στις δύο συσκευές, καθώς και τις αποδόσεις των μοντέλων στις εργασίες και να αποφανθούμε για το ποια μοντέλα λειτουργούν καλύτερα για την ίδια εργασία στις κινητές συσκευές που χρησιμοποιούμε.

	Samsung Galaxy S20 FE	**Samsung Galaxy A10**
Έτος Κυκλοφορίας	2020	2019
CPU	Octa-core (2x2.50 GHz Cortex-A76 & 4x2.00 GHz Cortex-A55 & 2x2.73 Exynos M5	Octa-core (2x1.6 GHz Cortex-A73 & 6x1.35 GHz Cortex-A53)
GPU	Mali-G77 MP11	Mali-G71 MP2
Συνολική RAM	6 GB	2 GB
Έκδοση λογισμικού	Android 11	Android 11

Πίνακας 6.1: *Χαρακτηριστικά συσκευών που χρησιμοποιήθηκαν για τις μετρήσεις*

6.3 Αποτελέσματα

Τα αποτελέσματα έχουν ληφθεί ύστερα από δοκιμές σε δύο διαφορετικές συσκευές. Οι συσκευές αυτές είναι ένα Samsung Galaxy S20 FE και ένα Samsung Galaxy A10 με τα χαρακτηριστικά τους να φαίνονται στον Πίνακα 6.1. Με μια γρήγορη ματιά, περιμένει κανείς οι χρόνοι εκτέλεσης των διαφόρων σταδίων να είναι πιο μικροί στην περίπτωση του Samsung Galaxy S20 FE, καθώς είναι πιο καινούριο τηλέφωνο και με καλύτερα χαρακτηριστικά. Συγκεκριμένα, το Samsung Galaxy S20 FE έχει πιο σύγχρονες CPU και GPU, ενώ έχει και μεγαλύτερη μνήμη RAM. Λόγω της καλύτερης CPU στην περίπτωση του Samsung Galaxy S20 FE περιμένουμε να δούμε αισθητά καλύτερα αποτελέσματα. Επιπλέον το Samsung Galaxy A10 είναι ένα τηλέφωνο με κάποια χρόνια έντονης χρήσης στο ιστορικό του, για αυτό και μπορούμε να παρατηρήσουμε ξεκάθαρα μια καθυστέρηση τόσο στο μενού όσο και στις υπόλοιπες εφαρμογές, ειδικά σε σχέση με το Samsung Galaxy S20 FE που είναι πιο καινούριο τηλέφωνο.

Πρακτικά, σε πρώτη φάση λαμβάνονται μετρήσεις χρόνου για όλα τα στάδια που προαναφέρθηκαν για τη συσκευή Samsung Galaxy S20 FE και στη συνέχεια για το Samsung Galaxy A10. Αρχικά στον Πίνακα 6.2 παρουσιάζονται οι μετρικές των μετρήσεων για κάθε στάδιο της συλλογής και επεξεργασίας της εικόνας, ενώ στην συνέχεια, στους Πίνακες 6.3, 6.4, 6.5 και 6.6 βλέπουμε τις μετρικές των μετρήσεων για τα στάδια της συμπερασματολογίας των μοντέλων και για τις δυο συσκευές. Σε αυτούς με κίτρινο χρώμα σημειώνεται το σε ποια παράμετρο εκτέλεσης τρέχει καλύτερα το κάθε μοντέλο. Με γαλάζιο σημειώνεται το σε ποια παράμετρο εκτέλεσης τρέχει καλύτερα κάποιο μοντέλο στο A10, αν αυτή είναι διαφορετική από του S20. Στα αποτελέσματα δεν συμπεριλαμβάνονται οι μετρικές του μοντέλου TensorFlow αφού οι τιμές βγαίνουν κάτω του ενός ms και άρα δεν αξίζει να γίνει περαιτέρω διερεύνηση, αφού ο χρόνος είναι πολύ μικρός.

6.4 Αξιολόγηση μετρήσεων

Αν παρατηρήσουμε τους πίνακες των μετρικών, θα διαπιστώσουμε ότι οι προβλέψεις μας υπέρ της συσκευής Samsung Galaxy S20 FE έχουν επιβεβαιωθεί. Αρχικά, ξεκινώντας από τον Πίνακα 6.2 πρέπει να αποσαφηνίσουμε ότι οι μετρήσεις έγιναν με διαφορετικές, τυχαίες εισόδους για την κάθε συσκευή. Οι είσοδοι αυτές είναι διαφόρων μεγεθών προκείμενου

	Average		Median		Min		Max		90P	
	S20	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**
Screenshot	206.7	327.1	186.5	329	134	9	313	740	263.2	501.3
Detect	184.3	2601	164.5	2642	95	5	539	5401	222.6	5079
Tokenization MobileBert	7.5	23.85	4	17.5	2	1	38	65	11.3	50
Tokenization DistilBert	3.95	13.9	3.5	8.5	1	1	8	36	7	33.1
Tokenization TensorFlow	1.8	6.5	1	3	0	0	6	50	3.1	10.7
Tokenization ELECTRA	3.9	7.9	4	4.5	2	1	8	29	6.1	17.6
Tokenization RoBERTa	2.05	4	1.5	3.5	1	1	6	10	5	7.2

Πίνακας 6.2: *Αποτελέσματα μετρικών των μετρήσεων κατά την επεξεργασία εισόδου (msec)*

	Average		Median		Min		Max		90P	
	S20	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**
CPU(1)	458.59	3223	458	3221	442	3188	478	3315	469	3239
CPU(2)	364.8	1894	376.5	1891	288	1815	460	2024	394	1914
CPU(4)	355.7	1764	353	1754	318	1651	434	1957	374.2	1823
CPU(8)	670.7	1815	601.5	1775	412	1506	1612	5407	1002	1974
NNAPI	451.9	3219	451	3214	447	3182	471	3311	457	3248

Πίνακας 6.3: *Αποτελέσματα μετρικών κατά την συμπερασματολογία του MobileBERT (msec)*

	Average		Median		Min		Max		90P	
	S20	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**
CPU(1)	382.38	4526	379	4521	376	4463	403	4483	396	4563
CPU(2)	270.77	2920	269	2910	260	2866	296	3392	277.1	2944
CPU(4)	286.9	2435	288	2436	256	2366	313	2503	307	2476
CPU(8)	553	2093	543.5	2083	419	1919	759	2431	665.1	2192
NNAPI	418.1	4610	418	4612	412	4549	424	4680	421	4648

Πίνακας 6.4: *Αποτελέσματα μετρικών στη συμπερασματολογία του DistilBERT (msec)*

	Average		Median		Min		Max		90P	
	S20	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**
CPU(1)	467.2	2918	470	2910	458	2866	479	3392	474.1	2944
CPU(2)	335	3670	333	3664	322	3563	399	3960	342	3707
CPU(4)	327.6	3204	325	3211	297	3059	398	3338	342.1	3266
CPU(8)	568.8	3135	569.5	3032	343	2816	959	7999	738.8	3297
NNAPI	509.5	5748	508	5736	488	5647	571	6383	516	5783

Πίνακας 6.5: *Αποτελέσματα μετρικών στη συμπερασματολογία του RoBERTa (msec)*

να υπάρχει ποικιλομορφία στις μετρήσεις. Παρακάτω παρουσιάζονται τα συμπεράσματα ανάλογα με το στάδιο που μελετάται:

1. **Στάδιο του Screenshot.** Προφανώς, όσο πιο μεγάλο περιεχόμενο έχει η σελίδα τόσο μεγαλύτερος είναι ο χρόνος που απαιτείται για την καταγραφή του. Παρατηρούμε ότι ο μέσος χρόνος για την καταγραφή στην περίπτωση της συσκευής S20 FE είναι μικρότερος από αυτόν στο A10 και μάλιστα είναι περίπου ίσος με 0.2s, χρόνος πολύ

	Average		Median		Min		Max		90P	
	S20	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**	**S20**	**A10**
CPU(1)	23.22	115.5	24	116	20	110	38	125	25	117
CPU(2)	28.26	102.1	28	102	24	93	33	113	30	107
CPU(4)	28.35	171.4	28	168	22	114	43	247	33	199.1
CPU(8)	175.94	588.3	155.5	573	26	395	374	835	284.8	669.4
NNAPI	28.38	117.4	28	117	25	111	36	129	30	121

Πίνακας 6.6: *Αποτελέσματα μετρικών στη συμπερασματολογία του ELECTRA (msec)*

μικρός σαν μέγεθος. Γενικά και στις δυο συσκευές ο χρόνος είναι τόσο μικρός που δεν επηρεάζει την εμπειρία του χρήστη στην εφαρμογή.

2. **Στάδιο του Detect.** Στην περίπτωση του χρόνου για την εξαγωγή του κειμένου από την εικόνα δεν μπορούμε να πούμε το ίδιο με το προηγούμενο στάδιο. Σε αυτή την περίπτωση παρατηρούμε μέγιστο χρόνο ίσο με 5 δευτερόλεπτα στην περίπτωση του A10. Ο χρόνος αυτός είναι αρκετός οπότε θα μπορούσε να γίνει διερεύνηση άλλων τεχνικών όπως, διαφορετικοί αλγόριθμοι οπτικής αναγνώρισης χαρακτήρων (OCR: Optical Character Recognition).

3. **Στάδιο του Tokenization.** Στην περίπτωση του χρόνου για τη τμηματοποίηση και αυτός είναι αρκετά μικρός σε όλες τις περιπτώσεις έτσι ώστε να είναι αποδεκτός και να μην καταστρέφει την εμπειρία του χρήστη.

Στη συνέχεια, παραδίδονται τα αποτελέσματα για τη συμπερασματολογία κάθε μοντέλου στους Πίνακες 6.3, 6.4, 6.5 και 6.6. Και σε αυτό το κομμάτι οι επιδόσεις του S20 είναι πολύ καλύτερες από του A10. Αυτό είναι απόλυτα λογικό, αφού, όπως είπαμε, το πρώτο έχει καλύτερα χαρακτηριστικά και μάλιστα καλύτερη CPU κάτι πολύ σημαντικό για τις συγκεκριμένες εκτελέσεις. Στην περίπτωση της εργασίας Απάντησης Ερωτήσεων το DistilBERT φαίνεται να έχει καλύτερα αποτελέσματα χρονικά για αυτές τις μετρήσεις. Σε αυτό μπορεί να οφείλεται η εκπαίδευση του συγκεκριμένου μοντέλου DistilBERT. Ωστόσο, επειδή εκτελείται σε κινητή συσκευή πολύ σημαντικό είναι ότι το DistilBERT (78.2 MB) έχει μικρότερο μέγεθος από το MobileBERT (96.0 MB) και μάλιστα για περίπου 20 MB. Στη συνέχεια, στην περίπτωση της εργασίας της Ταξινόμησης Κειμένων, σε αυτές τις μετρήσεις φαίνεται καλύτερα αποτελέσματα χρονικά να έχει το μοντέλο της εφαρμογής TensorFlow και μετά το μοντέλο ELECTRA. Αυτό είναι λογικό καθώς το μοντέλο TensorFlow (751 KB) είναι πολύ μικρό σε μέγεθος σε σχέση με τα υπόλοιπα, αφού όπως αναφέραμε έχει μόνο 4 επίπεδα. Στη συνέχεια, της κατάταξης είναι το ELECTRA (13.5 MB) και τέλος το RoBERTa (147 MB) που είναι το πιο μεγάλο και πιο αργό μοντέλο που δοκιμάζεται στην εφαρμογή.

Κεφάλαιο 7

Επίλογος

Στο κεφάλαιο αναλύονται τα συμπεράσματα της παρούσας διπλωματικής εργασίας και η συνεισφορά της στο τομέα. Στο τέλος δίνεται και μια πρόταση για μελλοντικές επεκτάσεις της μελέτης.

7.1 Συμπεράσματα

Στο προηγούμενο Κεφάλαιο παρουσιάστηκαν τα αποτελέσματα των μετρήσεων της εφαρμογής που αναπτύχθηκε καθώς και μια σύντομη ανάλυση. Συνοψίζοντας, η εφαρμογή αυτή αποτελεί ουσιαστικά ένα εργαλείο ανάπτυξης (developer tool) και όχι μια εφαρμογή την οποία μπορεί να χρησιμοποιήσει κάποιος χρήστης και να την έχει για κάποιο λόγο στο κινητό του. Με τέτοιου τύπου δοκιμές καταλαβαίνει κανείς ότι η δημιουργία εφαρμογών με ενσωματωμένους αλγορίθμους Βαθιάς Μάθησης σε τοπικό επίπεδο είναι κάτι παραπάνω από εφικτό. Τα μέσα πλέον είναι πολλά ενώ αυξάνονται καθημερινά. Στα πλαίσια της μελέτης αυτής έγινε σε πρώτη φάση αναζήτηση προεκπαιδευμένων αλγορίθμων που μπορούν να προστεθούν στην εφαρμογή και είναι μοντέλα keras. Η αναζήτηση στο Hugging Face ήταν χρονοβόρα, αφού έπρεπε να βρεθούν μοντέλα στην μορφή TensorFlow προκειμένου να μετατραπούν σε TensorFlow Lite. Πολλά από τα μοντέλα που βρέθηκαν δεν μπορούσαν να μετατραπούν, ενώ άλλα μετατράπηκαν και ή δεν λειτουργούσαν σωστά ή είχαν μεγάλο μέγεθος για την κινητή συσκευή. Επιπλέον πραγματοποιήθηκε και αναζήτηση για μοντέλα έτοιμα σε TensorFlow Lite μορφή αλλά δεν βρέθηκαν πολλά που να είναι κατάλληλα και προεκπαιδευμένα στις εργασίες που μελετώνται. Αυτή η προσπάθεια δείχνει ότι το συγκεκριμένο πεδίο, όπου NLP αλγόριθμοι εκτελούνται σε κινητές συσκευές βρίσκεται ακόμη υπό ανάπτυξη (σε σύγκριση και με την Όραση Υπολογιστών, όπου υπάρχουν δεκάδες προεκπαιδευμένα μοντέλα διαθέσιμα) και χρειάζεται έρευνα.

Επιπλέον, με την βοήθεια της παραπάνω ανάπτυξης οδηγούμαστε σε κάποια συμπεράσματα σχετικά με τα μέσα και τις τεχνικές που χρησιμοποιήθηκαν. Ξεκινώντας από την προεπεξεργασία κειμένου, ο χρόνος αυτός κατά την εκτέλεση ενός αλγορίθμου NLP είναι αρκετά μικρός, αφού τα περισσότερα από τα μοντέλα που χρησιμοποιούνται έχουν είσοδο περιορισμένων λέξεων. Για παράδειγμα, στην παρούσα εφαρμογή τα μοντέλα που δέχονται την μεγαλύτερη είσοδο είναι τα MobileBERT και DistilBERT με είσοδο μεγέθους 384 tokens. Αυτό πρακτικά σημαίνει ότι δεν έχουμε να κάνουμε με τεράστιες εισόδους που θα απαιτούσαν μεγάλο χρόνο προεπεξεργασίας.

Στη συνέχεια, σχετικά με την συμπερασματολογία των μοντέλων παρατηρούμε μια εξάρτηση της εκτέλεσης των μοντέλων από το εκάστοτε υλικό συσκευής και τα χαρακτηριστικά της, κάτι που παρατηρείται μέσω της σύγκρισης μετρήσεων μεταξύ μιας παλιάς και μίας πιο καινούριας συσκευής. Πέρα από την σύγκριση των δύο συσκευών που τα αποτελέσματα είναι τα αναμενόμενα, παρατηρούμε κάτι που δεν φαίνεται λογικό σε πρώτη ανάγνωση. Συγκεκριμένα, αν κρίνουμε από τον βασικό ορισμό της παραλληλοποίησης θα περιμέναμε, όταν το μοντέλο εκτελείται στην CPU με περισσότερα νήματα, να δούμε καλύτερα αποτελέσματα, αφού περισσότερες εργασίες θα γίνονταν ταυτόχρονα. Αυτό ωστόσο δεν επιβεβαιώνεται από τις μετρήσεις. Το γεγονός αυτό οφείλεται αρχικά στο ότι σε έναν πολυπύρηνο επεξεργαστή, αν τα περισσότερα νήματα χρησιμοποιούνται από μια συγκεκριμένα εργασία, τότε κάποιες λειτουργίες της εφαρμογής θα επηρεάζονται από τυχόν λειτουργίες που τρέχουν στο υπόβαθρο και συνδέονται είτε με άλλες εφαρμογές, είτε με το λειτουργικό σύστημα. Για παράδειγμα, στις συγκεκριμένες περιπτώσεις συσκευών, επειδή 8 είναι όλοι οι πυρήνες του επεξεργαστή, όταν τους χρησιμοποιούμε όλους, ο επεξεργαστής μπορεί να υπερφορτώνεται και να μην καταφέρνει να έχει τελικώς καλύτερη απόδοση. Τέλος, ένας άλλος λόγος για αυτό το γεγονός είναι ότι, εκτελώντας λειτουργίες του ίδιου του μοντέλου παράλληλα η μια με την άλλη, ουσιαστικά αυτό σπάει σε πολλά κομμάτια τα οποία πρέπει να επικοινωνούν μεταξύ τους, πράγμα που μπορεί να προσθέτει καθυστέρηση, αν τα κομμάτια είναι πολλά. Όλα αυτά μας οδηγούν στο συμπέρασμα ότι δεν είναι σταθερό το ποια παράμετρος εκτέλεσης φέρνει καλύτερες επιδόσεις, αλλά η επιλογή της εξαρτάται κάθε φορά από το μοντέλο και από την συσκευή. Στη συγκεκριμένη περίπτωση, το μοντέλο έτρεχε γρηγορότερα στην CPU, και μάλιστα όχι με τον μέγιστο αριθμό νημάτων και όχι στο NNAPI.

Όσον αφορά τα μοντέλα, με μια πρώτη ματιά η διαδικασία του κβαντισμού δεν φάνηκε να επηρέασε πολύ τις επιδόσεις τους, γεγονός που μπορεί να ενθαρρύνει την μετατροπή όλο και περισσότερων μοντέλων. Στο προηγούμενο κεφάλαιο των μετρήσεων, είδαμε μια ξεκάθαρη εξάρτηση του χρόνου εκτέλεσης ενός μοντέλου και του μεγέθους του επομένως, όσο πιο μικρό μπορούμε να κάνουμε ένα μοντέλο, τόσο καλύτερα για το ποσοστό επιβάρυνσης του στη συσκευή.

7.2 Μελλοντικές Επεκτάσεις

Στην συγκεκριμένη περίπτωση, έγινε η μελέτη της εκτέλεσης των αλγορίθμων από την άποψη του χρόνου, ωστόσο δεν είναι σημαντικό το να τρέχει γρήγορα ένα μοντέλο, αν δε βγάζει τα σωστά αποτελέσματα. Στην παρούσα ανάλυση δημιουργήθηκε ουσιαστικά ένα μέσο δοκιμής διαφόρων μοντέλων Επεξεργασίας Φυσικής Γλώσσας. Έτσι σε μια πιθανή επέκταση μπορούν να ενσωματωθούν και άλλοι αλγόριθμοι προκειμένου να δοκιμαστούν ως προς τον χρόνο εκτέλεσης τους, αλλά και ως προς την απόδοση τους. Συγκεκριμένα, μπορούν να γίνουν μετρήσεις και για το ποσοστό εγκυρότητας των αποτελεσμάτων προκειμένου να αποκτηθεί μια εικόνα σχετικά με τις αποδόσεις των μοντέλων σε νέες εισόδους. Επιπλέον, μπορεί κάποιος να εκπαιδεύσει το δικό του Νευρωνικό Δίκτυο και στη συνέχεια να το ενσωματώσει στην παρούσα εφαρμογή, προκειμένου να βγάλει συμπεράσματα σχετικά με την απόδοση του σε επίπεδο χρόνου εκτέλεσης και ακρίβειας. Εκτός αυτών, μπορούν να ενσωματωθούν στην εφαρμογή και αλγόριθμοι επεξεργασίας εικόνας, έτσι ώστε η λειτουργία των

δυο ειδών αλγορίθμων να γίνεται συνδυαστικά μιας και ήδη υπάρχει υλοποιημένη η διαδικασία καταγραφής της εικόνας μιας σελίδας. Για παράδειγμα, θα μπορούσε από διάφορες εικόνες Νευρωνικά Δίκτυα Επεξεργασίας Εικόνας να παράγουν κείμενο, λεξάντες σχετικά με το κείμενο και στη συνέχεια Νευρωνικά Δίκτυα επεξεργασίας κειμένου να κάνουν ανάλυση και επεξεργασία των κειμένων αυτών.

Τέλος, η εφαρμογή αυτή μπορεί να ενσωματωθεί σε εφαρμογή που προορίζεται για κάποια χρήση με τους ίδιους ή διαφορετικούς αλγορίθμους, προκειμένου η εφαρμογή αυτή να αποκτήσει πιο έξυπνες λειτουργίες. Για παράδειγμα με βάση την κατηγοριοποίηση κειμένων μπορεί να δημιουργηθεί μια εφαρμογή όπου χρειάζεται η λειτουργία αυτή προκειμένου να λειτουργήσει καλύτερα. Όλα αυτά και πολλά άλλα μπορούν να μελετηθούν μελλοντικά. Σίγουρα όλες οι τεχνικές που αναλύθηκαν σε αυτή τη μελέτη έχουν ανοίξει νέους δρόμους στον τομέα των εφαρμογών και στο μέλλον πρόκειται να δούμε εφαρμογές με όλο και καλύτερες επιδόσεις και "έξυπνη" λειτουργία.

Παραρτήματα

Παράρτημα Α΄

Εντολές μετατροπής μοντέλων

```
# Imports

import transformers
import tensorflow as tf
from tflite_support import metadata as _metadata
```

```
# Βήμα 1

model =
transformers.TFDistilBertForQuestionAnswering.from_pretrained('distilbert-base-uncased-distilled-squad')
```

```
# Βήμα 2

input_spec = tf.TensorSpec([1, 384], tf.int32)
model._saved_model_inputs_spec = None
model._set_save_spec(input_spec)
```

```
# Βήμα 3

converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS,tf.lite.OpsSet.SELECT_TF_OPS]
converter.optimizations = [tf.lite.Optimize.DEFAULT]
```

```
# Βήμα 4

tflite_model = converter.convert()
with open('model.tflite', 'wb') as f:
    f.write(tflite_model)
```

```
# Βήμα 5

populator = _metadata.MetadataPopulator.with_model_file("model.tflite")
populator.load_associated_files(["vocab.txt"])
populator.populate()
```

Εικόνα Α΄.1: *Μετατροπή του μοντέλου DistilBERT [15]*

```
# Imports

import transformers
import tensorflow as tf
from tflite_support import metadata as _metadata
```

```
# Βήμα 1

model =
transformers.TFAutoModelForSequenceClassification.from_pretrained('cardiffnlp/twitter-roberta-base-sentiment')
```

```
# Βήμα 2

input_spec = tf.TensorSpec([1, 256], tf.int32)
model._saved_model_inputs_spec = None
model._set_save_spec(input_spec)
```

```
# Βήμα 3

converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS,tf.lite.OpsSet.SELECT_TF_OPS]
converter.optimizations = [tf.lite.Optimize.DEFAULT]
```

```
# Βήμα 4

tflite_model = converter.convert()
with open('model.tflite', 'wb') as f:
    f.write(tflite_model)
```

```
# Βήμα 5

import json
f = open('vocab.json',encoding="utf-8")
vocabulary = json.load(f)
#print(vocab)

vocab= {}

for key, value in vocabulary.items():
    vocab[value]=key

with open('vocab.txt', 'w',encoding="utf-8") as f:
    for key, value in vocabulary.items():
        #print(vocab[value])
        f.write(vocab[value] + "\n")
```

```
# Βήμα 6

populator = _metadata.MetadataPopulator.with_model_file("model.tflite")
populator.load_associated_files(["vocab.txt"])
populator.populate()
```

```
# Βήμα 7

populator = _metadata.MetadataPopulator.with_model_file("model.tflite")
populator.load_associated_files(["labels.txt"])
populator.populate()
```

Εικόνα Α΄.2: *Μετατροπή του μοντέλου RoBERTa [15]*

Βιβλιογραφία

[1] Vaani Rawatt. ***Introduction to Machine Learning***. https://medium.com/@vaani.rawatt/introduction-to-machine-learning-a5ddc31ce404, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[2] ***What is Deep learning? Real-world examples and Challenges***. https://pstechnoblog.blogspot.com/2021/08/what-is-deep-learning-real-world.html, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[3] ***Neural network models (supervised)***. https://scikit-learn.org/stable/modules/neural_networks_supervised.html, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[4] ***How to implement CNN for NLP tasks like Sentence Classification***. https://medium.com/saarthi-ai/sentence-classification-using-convolutional-neural-networks-ddad72c7048c, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[5] ***Transformers Explained Visually: Overview of Functionality***. https://towardsdatascience.com/transformers-explained-visually-part-1-overview-of-functionality-95a6dd460452, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[6] ***Guide to app architecture***. https://developer.android.com/jetpack/guide, Ημερομηνία πρόσβασης: 15 Φεβρουαρίου 2022.

[7] ***An Explanatory Guide to BERT Tokenizer***. https://www.analyticsvidhya.com/blog/2021/09/an-explanatory-guide-to-bert-tokenizer, Ημερομηνία πρόσβασης: 2 Φεβρουαρίου 2022.

[8] ***BERT Fine-Tuning Tutorial with PyTorch***. https://mccormickml.com/2019/07/22/BERT-fine-tuning/, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[9] Xiaodan Song Renjie Liu Yiming Yang Denny Zhou Zhiqing Sun, Hongkun Yu. ***MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices***. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, abs/2004.02984:2158–2170, 2020.

[10] Ott Myle Goyal Naman Du Jingfei Joshi Mandar Chen Danqi Levy Omer Lewis Mike Zettlemoyer Luke Stoyanov Liu, Yinhan. ***RoBERTa: A Robustly Optimized BERT Pretraining Approach***. *8th International Conference on Learning Representations - ICLR 2020*, abs/1907.11692, 2019.

[11] Julien Chaumond Thomas Wolf Victor Sanh, Lysandre Debut. ***DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter***. *Workshop on Energy Efficient Machine Learning and Cognitive Computing - EMC2 2019*, abs/1910.01108, 2019.

[12] Quoc V. Le Christopher D. Manning Kevin Clark, Minh-Thang Luong. ***ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators***. *8th International Conference on Learning Representations - ICLR 2020*, 2020.

[13] James Briggs. ***How to Build a WordPiece Tokenizer For BERT***. https://towardsdatascience.com/how-to-build-a-wordpiece-tokenizer-for-bert-f505d97dddbb, Ημερομηνία πρόσβασης: 28 Ιανουαρίου 2022.

[14] ***An end-to-end open source machine learning platform***. https://www.tensorflow.org, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[15] ***TensorFlow Lite converter***. https://www.tensorflow.org/lite/convert, Ημερομηνία πρόσβασης: 05 Ιανουαρίου 2022.

[16] ***Εφαρμογές στα κινητά: Γιατί είναι τόσο δημοφιλείς στους χρήστες κινητών τηλεφώνων***. https://www.appgene.net/blog/efarmoges-gia-kinita-giati-einai-dimofileis, Ημερομηνία πρόσβασης: 15 Δεκεμβρίου 2021.

[17] IBM Cloud Education. ***Machine Learning***. https://www.ibm.com/cloud/learn/machine-learning, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[18] IBM Cloud Documentation. ***Supervised Training***. https://www.ibm.com/cloud/learn/supervised-learning, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[19] Berkeley Schoolof information. ***What Is Machine Learning (ML)?*** https://ischoolonline.berkeley.edu/blog/what-is-machine-learning, Ημερομηνία πρόσβασης: 24 Δεκεμβρίου 2021.

[20] IBM Cloud Documentation. ***Unsupervised Traning***. https://www.ibm.com/cloud/learn/unsupervised-learning, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[21] Błażej Osiński Konrad Budek. ***What is reinforcement learning? The complete guide***. https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[22] ***Machine Learning - Μηχανική μάθηση - τι είναι***. https://www.csc.com.gr/machine-learning, Ημερομηνία πρόσβασης: 24 Δεκεμβρίου 2021.

[23] ***Real-Life and Business Applications of Neural Networks***. https://www.smartsheet.com/neural-network-applications, Ημερομηνία πρόσβασης: 24 Δεκεμβρίου 2021.

[24] IBM Documentation. ***The Neural Networks Model***. https://www.ibm.com/docs/en/spss-modeler/18.0.0?topic=networks-neural-model, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[25] ***AI Basics: Training vs Inference - What's the Difference?*** https://community.arm.com/arm-community-blogs/b/ai-and-ml-blog/posts/ai-basics-training-vs-inference-whats-the-difference, Ημερομηνία πρόσβασης: 2 Φεβρουαρίου 2022.

[26] ***Modern Deep Learning Techniques Applied to Natural Language Processing***. https://nlpoverview.com/#4, Ημερομηνία πρόσβασης: 18 Φεβρουαρίου 2022.

[27] Yoshua Bengio Aaron Courville Ian Goodfellow. ***Deep Learning***. *MIT Press*, 2016.

[28] Lucila Ohno Machado Wendy W Chapman Prakash M Nadkarni. ***Natural language processing: an introduction***. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328/, Ημερομηνία πρόσβασης: 24 Δεκεμβρίου 2021.

[29] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Łukasz Kaiser Illia Polosukhin Ashish Vaswani, Noam Shazeer. ***Attention Is All You Need***. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, σελίδες 5998–6008, 2017.

[30] Lucila Ohno Machado Wendy W Chapman Prakash M Nadkarni. ***Natural language processing: an introduction***. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328, Ημερομηνία πρόσβασης: 5 Φεβρουαρίου 2022, 2011.

[31] ***Your Guide to Natural Language Processing (NLP)***. https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1, Ημερομηνία πρόσβασης: 27 Ιανουαρίου 2022.

[32] Barry Haddow Alexandra Birch Rico Sennrich. ***Neural Machine Translation of Rare Words with Subword Units***. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[33] ***Summary of the tokenizers***. https://huggingface.co/docs/transformers/tokenizer_summary, Ημερομηνία πρόσβασης: 28 Ιανουαρίου 2022.

[34] ***Training, Validation, and Test Sets***. https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets#Validation_data_set, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[35] Chris McCormick Nick Ryan. ***GLUE Explained: Understanding BERT Through Benchmarks***. https://mccormickml.com/2019/11/05/GLUE, Ημερομηνία πρόσβασης: 22 Φεβρουαρίου 2022.

[36] ***SQuAD2.0: The Stanford Question Answering Dataset***. https://rajpurkar.github.io/SQuAD-explorer/, Ημερομηνία πρόσβασης: 22 Φεβρουαρίου 2022.

[37] ***IMDB Dataset of 50K Movie Reviews***. *https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews*, Ημερομηνία πρόσβασης: 22 Φεβρουαρίου 2022.

[38] Richard Zemel Ruslan Salakhutdinov Raquel Urtasun Antonio Torralba Sanja Fidler Yukun Zhu, Ryan Kiros. ***Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books***. *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, σελίδες 19–27, 2015.

[39] ***What is Java technology and why do I need it?*** *https://www.java.com/en/download/help/whatis_java.html*, Ημερομηνία πρόσβασης: 01 Ιανουαρίου 2022.

[40] ***Java Introduction***. *https://www.w3schools.com/java/java_intro.asp*, Ημερομηνία πρόσβασης: 04 Ιανουαρίου 2022.

[41] Shibaji Debnath. ***Why java for android development?*** *https://www.shibajidebnath.com/java-android-development*, Ημερομηνία πρόσβασης: 15 Δεκεμβρίου 2021.

[42] James Chen. ***Android Operating System***. *https://www.investopedia.com/terms/a/android-operating-system.asp*, Ημερομηνία πρόσβασης: 24 Δεκεμβρίου 2021.

[43] JR Raphael. ***Android versions: A living history from 1.0 to 12***. *https://www.computerworld.com/article/3235946/android-versions-a-living-history-from-1-0-to-today.html*, Ημερομηνία πρόσβασης: 30 Δεκεμβρίου 2021.

[44] ***Android Open Source Project***. *https://source.android.com*, Ημερομηνία πρόσβασης: 01 Ιανουαρίου 2022.

[45] ***ViewModel Overview***. *https://developer.android.com/topic/libraries/architecture/viewmodel*, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[46] ***Android Threading: All You Need to Know***. *https://www.toptal.com/android/android-threading-all-you-need-to-know*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[47] ***Everything you need to build on Android***. *https://developer.android.com/studio/features*, Ημερομηνία πρόσβασης: 15 Δεκεμβρίου 2021.

[48] ***General Python FAQ***. *https://docs.python.org/3/faq/general.html*, Ημερομηνία πρόσβασης: 17 Δεκεμβρίου 2021.

[49] ***Python (programming language)***. *https://en.wikipedia.org/wiki/Python_(programming_language)*, Ημερομηνία πρόσβασης: 17 Δεκεμβρίου 2021.

[50] ***About Keras***. *https://keras.io/about*, Ημερομηνία πρόσβασης: 30 Δεκεμβρίου 2021.

[51] ***What is the Jupyter Notebook?*** *https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html#notebook-document*, Ημερομηνία πρόσβασης: 28 Ιανουαρίου 2022.

[52] ***TensorFlow Lite***. *https://www.tensorflow.org/lite/guide*, Ημερομηνία πρόσβασης: 20 Δεκεμβρίου 2021.

[53] ***TensorFlow Lite and TensorFlow operator compatibility***. *https://www.tensorflow.org/lite/guide/ops_compatibility.*, Ημερομηνία πρόσβασης: 18 Φεβρουαρίου 2022.

[54] ***Model optimization***. *https://www.tensorflow.org/lite/performance/model_optimization*, Ημερομηνία πρόσβασης: 10 Ιανουαρίου 2022.

[55] ***Hugging Face: A Step Towards Democratizing NLP. It's not an emoji, it's NLP for everyone***. *https://towardsdatascience.com/hugging-face-a-step-towards-democratizing-nlp-2c79f258c951*, Ημερομηνία πρόσβασης: 10 Ιανουαρίου 2022.

[56] Arun Maiya. ***Text Classification with Hugging Face Transformers in TensorFlow 2***. *https://towardsdatascience.com/text-classification-with-hugging-face-transformers-in-tensorflow-2-without-tears-ee50e4f3e7ed*, Ημερομηνία πρόσβασης: 10 Ιανουαρίου 2022.

[57] ***Transformers***. *https://huggingface.co/docs/transformers/index*, Ημερομηνία πρόσβασης: 10 Ιανουαρίου 2022.

[58] Chang Ming Wei Lee Kenton Toutanova Kristina Devlin, Jacob. ***BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding***. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, σελίδες 4171–4186, 2019.

[59] Rani Horev. ***BERT Explained: State of the art language model for NLP***. *https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270*, Ημερομηνία πρόσβασης: 15 Ιανουαρίου 2022.

[60] Raman Kumar. ***All You Need to know about BERT***. *https://www.analyticsvidhya.com/blog/2021/05/all-you-need-to-know-about-bert*, Ημερομηνία πρόσβασης: 16 Ιανουαρίου 2022.

[61] ***Knowledge Distillation***. *https://intellabs.github.io/distiller/knowledge_distillation.html*, Ημερομηνία πρόσβασης: 20 Ιανουαρίου 2022.

[62] ***GLUE Explained: Understanding BERT Through Benchmarks***. *https://mccormickml.com/2019/11/05/GLUE/*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[63] ***F-score***. https://en.wikipedia.org/wiki/F-score, Ημερομηνία πρόσβασης: 27 Φεβρουαρίου 2022.

[64] ***Robustly optimized BERT Pretraining Approaches***. https://towardsdatascience.com/robustly-optimized-bert-pretraining-approaches-537dc66522dd, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[65] ***RoBERTa: An optimized method for pretraining self-supervised NLP systems***. https://ai.facebook.com/blog/roberta-an-optimized-method-for-pretraining-self-supervised-nlp-systems, Ημερομηνία πρόσβασης: 25 Ιανουαρίου 2022.

[66] Victor Sanh. ***Smaller, faster, cheaper, lighter: Introducing DistilBERT, a distilled version of BERT***. https://medium.com/huggingface/distilbert-8cf3380435b5, Ημερομηνία πρόσβασης: 16 Ιανουαρίου 2022.

[67] ***A Step-by-Step NLP Guide to Learn ELMo for Extracting Features from Text***. https://www.analyticsvidhya.com/blog/2019/03/learn-to-use-elmo-to-extract-features-from-text, Ημερομηνία πρόσβασης: 27 Ιανουαρίου 2022.

[68] ***Question Answering***. https://paperswithcode.com/task/question-answering, Ημερομηνία πρόσβασης: 27 Φεβρουαρίου 2022.

[69] ***Text Classification***. https://paperswithcode.com/task/text-classification, Ημερομηνία πρόσβασης: 27 Φεβρουαρίου 2022.

[70] ***BERT Question and Answer***. https://www.tensorflow.org/lite/examples/bert_qa/overview, Ημερομηνία πρόσβασης: 15 Ιανουαρίου 2022.

[71] ***DistilBERT base uncased distilled SQuAD***. https://huggingface.co/distilbert-base-uncased-distilled-squad, Ημερομηνία πρόσβασης: 5 Φεβρουαρίου 2022.

[72] ***Text classification***. https://www.tensorflow.org/lite/examples/text_classification/overview, Ημερομηνία πρόσβασης: 12 Φεβρουαρίου 2022.

[73] ***Electra***. https://huggingface.co/monologg/electra-small-finetuned-imdb/tree/main, Ημερομηνία πρόσβασης: 27 Φεβρουαρίου 2022.

[74] ***Twitter-roBERTa-base for Sentiment Analysis***. https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment, Ημερομηνία πρόσβασης: 27 Φεβρουαρίου 2022.

[75] ***Multi-Class Neural Networks: Softmax***. https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax, Ημερομηνία πρόσβασης: 10 Φεβρουαρίου 2022.

[76] ***WebViewClient***. https://developer.android.com/reference/android/webkit/WebViewClient, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[77] ***Canvas***. *https://developer.android.com/reference/android/graphics/Canvas*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[78] ***Android Image Cropper***. *https://github.com/ArthurHub/Android-Image-Cropper*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[79] ***Paint***. *https://developer.android.com/reference/android/graphics/Paint*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.

[80] ***TensorFlow Lite inference***. *https://www.tensorflow.org/lite/guide/inference*, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[81] ***TensorFlow Lite on GPU***. *https://www.tensorflow.org/lite/performance/gpu_advanced*, Ημερομηνία πρόσβασης: 20 Φεβρουαρίου 2022.

[82] ***Intent***. *https://developer.android.com/reference/android/content/Intent*, Ημερομηνία πρόσβασης: 16 Φεβρουαρίου 2022.