ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΕΛΟΠΟΝΝΗΣΟΥ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ
ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΠΑΠΑΛΕΞΑΝΔΡΟΠΟΥΛΟΣ ΓΕΩΡΓΙΟΣ

ΣΤΕΦΑΝΟΣ

ΘΕΜΑ:

”ΔΙΕΡΕΥΝΗΣΗ ΤΗΣ ΧΡΗΣΗΣ ΤΗΣ ΒΙΒΛΙΟΘΗΚΗΣ TENSORFLOW ΚΑΙ ΤΩΝ ΒΑΘΕΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΣΕ ΚΑΤΑΝΕΜΗΜΕΝΟ ΠΕΡΙΒΑΛΛΟΝ ΤΕΧΝΟΛΟΓΙΑΣ RASPBERRY”

Επιβλέπων καθηγητής: κ.Ταμπακάς Βασίλειος

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά, θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν με οποιονδήποτε τρόπο στην επιτυχή εκπόνηση αυτής της πτυχιακής εργασίας. Θα πρέπει να ευχαριστήσω θερμά τον καθηγητή μου κ.Ταμπακά Βασίλειο για την ευκαιρία που μου έδωσε να εκπονήσω αυτήν την πτυχιακή εργασία, αλλά και για την επίβλεψή του. Ήταν πάντα διαθέσιμος να μου προσφέρει τις γνώσεις, την εμπειρία και τις συμβουλές του, για θέματα μεθοδολογίας της έρευνας και για τη βαθύτερη κατανόηση του θέματος των νευρωνικών δικτύων και της εκπόνησης πάνω στην πλατφόρμα Raspberry . Εκτός των άλλων, ήταν πάντα διαθέσιμος να ασχοληθεί με κάθε απορία μου, εντός και εκτός των πλαισίων της παρούσας εργασίας. Η επίβλεψη του συνέφερε σημαντικά στην έγκυρη εκπόνηση της πτυχιακής, εντός των απαιτούμενων χρονικών πλαισίων αλλά και να τον ευχαριστήσω βαθύτατα για την αναζωπύρωση του ενδιαφέροντος που μου προκάλεσε για περαιτέρω διεύρυνση των γνώσεων μου στο μεταπτυχιακό πρόγραμμα σπουδών “ Τεχνολογίες και Υπηρεσίες Ευφυών Συστημάτων Πληροφορικής και Επικοινωνιών” του τμήματος μας. Τέλος θέλω να ευχαριστήσω θερμά την οικογένεια μου και ιδίως την αδερφή μου Γιώτα που και αυτή με την πολύτιμη βοήθεια συνείσφερε σημαντικά στην πτυχιακή αυτή εργασία.

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ:

Υπεύθυνη Δήλωση Φοιτητή Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη εργασία.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή ΠΑΠΑΛΕΞΑΝΔΡΌΠΟΥΛΟΥ ΓΕΩΡΓΙΟΥ ΣΤΈΦΑΝΟΥ που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο Πανεπιστήμιο Πελοποννήσου, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.

Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του **Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών**

Υπολογιστών Πάτρας(Πρώην Τμήμα Μηχανικών Πληροφορικής ΤΕΙ Αντιρρίου

ΔΟΜΗ ΚΑΙ ΣΤΟΧΟΙ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ:

Η διπλωματική εργασία μου αποτελείται από εννοιολογικά τμήματα. Αρχικά παρουσιάζεται η σημαντικότητα των βαθέων νευρωνικών δικτύων και ο απαραίτητος πλέον ρόλος που έχει αναλάβει η βιβλιοθήκη Tensorflow στην ανάπτυξη αυτών. Στην συνέχεια, για ακαδημαϊκούς σκοπούς και ανάγκη μελέτης των εργαλείων αυτών παρουσιάζεται το μοντέλο ανίχνευσης ψευδών ειδήσεων που αναπτύχθηκε. Χρησιμοποίησα διάφορους τρόπους και διαφορετικούς αλγόριθμους που ποικίλουν, σε συνάρτηση με τον χρόνο εκπαιδεύσεως , στην ακρίβεια των αποτελεσμάτων-ταξινομήσεων που έφεραν καθώς και στην πολυπλοκότητα της ανάπτυξης αυτών. Τέλος, σημαντικό κομμάτι λαμβάνει η κατανεμημένη πλατφόρμα Raspberry στην οποία έγινε μεταφορά και χρήση αυτών των μοντέλων για μελέτη του περιβάλλοντος αυτού και των πλεονεκτημάτων που διαθέτει.

Στόχος της διπλωματικής ήταν η μελέτη και κατανόηση σε βάθος της Τεχνητής Νοημοσύνης και των νευρωνικών δικτύων. Ο ρόλος που παίζει, πόσο σημαντική και πόσο συχνά χρησιμοποιείται η βιβλιοθήκη Tensorflow για την ανάπτυξη πολύπλοκων και απλών μοντέλων όπως σε αυτό της ανίχνευσης ψευδών ειδήσεων. Τέλος, ήθελα να παρατηρήσω την κατανεμημένη πλατφόρμα Raspberry ως προς την δυσκολία υλοποίησης ενός νευρωνικού δικτύου πάνω σε αυτή και πόσο χρήσιμη μπορεί να φανεί με το μικρό κόστος αγοράς που έχει.

THANKS GIVING:

First of all, I would like to thank all those who contributed in any way to the successful elaboration of this dissertation. I must warmly thank my professor Mr. Tampaka Vassilios for the opportunity he gave me to prepare this dissertation, but also for his supervision. He has always been available to provide me with his knowledge, experience and advice on research methodology and a deeper understanding of the subject of neural networks and work on the Raspberry Platform. Among other things, he was always available to answer any of my questions, inside and outside the context of this work. His supervision contributed significantly to the valid preparation of the dissertation, within the required time frames but also to thank him deeply for the resurgence of interest that caused me to further expand my knowledge in the postgraduate program "Information Technology and Intelligent Communication Services" of our department. Finally, I would like to warmly thank my family and especially my sister Giota who, with her valuable help, also contributed significantly to this dissertation.

STATUTORY DECLERATION:

Responsible Student Statement, I certify that I am the author of this dissertation and that any assistance I have received it is fully acknowledged and referred in the dissertation. I have also linked any sources I used and the data I collected, ideas or words, whether they are exact or paraphrased. I also certify that this dissertation was prepared by me personally specifically for this cause.

The approval of the dissertation by the Department of Electrical and Computer Engineering of the University of Peloponnese does not necessarily imply acceptance of the author’s view by the Department.

any medium, for teaching and research, free of charge and for the entire duration of the intellectual property rights. Open access to the full text for study and reading does not in any way imply endorsement of the intellectual property rights of the author / creator nor does it permit reproduction, republishing, copying, storage, sale, commercial use, transmission, distribution, publication, execution, "upload »(Downloading)," uploading ", translation, modification in any way, in part or in summary of the work, without the explicit prior written consent of the author / creator. The author / creator retains all of his moral and property rights.

I also certify that this dissertation is written by me personally, especially for the requirements of the curriculum of the Department of Electrical and Computer Engineering of Patra's (Former Department of Informatics Engineering TEI Antirrio).

STRUCTURE AND GOALS OF THE DISSERTATION:

My dissertation consists of conceptual sections. Firstly, I present the importance of deep neural networks and the much-needed role of the Tensorflow library in their development. In continue, for academic purposes and the need to study those tools, I developed a fake news detection model, while I used different ways and different algorithms, preserving their behavior in addition at the training time, the accuracy of the results- the classification they did- and to do so the complexity of their development. Finally, an important part is taken by the distributed Raspberry platform in which these models were transferred and converted so they can be used to preserve the behavior again and the advantages this platform brings

The main aim of the dissertation was the in-depth study I did for the scope of Artificial Intelligence and neural networks. Also the importance of the Tensorflow library and how often it is used to develop complex or simple models such as the fake news. Finally, I would like to preserve Raspberry distributed platform in terms of the difficulty of implementing a neural network and how useful it can be with its low purchase cost.

Λέξεις – κλειδιά: Τεχνητή νοημοσύνη, Μηχανική Μάθηση, Βαθιά Νευρωνικά Δίκτυα, Tensorflow.

Περιεχόμενα

Πίνακας Εικόνων & Σχημάτων

16. **Παραπληροφόριση από μέσα κοινωνικής δικτύωσης** (https://asia.nikkei.com/Spotlight/Asia-Insight/Asia-s-war-on-fake-news-raises-real-fears-for-free-speech)

17. **Δικτύωση Raspberry με Laptop** (https://www.weave.works/blog/kubernetes-raspberry-pi/)

18. **Διαδικασία μορφοποίησης κειμένου** (https://towardsdatascience.com/nlp-preparing-text-for-deep-learning-model-using-tensorflow2-461428138657)

19. **Σύγχρονος και ασύγχρονος τρόπος εκπαίδευσης παράλληλων δεδομένων** (http://kau.diva-portal.org/smash/get/diva2:1110319/FULLTEXT02.pdf)

1. Εισαγωγή

Εικόνα 1: Απεικόνιση προβλήματος παραπληροφόρησης (https://www.bbc.com/news/technology-52245992)

Η επιστήμη τον υπολογιστών όπως είναι ευρέως γνωστό, αποτελεί μια από τις σημαντικότερες και πολύ συζητημένες επιστήμες από την αρχή της εμφάνισης της καθώς η ανάπτυξη που έχει λάβει τα τελευταία είναι ραγδαία και η απήχηση την καθιστά πολλές φορές και απαραίτητη σε καθημερινή χρήση. Οι κλάδοι που μπορεί κάποιος να δείξει ενδιαφέρον ποικίλουν και η εμβάθυνση που μπορεί να λάβει κάθε υποτομέας του είναι αξιοσημείωτη για κάθε έναν από αυτούς.

Ένα ιδιαιτέρως σημαντικό αντικείμενο μελέτης της επιστήμης αυτής, που αναλύεται και αναπτύσσεται στην πτυχιακή αυτή είναι αυτό της Τεχνητής Νοημοσύνης. Ιδιαίτερα σημαντικό ιστορικά αποτελεί το γεγονός ότι σαν σκέψη προϋπήρχε από το 1940 αλλά η έκταση που έχει λάβει τα τελευταία 10 χρόνια(2011-2021) με την εύρεση και την δημιουργία εργαλείων καθιστά πολύ πιο εύκολη την ενασχόληση των μελετητών. Με πολύ μικρές απαιτήσεις υλικού και λογισμικού(Software and Hardware) και μερικές γραμμές από κώδικα μπορεί κάποιος να δημιουργήσει την ‘’δικιά του’’ Τεχνητή Νοημοσύνη. Ο λόγος για τον οποίο τόσα χρόνια δεν είχε αναπτυχθεί τόσο πολύ αλλά είχε παγώσει, ήταν διότι σύμφωνα με τους ερευνητές δεν υπήρχε μεγάλος όγκος δεδομένων για να μπορέσουμε να τα επεξεργαστούμε και επομένως να βγάλουμε συμπεράσματα-αποτελέσματα.

Συνεπώς λοιπόν, για να μπορέσουμε να εμβαθύνουμε, με τον όρο Τεχνητή Νοημοσύνη αναφερόμαστε στον κλάδο της πληροφορικής ο

οποίος έχει σαν κύρια ιδέα την ανάπτυξη, την σχεδίαση και την υλοποίηση υπολογιστικών συστημάτων, τα οποία θα μπορέσουν στην συνέχεια αυτόνομα να μιμηθούν την συμπεριφορά πλασμάτων της φύσης ή αντικείμενα με την Νοημοσύνη που διαθέτει ένας άνθρωπος, δηλαδή έστω και μια στοιχειώδη ευφυΐα.[36]

Πιο αναλυτικά, μια τέτοια μηχανή έχει την ικανότητα με την πάροδο του χρόνου να μαθαίνει συνεχώς πώς να συμπεριφέρεται πιο ομαλά σύμφωνα με τους στόχους που του ορίζουμε, να προσαρμόζεται στο περιβάλλον αυτό, να εξάγει συμπεράσματα, να κατανοεί από τα συμφραζόμενα και να προβλέπει κάτι στην συνέχεια, ναι επιλύει διάφορα προβλήματα κλπ. Χρησιμοποιούμε τον όρο πλάσματα της φύσης ή αντικείμενα επειδή μερικά από τα ρομπότ που έχουν αναπτυχθεί υπάρχουν με[36]:

- Την μορφή ανθρώπου,
- Την μορφή σκύλου,
- Την μορφή σκούπας-σφουγγαρίστρας,
- Την μορφή αεροσκάφους(drone)

Γενικά, πάρα πολλά αντικείμενα που χρησιμοποιούμε πλέον καθημερινά έχουν ένα τμήμα ευφυΐας (smart things) και αυτοματοποιούν συνεχώς εργασίες που έκανε ο άνθρωπος μόνος του. Σκοπός δηλαδή της ανάπτυξης τέτοιων μηχανών είναι η αυτοματοποίηση και η εξαγωγή συμπερασμάτων με δικιά τους σκέψη. Σχεδόν όλοι έχουμε ένα smart-phone που διαθέτει εικονικό βοηθό, ο οποίος με φωνητική εντολή δική μας μπορεί να μας δώσει απάντηση για κάποιο ερώτημα η να μας προτείνει μια λύση για ένα πρόβλημα(π.χ. Siri,Alexa,Bixby). Όλες αυτές οι συσκευές η τα λογισμικά αυτόνομα όπως υπάρχουν αποτελούν ανάπτυξη της ιδέας της Τεχνητής Νοημοσύνης. Σύμφωνα με τα παραπάνω λοιπόν μπορούμε να διακρίνουμε δυο κατηγορίες αυτής[37]:

- Τα λογισμικά: δηλαδή εικονικοί βοηθοί, λογισμικό ανάλυσης εικόνας, μηχανές αναζήτησης, συστήματα αναγνώρισης προσώπου και ομιλίας, συστήματα

- "Ενσωματωμένη τεχνητή νοημοσύνη": Πρόκειται δηλαδή για τον όρο Διαδίκτυο τον πραγμάτων (Internet of Things), ρομπότ, αυτόνομα αυτοκίνητα, τηλεκατευθυνόμενα αεροσκάφη (drones).

Πρόκειται επομένως για έναν συνδυασμό πολλαπλών κλάδων και επιστημών όπως της πληροφορικής, της ψυχολογίας, της νευρολογίας και της γλωσσολογίας όπως θα δούμε και στην συνέχεια, της επιστήμης μηχανικών με κύριο στόχο την σύνθεση μιας αυτόνομης συλλογιστικής μάθησης και προσαρμογής στο περιβάλλον και αυτό προσπαθεί να επιτευχθεί με δύο τρόπους. Με χρήση της συμβολικής Τεχνητής Νοημοσύνης, η οποία επιχειρεί να εξομοιώσει την ανθρώπινη νοημοσύνη αλγοριθμικά, με χρήση λογικών κανόνων υψηλού επιπέδου και την υπό συμβολική Τεχνητή Νοημοσύνη, η οποία προσπαθεί να αναπαράγει την ανθρώπινη ευφυΐα χρησιμοποιώντας στοιχειώδη αριθμητικά μοντέλα που συνθέτουν αθροιστικά έξυπνες συμπεριφορές με την ακολουθιακή αυτό οργάνωση απλούστερων δομικών συστατικών[37].

Παραδείγματα καθημερινών εμφανίσεων:

- **Διαδικτυακές αγορές και διαφήμιση.** Ένα από τα πιο διαδεδομένα αν όχι το μεγαλύτερο κομμάτι που χρησιμοποιείται ευρέως, είναι για την παροχή εξατομικευμένων συστάσεων. Δηλαδή με βάσει τις αναζητήσεις μας σε ένα ηλεκτρονικό κατάστημα ή λόγο προηγούμενων αγορών σε αυτό προτείνεται κάποιο προϊόν χωρίς την έκβαση του ανθρώπινου παράγοντα. Επίσης χρησιμοποιείται για τη βελτιστοποίηση προϊόντων, τον προγραμματισμό των αποθεμάτων, τον εφοδιαστικό τομέα...κλπ.
- **Διαδικτυακή αναζήτηση.** Οι μηχανές αναζήτησης πλέον έχουν την δυνατότητα με τον μεγάλο όγκο δεδομένων που παρέχουν οι χρήστες να παρέχουν αποτελέσματα με μεγαλύτερη ακρίβεια.
- **Αυτόματες μεταφράσεις.** Σύνηθες και ευρέως διαδεδομένο λογισμικό είναι αυτό της αυτόματης μετάφρασης και υποτιτλισμού, που μπορούν να ανταποκριθούν είτε σε γραπτό είτε σε προφορικό λόγο, χρησιμοποιούν την Τεχνητή Νοημοσύνη για την παροχή και την βελτίωση μεταφράσεων.

- **Έξυπνα σπίτια, πόλεις και υποδομές.** Έξυπνοι θερμοστάτες, έξυπνες πρίζες, λάμπες, αναλύουν την συμπεριφορά μας και κρατάνε δεδομένα με στόχο την βέλτιστη χρήση τους για εξοικονόμηση ενέργειας, για αποφυγή της κυκλοφοριακής συμφόρησης και βελτίωση της συνδεσιμότητας.
- **Αυτοκίνητα.** Πλέον έχει αργήσει και γίνεται αισθητή και η παρουσία των αυτόνομων λεωφορείων ή και οχημάτων ιχ. Όπως και τα συστήματα πλοήγησης βασίζονται στην ΤΝ.
- **Κυβερνοασφάλεια.** Συμβάλουν στην προστασία των προσωπικών δεδομένων ως προς επίθεση που μπορεί να δεχθούν από τρίτους.
- **Τεχνητή Νοημοσύνη κατά του COVID-19.** Η πανδημία που είναι το επίκεντρο των συζητήσεων τον τελευταίο χρόνο είναι και αυτό μια πολύ καλή περίπτωση να αναφερθεί, διότι με την παροχή των πληροφοριών η ΤΝ συνέβαλε στην ανακάλυψη του εμβολίου.

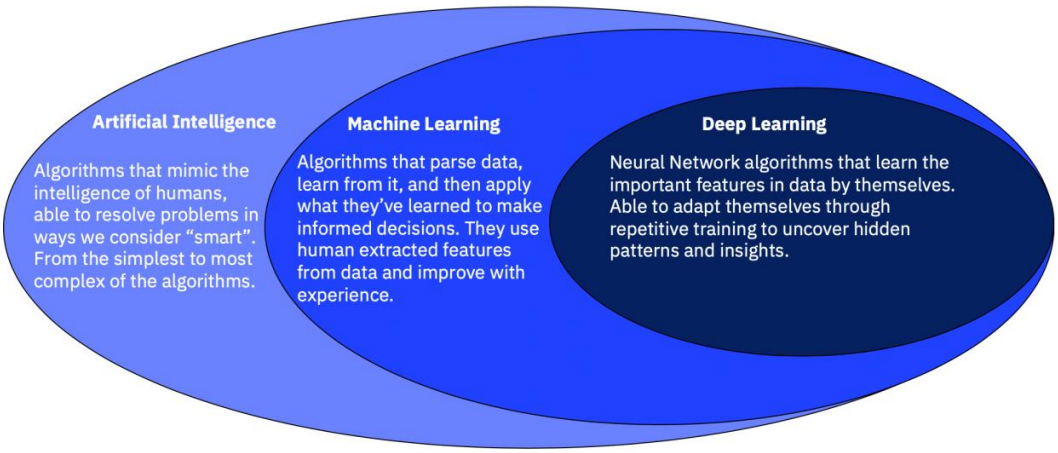
συμπεράσματα στα οποία οδηγηθήκαμε, τη σχετική βιβλιογραφία, ενώ στο τέλος παρατίθεται το παράρτημα με τους απαραίτητους πίνακες.[37]

2. Μηχανική μάθηση και ταξινόμηση κειμένου

Σε αυτό το κεφάλαιο θα αναλύσουμε ορισμένες βασικές έννοιες για την κατανόηση του θέματος της πτυχιακής αυτής. Γίνεται αναφορά στον κλάδο της Τεχνητής Νοημοσύνης και στους υποκλάδους της, Μηχανική Μάθηση και Βαθιά Μάθηση και πως σχετίζονται μεταξύ τους και μας χρησιμεύουν στην ανάπτυξη μοντέλων συστημάτων. Επίσης αναλύουμε τους 3 τύπους μάθησης που με γνώμονα τον τρόπο λειτουργίας τους κατηγοριοποιούνται τα διάφορα προβλήματα που έχουμε να επιλύσουμε. Στην συνέχεια, παραθέτουμε το πολυσύχναστο φαινόμενο της κατηγοριοποίησης κειμένου και τα διάφορα καθημερινά προβλήματα στα οποία εμφανίζεται καθώς και τον τρόπο (text classification) με τον οποίο συνήθως επιλύνονται, άλλα και την μεθοδολογία που χρησιμοποιούμε για ένα όσο τον δυνατόν βέλτιστο αποτέλεσμα.

2.1 Μηχανική Μάθηση

Όπως και στο σχήμα που θα δούμε παρακάτω οι όροι Τεχνητή Νοημοσύνη (Artificial Intelligence) -Μηχανική Μάθηση(Machine Learning)-Βαθέα Νευρωνικά Δίκτυα(Deep Neural Networks) είναι άμεσα σχετιζόμενα μεταξύ τους. Η μηχανική μάθηση είναι ένα υποπεδίο της επιστήμης των υπολογιστών που χρησιμοποιείται για την προσέγγιση προβλημάτων Τεχνητής Νοημοσύνης. Ομοίως και τα νευρωνικά δίκτυα που χρησιμοποιούμε και αναλύονται στην συνέχεια και όπως μπορούμε να καταλάβουμε μπορούμε να χρησιμοποιήσουμε ένα τέτοιο δίκτυο για να λύσουμε ένα πρόβλημα Μηχανικής Μάθησης όπως γίνεται στην πτυχιακή αυτή. Η μηχανική μάθηση διερευνά τη μελέτη καθώς και την κατασκευή αλγορίθμων που μπορούν να μαθαίνουν από τα δεδομένα και να κάνουν προβλέψεις σχετικά με αυτά.[38]



Εικόνα 2: Διαφορές των αλγορίθμων Τεχνητής Νοημοσύνης (https://www.thegreengrid.org/en/newsroom/blog/ai-machine-learning-and-deep-learning-what%E2%80%99s-difference)

Οι αλγόριθμοι αυτοί λειτουργούν κατασκευάζοντας μοντέλα από δεδομένα ανεξιχνίαστα ή πειραματικά, προκειμένου να κάνουν προβλέψεις βασισμένες στα δεδομένα εκμάθησης ή να εξάγουν αποφάσεις που εκφράζονται ως το αποτέλεσμα. Η ιδέα της μηχανικής μάθησης γεννήθηκε από την ανάγκη αναγνώρισης προτύπων και την ισχυριζόμενη θεωρία ότι οι υπολογιστές μπορούν να μάθουν από δεδομένα, με σκοπό να εκτελέσουν συγκεκριμένα καθήκοντα, χωρίς ανθρώπινη παρέμβαση, αναπτύσσοντας με αυτόν τον τρόπο, ένα είδος νοημοσύνης. Η επαναληπτική μέθοδος που ακολουθεί η μηχανική μάθηση, έχει ως απώτερο σκοπό να παράγει προβλέψεις, για νέα δεδομένα που ενδέχεται να τροφοδοτήσουν το σύστημα, δηλαδή δεδομένα που δεν έχει επεξεργαστεί. Τα μοντέλα μηχανικής μάθησης βασισμένα σε προηγούμενους υπολογισμούς και ιστορικά δεδομένα εκπαιδεύονται να παράγουν αξιόπιστες, προβλέψεις, αποφάσεις και αποτελέσματα. Είναι ένας κλάδος της επιστήμης των υπολογιστών, που αν και όπως είδαμε δεν είναι καινούργιος, βρίσκεται σε έξαρση τα τελευταία χρόνια, έχοντας σε εφαρμογή όλο και περισσότερες διαδικασίες που απαιτούσαν χρόνο και ανθρώπινη παρέμβαση. Η δυνατότητα της αυτόματης εφαρμογής σύνθετων μαθηματικών υπολογισμών, σε μεγάλου όγκου δεδομένα, σε μικρό χρονικό διάστημα και επαναληπτικά, αποτελεί εξέλιξη των τελευταίων χρόνων και οφείλεται στην πρόοδο της μηχανικής μάθησης.[39]

Οι βασικές κατηγορίες που βασίζεται η μηχανική μάθηση και ταξινομεί τις εργασίες ανάλογα με την εκπαιδευτική διαδικασία που ακολουθείται ή την <<ανατροφοδότηση>> σε ένα σύστημα είναι οι εξής:

- **Επιτηρούμενη μάθηση-επιβλεπόμενη μάθηση (supervised learning):**
  Πρόκειται όταν το σύστημα δέχεται σαν εισόδους τα δεδομένα αλλά και τα επιθυμητά αποτελέσματα, με στόχο να αφομοιώσει μέσω επεξεργασίας, τρόπους για να βρίσκει το επιθυμητό αποτέλεσμα ή έναν γενικό κανόνα προκειμένου να κάνει την αντιστοιχία. Αναφερόμαστε δηλαδή στα προβλήματα που ο στόχος είναι γνωστός και θέλουμε να καταφέρουμε μέσω των αλγορίθμων σε δεδομένα που δεν έχει συναντήσει να βρίσκει το στόχο-αποτέλεσμα.
- **Μη επιτηρούμενη μάθηση-μάθηση χωρίς επίβλεψη(unsupervised learning):**
  Περίπτωση συστήματος κατά την οποία δεν παρέχεται κάποια εμπειρία στον αλγόριθμο μάθησης και συνεπώς πρέπει από μόνος του να βρει την ομάδα που ανήκουν τα δεδομένα εισόδου. Η μάθηση αυτή επίσης συχνά αποτελεί αυτοσκοπό με στόχο την ανακάλυψη κρυμμένων μοτίβων και αλληλοσυσχετίσεων στα δεδομένα.
- **Ενισχυτική μάθηση(reinforcement learning):**
  Στην μάθηση αυτή, το σύστημα αλληλοεπιδρά με ένα δυναμικό περιβάλλον με σκοπό να επιτευχθεί ένας συγκεκριμένος στόχος, χωρίς κάποιος επιβλέπον να του επισημαίνει ρητά αν έχει φτάσει στο στόχο του ή σε κάποιο επιθυμητό αποτέλεσμα με συνέπεια να <<επιβραβεύει>> μόνος του τον εαυτό του ένας πράκτορας(agent).

[39]

2.2 Κατηγοριοποίηση Κειμένου

Η αυτοματοποιημένη κατηγοριοποίηση (ή ταξινόμηση) κειμένων σε προκαθορισμένες κατηγορίες έχει γνωρίσει ευμεγέθη άνθηση τα τελευταία περίπου 10 χρόνια , λόγω της αυξημένης διαθεσιμότητας όγκου εγγράφων σε ψηφιακή μορφή, πράγμα που αποτέλεσε επακόλουθη την ανάγκη για οργάνωση και κατηγοριοποίηση τους.

Από ερευνητική πλευρά, η προσέγγιση για επίλυση που επικρατεί σαφέστατα είναι με την χρήση μηχανικών μάθηση. Ένας ταξινομητής, δημιουργείται έπειτα από ένα σύνολο επαγωγικών διαδικασιών, εκπαιδευόμενος από ένα σύνολο προ-ταξινομημένων εγγράφων(έγγραφα που γνωρίζουμε ήδη τον στόχο τους, που πρέπει να ταξινομηθούν) και των χαρακτηριστικών που απαρτίζουν την κατηγορία αυτή. Όπως μπορούμε να συμπεράνουμε λοιπόν, οι διαδικασίες για να ταξινομηθούν αυτά τα έγγραφα γίνονταν χειρωνακτικά από ανθρώπινο δυναμικό και πρόκειται για μια αρκετά χρονοβόρα. Η μηχανική μάθηση βοήθησε ως προς την κατεύθυνση αυτή, στην εξοικονόμηση εργατικής δύναμης και χρόνου, ενώ η εφαρμοσιμότητά της σε διάφορους τομείς, την καθιστά πρώτη επιλογή, σε τέτοιας φύσης προβλήματα.[47]

Η κατηγοριοποίηση ή αλλιώς ταξινόμηση (classification) είναι μία από τις εφαρμογές μηχανικής μάθησης, κατά την οποία ένα στοιχείο κατηγοριοποιείται σε ένα σύνολο προκαθορισμένων κατηγοριών. Γενικότερα, στόχο της ταξινόμησης αποτελεί η ανάπτυξη ενός μοντέλου, το οποίο μετά την εκπαίδευση του με δεδομένα εκμάθησης, ως παραδείγματα, θα μπορεί να χρησιμοποιηθεί για την κατηγοριοποίηση μελλοντικών εισόδων στο σύστημα. Η κατηγοριοποίηση αυτή έχει χρησιμοποιηθεί σε απλά προβλήματα όπως , για τον διαχωρισμό κειμένων με βάση την θεματολογίας του, emails με βάση τον αποστολέα ή την επικεφαλίδα τους αλλά και σε πιο σημαντικά όπως την πρόβλεψη ασθενειών, την ανίχνευση καρκινικών κυττάρων χαρακτηρίζοντας τα ως καλοήθη ή κακοήθη, την κατηγοριοποίηση τον πελατών σε μια βάση δεδομένων ανάλογα με τις προτιμήσεις τους ή τις πρόσφατες αναζητήσεις τους στο διαδίκτυο κ.α.[47]

Για να υλοποιηθεί πιο εύκολα ο στόχος σε προβλήματα σαν και αυτά, είναι σημαντική η της ακολουθίας των εξής δυο σταδίων:

1. **Εκμάθηση (Learning):** Στο πρώτο αυτό στάδιο της διαδικασίας δημιουργείται ένα μοντέλο με βάση ένα σύνολο κατηγοριοποιημένων παραδειγμάτων – σύνολο δεδομένων (Dataset). Τα δεδομένα του συνόλου αυτού έπειτα χωρίζονται σε δύο κατηγορίες, στην μια στην οποία παρουσιάζονται ως δεδομένα εκπαίδευσης (Training Data) και στην άλλη σαν δεδομένα δοκιμής (Test Data). Ένας αλγόριθμος κατηγοριοποίησης, αναλύει τα δεδομένα εκμάθησης, βρίσκοντας συσχετίσεις μεταξύ των δεδομένων ή και μοτίβα που δεν γίνονταν αντιληπτά, προκειμένου να σχηματιστεί το μοντέλο. Η κατηγοριοποίηση αποτελεί μέθοδο εποπτευόμενης μάθησης (supervised learning), λόγω ότι η κατηγορία των δεδομένων εκπαίδευσης, είναι ήδη προκαθορισμένη και γνωστή για το μοντέλο αυτό. Για την αναπαράσταση του μοντέλου ταξινόμησης, χρησιμοποιούνται μαθηματικοί τύποι, σχεδιαγράμματα ή δέντρα απόφασης ή κανόνες κατηγοριοποίησης.

2. **Κατηγοριοποίησης (Classification):** Στο δεύτερο στάδιο, μετά την δημιουργία του μοντέλου, είναι η αξιολόγηση του. Εδώ χρησιμοποιούμε τα δοκιμαστικά δεδομένα που αναφέραμε νωρίτερα(Test Data), δεδομένα δηλαδή που το σύστημα δεν έχει δει. Το μοντέλο κατηγοριοποιεί τα δεδομένα αυτά και στην συνέχεια οι κατηγορίες που προβλέφθηκαν συγκρίνονται με τις κατηγορίες ή τους στόχους που είχαν καθοριστεί από τα εκπαιδευτικά δεδομένα και μας επιστρέφεται σε ποσοστό η ακρίβεια που υπήρξε. Η ακρίβεια του μοντέλου ταξινόμησης υπολογίζεται από το ποσοστό των δειγμάτων δοκιμής που κατηγοριοποιήθηκε σωστά μετά την εκπαίδευσή του με τα δεδομένα εκμάθησης. Ένα μοντέλο, που το ποσοστό επιτυχίας του ξεπερνά την τυχαία βασική επίδοση(0,5CA) γενικά κρίνεται αποδεκτό, αλλά η αποδεκτή επίδοση εξαρτάται από την φύση του προβλήματος που καλείται να επιλύσει ο ταξινομητής και ορίζεται, συνήθως, πριν ξεκινήσει η διαδικασία.[47]

2.3 Επεξεργασία Φυσικής Γλώσσας

Η Επεξεργασία Φυσικής Γλώσσας (ΕΦΓ) είναι ένας διεπιστημονικός κλάδος της επιστήμης της πληροφορικής, της τεχνητής νοημοσύνης και της υπολογιστικής γλωσσολογίας που ασχολείται με τις αλληλεπίδραση μεταξύ υπολογιστή και φυσικής γλώσσας, αλλά και υπολογιστή-ανθρώπου γενικότερα. Στόχο της Επεξεργασία Φυσικής Γλώσσας είναι η κατανόηση της φυσικής γλώσσας, δηλαδή η προσπάθεια να γίνουν ικανοί οι υπολογιστές να εξάγουν νοήματα από ανθρώπινα ή γλωσσικά δεδομένα, αλλά και η παραγωγή φυσικής γλώσσας σε συνέχεια μιας πρότασης.

Η Επεξεργασίας Φυσικής Γλώσσας πρόκειται για μια επιστήμη η οποία εμφανίστηκε για πρώτη φορά σαν ιδέα την δεκαετία του 50'. Το 1950ο Alan Turing δημοσίευσε ένα άρθρο με τίτλο "Computing Machinery and Intelligence" το οποίο πρότεινε αυτό που σήμερα ονομάζεται Turing Test ως κριτήριο νοημοσύνης, δηλαδή μια μέθοδος έρευνας που χρησιμοποιείται στην τεχνητή νοημοσύνη (AI) για να καθοριστεί αν ένας υπολογιστής είναι ικανός να σκέφτεται σαν άνθρωπος.[50]

Οι σύγχρονες προσεγγίσεις της μηχανικής μάθησης και οι αλγόριθμοι που χρησιμοποιούν για την Επεξεργασία Φυσικής Γλώσσας βασίζονται ιδιαίτερα στην στατιστική μηχανική μάθηση. Με την χρήση τους οι αλγόριθμοι αυτοί που υλοποιούνται με σκοπό την αντιμετώπιση τέτοιων προβλημάτων, στηρίζονται στην στατιστική συμπερασματολογία, δηλαδή την αυτοματοποιημένη μάθηση κανόνων μέσα από την ανάλυση μεγάλου όγκου δεδομένων από τον πραγματικό κόσμο.[50]

Ενδείκνυται πολλές διαφορετικές κατηγορίες αλγορίθμων μηχανικής μάθησης που έχουν εφαρμοστεί σε σχετικά προβλήματα με κοινό πεδίο την Επεξεργασίας Φυσικής Γλώσσας. Αυτοί οι αλγόριθμοι των διάφορων κατηγοριών λαμβάνουν ως είσοδο ένα μεγάλο σύνολο χαρακτηριστικών, με σκοπό την υλοποίηση των στατιστικών μοντέλων που αναφέραμε νωρίτερα, τα οποία με την σειρά τους λαμβάνουν πιθανολογικές αποφάσεις. Ένα από τα σημαντικότερα πλεονεκτήματα που έχουν τέτοια μοντέλα είναι ότι μπορούν να εκφράσουν την σχετική βεβαιότητα πολλών διαφορετικών πιθανών απαντήσεων και

όχι μόνο μιας, παράγοντας αρκετά υψηλό ποσοστό ορθού αποτελέσματος όταν περιλαμβάνεται ως συστατικό ενός μεγαλύτερου συστήματος.[51]

Μερικά από τα πιο συνηθισμένα πεδία έρευνας που εμφανίζεται ο κλάδος της Επεξεργασίας Φυσικής Γλώσσας είναι τα παρακάτω:

- **Ανάλυση λόγου (speech analysis):**
  Πρόκειται για ένα πεδίο που σχετίζεται με μελέτες σχετικά με την αναγνώριση της δομής του λόγου ενός κειμένου, π.χ. την φύση των σχέσεων μεταξύ δυο προτάσεων(επεξήγηση, αντίθεση κ.α.). Μια άλλη αξιοσημείωτη μελέτη στο πεδίο αυτό είναι η κατηγοριοποίηση σε ένα κομμάτι κειμένου και η δημιουργία παραφράσεων.
- **Αναγνώριση ομιλίας (speech recognition):**
  Όταν μιλάμε για αναγνώριση ομιλίας αναφερόμαστε κατευθείαν στο κομμάτι της αυτόματης μετατροπής του προφορικού ανθρώπινου λόγου σε κείμενο από τις υπολογιστικές μηχανές. Χρησιμοποιείται κυρίως σε λογισμικά κινητού κάνοντας ευκολότερη την χρήση του αλλά και σε GPS.
- **Αυτόματη απόκριση σε ερώτηση (Automatic response):**
  Πρόκειται για την αυτόματη αναζήτηση για την εύρεση της πιο βέλτιστης απάντησης σε ένα σύνολο απαντήσεων μετά από μια ερώτηση.
- **Αυτόματη περίληψη (Automate summary):**
  Η παραγωγή μιας αναγνώσιμης περίληψης ενός κειμένου. Συχνά χρησιμοποιείται για να παρέχει περιλήψεις σε κείμενα γνωστής διάταξης, όπως άρθρα στο οικονομικό μέρος μιας εφημερίδας.
- **Εξαγωγή πληροφοριών (Extract information):**
  Είναι η περίπτωση της ανάκτησης πληροφοριών από μη δομημένα δεδομένα όπως κείμενα, δεδομένα ιστοσελίδων.
- **Συντακτική ανάλυση (Syntax analysis):**
  Το πεδίο αυτό αφορά την ανάλυση του συντακτικού κορμού μιας πρότασης με σκοπό την επίλυση των οποιοδήποτε συντακτικών αμφισημιών.

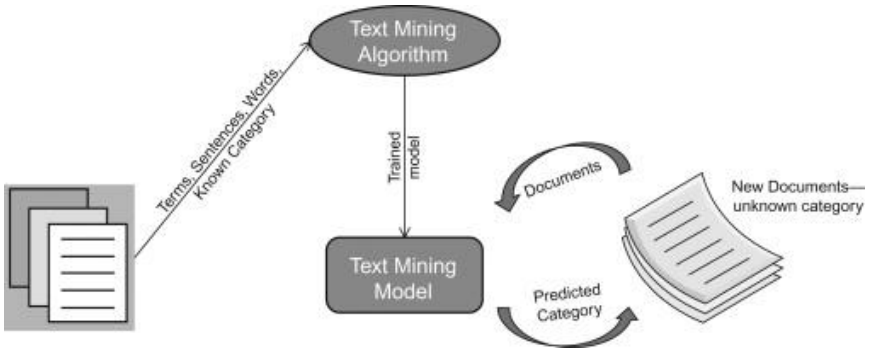
- **Εύρεση των μερών του λόγου (Searching speech parts):** Ο αυτόματος εντοπισμός των μερών του λόγου σε μία πρόταση και η επίλυση της συντακτικής αμφισημίας.

- **Παραγωγή φυσικής γλώσσας (Natural language Inference):** Η μετατροπή των πληροφοριών που εξήχθησαν σε αναγνώσιμο φυσικό λόγο, γραπτό ή προφορικό.

[51]

2.4 Μεθοδολογία Ταξινόμησης Κειμένου

Σε αυτή την ενότητα θα εξηγήσουμε τον τρόπο σχεδίασης και λειτουργίας ενός κατηγοριοποιητή κειμένου σε όλο τον κύκλο ζωής του, μέχρι την εξαγωγή των επιθυμητών αποτελεσμάτων.



Εικόνα 3: Διάγραμμα ροής Ταξινόμησης Κειμένου
(https://www.sciencedirect.com/topics/computer-science/text-mining)

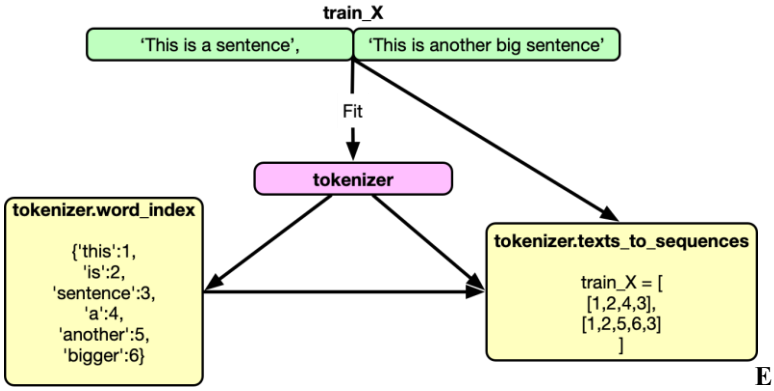
Με τον τρόπο αυτό η ταξινόμηση κειμένου, αντί να βασίζεται σε χειρωνακτική ανθρώπινη εργασία, χειροκίνητα από επεξεργασμένους κανόνες και την ανθρώπινη κρίση, με την μηχανική μάθηση μαθαίνει να κάνει ταξινομήσεις βάσει των προηγούμενων παρατηρήσεων και σαν αποτέλεσμα να γίνονται όλες οι διεργασίες αυτές αυτόματα. Χρησιμοποιώντας ως παραδείγματα κείμενα που έχουν επισημανθεί ως δεδομένα εκμάθησης, ένας αλγόριθμος μηχανικής μάθησης μπορεί να μάθει τις διαφορετικές συσχετίσεις που έχουν αναμεταξύ τους οι λέξεις, οι προτάσεις ή ολόκληρα τεμάχια κειμένων. Δηλαδή

αναμένεται για μια συγκεκριμένη είσοδο (στην περίπτωση αυτή ολόκληρο κείμενο, άρθρο ή πρόταση), μια συγκεκριμένη έξοδος όπως στο πρόβλημα που θα αναλύσουμε αν είναι παραπληροφόρηση ή πραγματικότητα. Το πρώτο βήμα που έχουμε να κάνουμε για έναν τέτοιο αλγόριθμο είναι η εξαγωγή και η επιλογή των χαρακτηριστικών που θα χρησιμοποιήσουμε από το σύνολο αυτών που μας παρέχονται. Έπειτα εφαρμόζεται μια μέθοδος ώστε να μετατρέψει κάθε κείμενο σε αριθμητική αναπαράσταση στην μορφή ενός διανύσματος. Μια από τις πιο συνηθισμένες τεχνικές προσεγγίσεις είναι η bag-of-words, όπου ένα διάνυσμα αντιπροσωπεύει τη συχνότητα μιας λέξης σε ένα προκαθορισμένο λεξικό λέξεων. Επιπλέον σημαντικό είναι να τονίσουμε πως το πλήθος των προτάσεων ή κειμένων που θα δώσουμε στο σύστημα θα πρέπει να είναι του ίδιου τύπου και του ίδιου μεγέθους.[48]

Πιο αναλυτικά, αφού μετατρέψουμε τις προτάσεις σε αριθμητικής αναπαράσταση (γεγονός που βοηθάει και στην γρήγορη επεξεργασία από τα λογισμικά) στην συνέχεια πρέπει να τους δώσουμε το κατάλληλο σχήμα. Ας φανταστούμε δυο προτάσεις, η μία μήκους 12 λέξεων και η άλλη μήκους 22, στην περίπτωση αυτή θα υπήρχε αστοχία και για τον λόγο αυτό ορίζουμε ένα συγκεκριμένο μέγεθος ορίου έτσι ώστε οι Tensors που θα παρουσιαστούν σαν είσοδοι στο σύστημα να έχουν το ίδιο μέγεθος. Αν η πρώτη πρόταση παρουσιαστεί σαν 2D Tensor (4,4) ορίζοντας σαν όρια το μήκος 16 στην πρώτη πρόταση προσθέτουμε μηδενικά(που θα αναπαριστούν κενές λέξεις) και στην δεύτερη πρόταση αφαιρούμε αντίστοιχα.[49]

Ένα άλλο σημαντικό κομμάτι που πρέπει να κατανοήσει κάποιος αν θέλει να μελετήσει την Επεξεργασία Φυσικής Γλώσσας και να αναπτύξει ένα αντίστοιχο μοντέλο είναι το εξής, εάν έχουμε ορίσει στο λεξικό μας να έχει τις ακόλουθες λέξεις {This, is, a, wonderful, day } και θέλαμε να αποκόψουμε από το κείμενο: ‘’This is a wonderful’’ θα είχαμε σαν έξοδο έπειτα από την επεξεργασία ένα διάνυσμα που θα αναπαριστούσε (1,1,0,1,0). Στη συνέχεια, ο αλγόριθμος μηχανικής μάθησης θα τροφοδοτούνταν με τα δεδομένα εκπαίδευσης που θα αποτελούνταν από ζεύγη συνόλων χαρακτηριστικών (ένα διανύσματα για κάθε παράδειγμα κειμένου) και ετικέτες (την κατηγορία που ανήκει το κείμενο π.χ. αθλήματα,

πολιτική, θετικό, αρνητικό ή κ.α.) για την παραγωγή ενός μοντέλου ταξινόμησης κειμένου.[49]



Εικόνα 4: Διάγραμμα ροής Pad Sequence (https://towardsdatascience.com/using-deep-learning-for-end-to-end-multiclass-text-classification-39b46aecac81)

3. Ταξινομητές και Βιβλιοθήκες

Σε αυτό το σημείο γίνεται αναφορά στο λειτουργικό υπόβαθρο που απαιτείται για την κατανόηση της λειτουργίας των ταξινομητών (classifiers) και πως αυτοί χρησιμεύουν στην επίλυση των προβλημάτων. Παραθέτουμε αυτούς που υλοποιήθηκαν στην συγκεκριμένη εργασία αφού το συνολικό τους πλήθος ήταν αρκετό για να καλυφθεί και επισημάνονται οι πιο συνήθεις και οι πιο αποδοτικοί για όμοια προβλήματα δυαδικής ταξινόμησης και ιδίως της κατηγοριοποίησης κειμένου. Επίσης, αναφέρονται δυο από τα σημαντικότερα νευρωνικά δίκτυα (CNN - RNN) με μια σύντομη περιγραφή του τρόπου λειτουργίας τους καθώς και εργαλεία και βιβλιοθήκες που είναι σχεδόν απαραίτητα για την υλοποίηση τους.

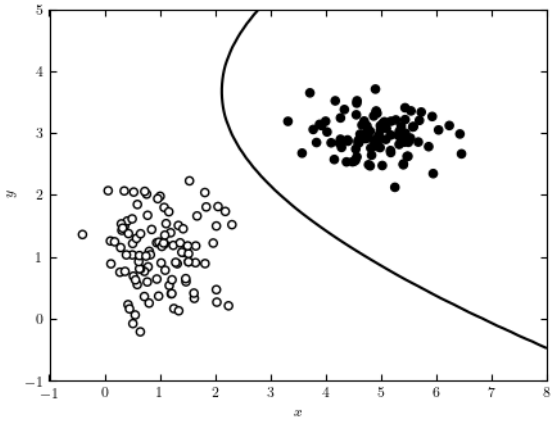
3.1 Ταξινομητές

Ένας αλγόριθμος που χρησιμοποιείται για την Κατηγοριοποίηση στοιχείων και αποτελεί κομμάτι ενός συστήματος ή μίας εφαρμογής ονομάζεται κατηγοριοποιητής. Ο όρος "κατηγοριοποιητής" συνήθως συναντάται και ως "ταξινομητής". Αρκετές φορές, ο όρος αυτός αναφέρεται και στην μαθητική συνάρτηση που χρησιμοποιεί ο αλγόριθμος για να αναλύσει τα δοκιμαστικά δεδομένα εισόδου και να τα ταξινομήσει. Στην παρούσα εργασία, οι αλγόριθμοι μηχανικής μάθησης χρησιμοποιήθηκαν για να ταξινομήσουν τα κείμενα σε δύο κατηγορίες ή κλάσεις (δυαδική ταξινόμηση-binary classification).[34]

Στην συνέχεια θα αναλύσουμε με σύντομη περιγραφή κάθε έναν αλγόριθμο κατηγοριοποίησης ξεχωριστά που χρησιμοποιήσαμε. Υπάρχουν και άλλοι που μπορεί να μας δώσουν εξίσου καλά αποτελέσματα. Αξίζει επομένως να σημειωθεί, ότι κάθε αλγόριθμος ταιριάζει σε ορισμένους τύπους προβλημάτων και δεδομένων καλύτερα και συχνά χρειάζεται η παραμετροποίηση τους για να επιτύχουν την βέλτιστη δυνατή απόδοση για ένα σύνολο δοκιμαστικών δεδομένων.[34]

Naïve Bayes

Ο Naïve Bayes υποκατηγορία του Bayesian αλγόριθμου, αποτελεί μια μέθοδο κατηγοριοποίησης βασισμένη στην στατιστική θεωρία του Bayes. Κατά την διάρκεια εφαρμογή του, πραγματοποιείται μια πιθανολογική πρόβλεψη, δηλαδή ένα στοιχείο κατατάσσεται σε μια κατηγορία, βάση της πιθανότητας του να ανήκει σε αυτή με γνώμονα τα χαρακτηριστικά που του παρέχουμε. Ο αλγόριθμος έχει εκπαιδευτή με τα δεδομένα εκμάθησης και έχει κατανοήσει-αποθηκεύσει τα χαρακτηριστικά της κάθε κλάσης. Ο Naïve Bayes θεωρεί ανεξάρτητη την επίδραση ενός χαρακτηριστικού σε μια κατηγορία, από τις τιμές των υπόλοιπων χαρακτηριστικών, με σκοπό την αποφυγή σύνθετων υπολογιστικών πράξεων. Είναι εξαιρετικά απλός, ως προς την πολυπλοκότητα του και ιδιαίτερα χρήσιμος για πολύ μεγάλα σύνολα δεδομένων. Εάν η παραδοχή της εξαρτώμενης ανεξαρτησίας του Naïve Bayes ισχύει, ένας τέτοιος Bayesian ταξινομητής θα συγκινεί ταχύτερα από τα υπόλοιπα μοντέλα και θα χρειάζεται λιγότερα δεδομένα για την εκπαίδευσή του. Το κύριο μειονέκτημα είναι ότι δεν μπορεί να μάθει αλληλεπιδράσεις μεταξύ των χαρακτηριστικών συνεπώς σε πολύπλοκα μοντέλα δεν θα ήταν και η καλύτερη επιλογή. [33]



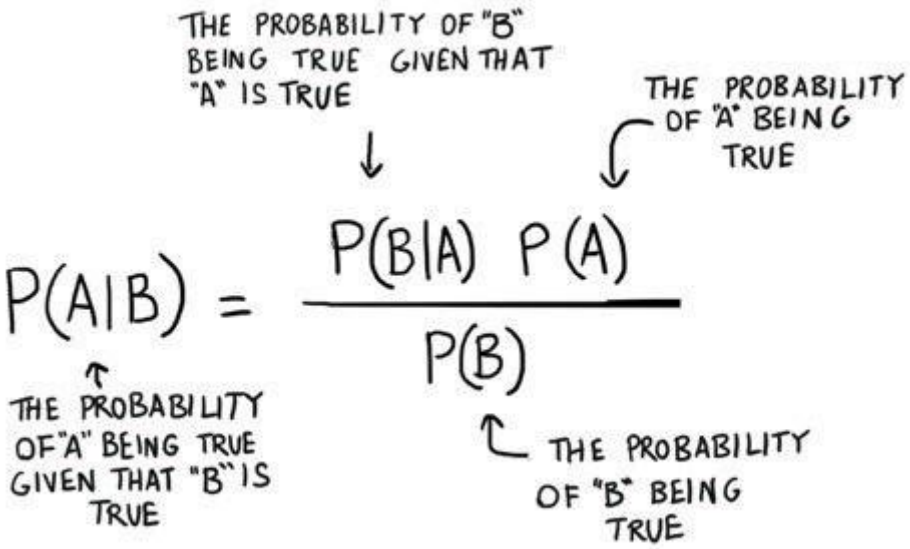
Εικόνα 5: Naïve Bayes αλγόριθμος (https://www.astroml.org/book_figures/chapter9/fig_simple_naivebayes.html)

Στην εικόνα διακρίνουμε ένα όριο αποφάσεων που υπολογίστηκε για ένα απλό σύνολο δεδομένων χρησιμοποιώντας την ταξινόμηση Gaussian Naïve Bayes. Η γραμμή δείχνει το όριο της απόφασης που έθεσε ο αλγόριθμος, το οποίο αντιστοιχεί στην καμπύλη όπου ένα νέο σημείο έχει ίση

μεταγενέστερη πιθανότητα να είναι μέρος της εκάστοτε τάξης σε μια τόσο απλή περίπτωση,

είναι δυνατό να βρεθεί μια ταξινόμηση με τέλεια πληρότητα. Αυτό δεν συμβαίνει ρεαλιστικά συχνά στον πραγματικό κόσμο.

Στην συνέχεια βλέπουμε την μαθηματική προσέγγιση του αλγορίθμου αυτού να εκφράζεται συμβολικά ως P(A|B), δηλαδή την πιθανότητα του γεγονότος Α να συμβεί σαν αποτέλεσμα δεδομένου ότι το γεγονός Β έχει ήδη συμβεί.[35]



Εικόνα 6: Naïve Bayes μαθηματική εξίσωση (https://laptrinhx.com/naive-bayes-classification-3425170402/g)

Μερικά από τα πλεονεκτήματα χρήσης του:

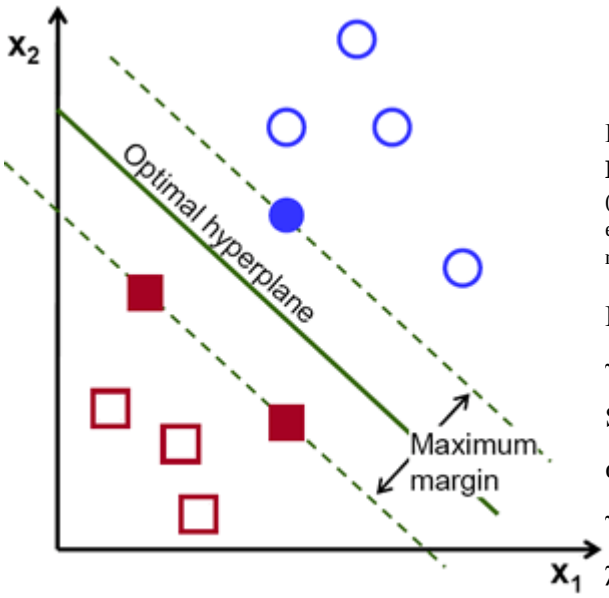
- Σε μια περίπτωση στατιστικής ταξινόμησης, ελαχιστοποιεί την πιθανότητα σφάλματος.
- Υπολογίζει ρητά τις πιθανότητες για υπόθεση και είναι ιδιαίτερα αποδοτικός ως προς τον ‘θόρυβο’, όπως χαρακτηρίζονται δεδομένα τα οποία έχουν τα χαρακτηριστικά μιας κλάσης Α αλλά ανήκουν στην κλάση Β, στα δεδομένα εισόδου.

- Παρέχει μια χρήσιμη προοπτική για την κατανόηση και την αξιολόγηση πολλών αλγορίθμων μάθησης.

[35]

Support Vector Machine

Ο αλγόριθμος κατηγοριοποίησης Μηχανών Διανυσμάτων Υποστήριξης (Support Vector Machine) είναι ένας από τους πιο συνηθισμένους που χρησιμοποιείται στην Μηχανική Μάθηση για στατιστικά προβλήματα ή προβλήματα δυαδικής ταξινόμησης αλλά είναι και αποδοτικός σε μοντέλα με πολλαπλές κλάσεις στόχων. Η ιδέα γύρω από τον αλγόριθμο αυτό σε ένα πρόβλημα βελτιστοποίησης είναι η διαδικασία με την οποία μεγιστοποιείται η απόσταση μεταξύ του ορίου απόφασης που διαχωρίζει τις κλάσεις. Χαρτογραφεί παραδείγματα εκπαίδευσης σε σημεία στο διάστημα έτσι ώστε να μεγιστοποιείται το πλάτος του κενού μεταξύ των δύο κατηγοριών. Στη συνέχεια, νέα παραδείγματα χαρτογραφούνται στον ίδιο χώρο και προβλέπεται ότι ανήκουν σε μια κατηγορία με βάση την πλευρά του κενού που πέφτουν. [32]



Εικόνα 7: Support Vector Machine
(https://randlow.github.io/posts/machine-learning/kaggle-home-loan-credit-risk-model-svm/)

Εκτός από την εκτέλεση γραμμικής ταξινόμησης, τα SVM μπορούν να εκτελέσουν αποτελεσματικά μια μη γραμμική ταξινόμηση χρησιμοποιώντας αυτό που ονομάζεται κόλπο πυρήνα, χαρτογραφώντας σιωπηρά τις εισόδους τους σε χώρους μεγάλων διαστάσεων.[32]

Λόγοι προτίμησης του αλγόριθμου:

- Λειτουργεί αρκετά καλά για ένα σύνολο δεδομένων όπου είναι σχετικά σαφές ο τρόπος διαχωρισμού μεταξύ των τάξεων
- Είναι πιο αποτελεσματικό σε χώρους μεγάλων καταστάσεων (Bid Data).
- Είναι αποτελεσματικό σε περιπτώσεις όπου ο αριθμός των διαστάσεων είναι μεγαλύτερος από τον αριθμό των δειγμάτων
- Λειτουργεί και χωρίς μεγάλη χρήση της μνήμης

Μερικοί από τους περιορισμούς που διαθέτει:

- Απαιτεί πλήρης επισήμανση των στόχων-κλάσεων των δεδομένων εισαγωγής
- Είναι δύσκολο να αναλυθούν οι παράμετροι ενός εκπαιδευμένου μοντέλου
- Μη βαθμονομημένες πιθανότητες συμμετοχής στην τάξη - Το SVM προέρχεται από τη θεωρία του Vapnik, η οποία αποφεύγει την εκτίμηση των πιθανοτήτων σε πεπερασμένα δεδομένα

[32]

Passive Aggressive Classifier

Ο αλγόριθμος Παθητικός-Επιθετικός ανήκει και αυτός στην οικογένεια των κατηγοριοποιητών αλγορίθμων μηχανικής μάθησης. Σε αυτή την περίπτωση δεν πρόκειται για έναν ευρέως γνωστό αλγόριθμο αφού αρκετοί αρχάριοι αλλά και ενδιάμεσοι ερευνητές δεν τον γνωρίζουν. Ωστόσο, είναι πολύ χρήσιμος και αποτελεσματικός για ορισμένες εφαρμογές. Χρησιμοποιείται όμως αρκετά σε προβλήματα παραπληροφόρησης αφού λειτουργεί αρκετά αποδοτικά για δεδομένα που προστίθενται συνεχώς σε πραγματικό χρόνο. Αν επεξεργαστούμε εν συνεχεία δεδομένα του διαδικτύου για τον σκοπό αυτό, αναφερόμαστε σε έναν τεράστιο όγκο δεδομένων και η χρήση ενός αλγόριθμου Online μάθησης θα ήταν ιδανική περίπτωση.[31]

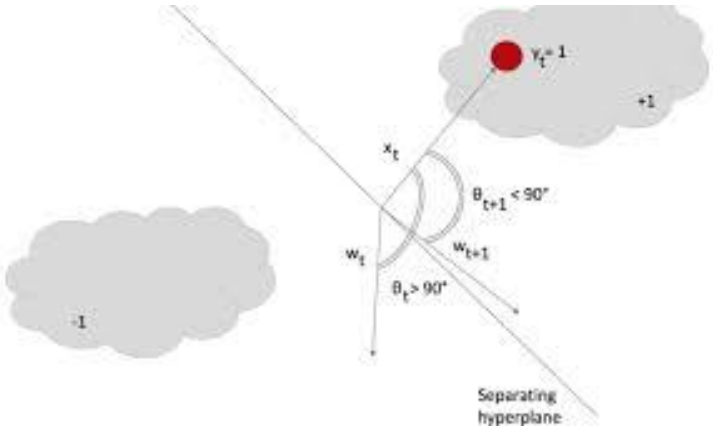
Πιο αναλυτικά, χρησιμοποιείται για μάθηση μεγάλης κλίμακας. Είναι ένας από τις λίγες περιπτώσεις <<διαδικτυακών αλγορίθμων μάθησης>>. Τα δεδομένα εισαγωγής στις περιπτώσεις αυτές έρχονται

σε διαδοχική σειρά και το μοντέλο ενημερώνεται βήμα προς βήμα, σε αντίθεση με την μαζική εκμάθηση, όπου ολόκληρο το σύνολο δεδομένων εκπαίδευσης χρησιμοποιείται ταυτόχρονα. Αυτό είναι ιδιαιτέρως χρήσιμο, σε περιπτώσεις όπως αναφέραμε μεγάλου όγκου που συνεπώς είναι υπολογιστικά ανέφικτο να εκπαιδευτεί σε ολόκληρο το σύνολο δεδομένων. Μπορούμε απλώς να πούμε ότι ένας αλγόριθμος μάθησης μέσω διαδικτύου θα λάβει ένα παράδειγμα εκπαίδευσης, θα ενημερώσει τον ταξινομητή και, στη συνέχεια, θα απορρίψει το παράδειγμά.[31]

Τον τρόπο λειτουργίας τους μπορεί να τον φανταστεί κανείς σύμφωνα με το όνομα του Παθητικός-Επιθετικός (Passive Aggressive).

- Παθητικός (Passive): Εάν η πρόβλεψη είναι σωστή, διατηρεί το μοντέλο και δεν κάνει καμία αλλαγή, δηλαδή τα δεδομένα στο εκάστοτε παράδειγμα δεν είναι αρκετά για να προκαλέσουν αλλαγές στο μοντέλο.
- Επιθετικός (Aggresive): Εάν η πρόβλεψη είναι λανθασμένη, κάνει αλλαγές στο μοντέλο, με σκοπό κάποια αλλαγή να το διορθώσει.

[31]



Εικόνα 8:Passive Aggressive Classifier (https://www.bonaccorso.eu/2017/10/06/ml-algorithms-addendum-passive-aggressive-algorithms/)

3.2 Βαθιά Νευρωνικά Δίκτυα

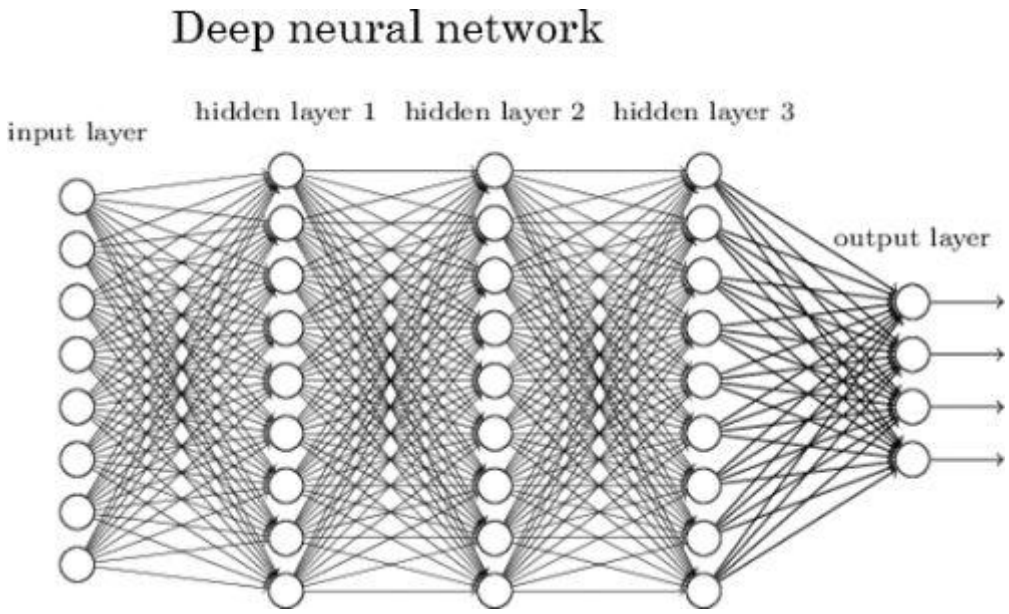
Τα βαθιά νευρωνικά δίκτυα (deep neural networks) είναι ένα σύνολο αλγορίθμων, μοντελοποιημένα και εμπνευσμένα σύμφωνα με τον τρόπο που λειτουργεί ο ανθρώπινος εγκέφαλος και έχουν σχεδιαστεί ειδικά για εκτελούν εργασίες και να αναγνωρίζουν μοτίβα. Ερμηνεύουν τα δεδομένα που συλλέχθηκαν από τους αισθητήρες μέσω ενός είδους αντίληψης μηχανής, επισήμανσης ή ομαδοποίησης ακατέργαστων εισόδων. Τα μοτίβα που αναγνωρίζουν είναι αριθμητικά, περιέχονται σε διανύσματα, στα οποία πρέπει να μεταφραστούν όλα τα δεδομένα του πραγματικού κόσμου, π.χ. εικόνες, ήχος, βίντεο, κείμενο. Είναι δίκτυα από διασυνδεδεμένα νευρωνικά υπολογιστικά στοιχεία, που έχουν την δυνατότητα να ανταποκρίνονται σε ερεθίσματα που δέχονται στην είσοδο τους, να μαθαίνουν και να προσαρμόζονται στο περιβάλλον τους.[30]

Μοιάζουν στον ανθρώπινο εγκέφαλο κυρίως στα εξής:

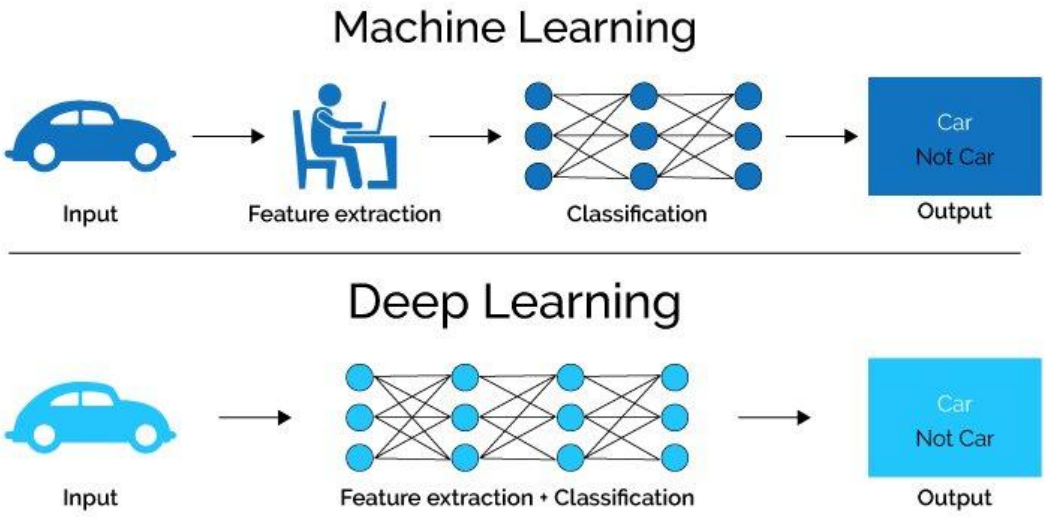
- Η γνώση αποκτάται από το δίκτυο μέσα από μια διαδικασία μάθησης-εκπαίδευσης
- Η γνώση αποθηκεύεται στις δυνάμεις σύνδεσης νευρώνων γνωστές σαν συνοπτικά βάρη.

Τα νευρωνικά δίκτυα μας βοηθούν να συγκεντρώσουμε και να ταξινομήσουμε τα δεδομένα με αποδοτικό και σχεδόν βέλτιστο τρόπο. Μπορούμε να τα θεωρήσουμε σαν ένα επίπεδο συμπλέγματος και ταξινόμησης πάνω από τα δεδομένα που αποθηκεύομε και διαχειριζόμαστε. Συμβάλλουν στην ομαδοποίηση δεδομένων που δεν έχουν κατηγοριοποιηθεί σύμφωνα με ομοιότητες μεταξύ των εισόδων του παραδείγματος και ταξινομούν τα δεδομένα όταν έχουν ένα σύνολο δεδομένων με ετικέτα για να εκπαιδευτούν. (Τα νευρικά δίκτυα μπορούν επίσης να εξαγάγουν λειτουργίες που τροφοδοτούνται σε άλλους αλγόριθμους για ομαδοποίηση και ταξινόμηση.[29]

Έτσι μπορούμε να τα σκεφτούμε ως συστατικά των μεγαλύτερων εφαρμογών μηχανικής μάθησης που περιλαμβάνουν αλγόριθμους για την ενίσχυση της μάθησης, την ταξινόμηση και την παλινδρόμηση.)



Εικόνα 9: Βαθύ Νευρωνικό Δίκτυο (https://www.kdnuggets.com/2017/05/deep-learning-big-deal.html)



Εικόνα 10: Βαθύ Νευρωνικό Δίκτυο (https://www.kdnuggets.com/2017/05/deep-learning-big-deal.html)

Η μηχανική των χαρακτηριστικών είναι το κύριο βήμα κατά την διάρκεια ανάπτυξης ενός μοντέλου με δυο βασικές υπό διαδικασίες. Την εξαγωγή χαρακτηριστικών και την επιλογή χαρακτηριστικών. Στην εξαγωγή χαρακτηριστικών, εξάγουμε όλες τις απαιτούμενες δυνατότητες για τη δήλωση προβλημάτων και στην επιλογή χαρακτηριστικών, επιλέγουμε τα σημαντικά χαρακτηριστικά που βελτιώνουν την απόδοση της μηχανικής μάθησης ή του μοντέλου

βαθιάς μάθησης. Για την εξαγωγή λειτουργιών ας θεωρήσουμε ένα πρόβλημα ταξινόμησης εικόνας, η εξαγωγή αυτή γίνεται με μη αυτόματο τρόπο από μια εικόνα χρειάζεται ισχυρή γνώση του θέματος καθώς και του τομέα. Είναι μια εξαιρετικά χρονοβόρα διαδικασία. Χάρη στη Βαθιά Εκμάθηση, μπορούμε να αυτοματοποιήσουμε τη διαδικασία του Feature Engineering.[29]

Ένα νευρωνικό δίκτυο το οποίο έχει περισσότερο από ένα κρυμμένο επίπεδο θεωρείται Βαθύ(Deep). Στην πτυχιακή αυτή θα εστιάσουμε σε δύο σημαντικούς τύπους των Βαθέων Νευρωνικών Δικτύων.[29]

- ✓ Στα Convolutional Neural Networks (CNN)
- ✓ Στα Recurrent Neural Networks (RNN)

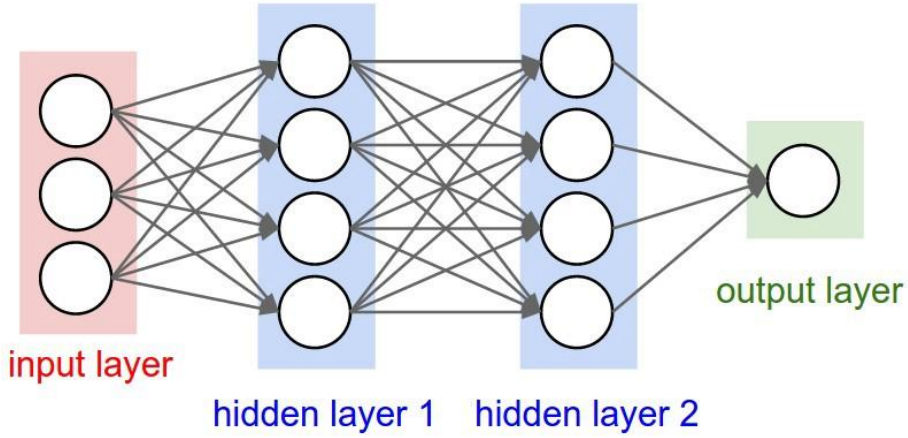
Convolutional Neural Networks (CNN)

Είναι μια παραλλαγή των νευρωνικών δικτύων που είναι ιδιαιτέρως γνωστή στην επιστήμη αυτή. Το όνομα του προέρχεται από τον τύπο που έχουν τα κρυφά επίπεδα του. Τα μεταξύ τους επίπεδα αυτά συνήθως είναι συννελικτικά (convolutional), συγκεντρωτικά (pooling), πλήρως συνδεδεμένα(fully connected) και επίπεδα κανονικοποίησης(normalization). Επίσης αντί να χρησιμοποιήσουν τις κλασσικές περιπτώσεις για συναρτήσεις ενεργοποίησης, τα συννελικτικά και συγκεντρωτικά επίπεδα χρησιμοποιούνται για τον σκοπό αυτό. [28]

Το Convolution λειτουργεί σε δύο σήματα (σε 1D) ή σε δύο εικόνες (σε 2D): μπορείτε να θεωρήσετε ένα ως το σήμα "εισόδου" (ή εικόνα) και το άλλο (που ονομάζεται πυρήνας) ως "φίλτρο" στο εικόνα εισόδου, παράγοντας μια εικόνα εξόδου (έτσι η συνέλιξη παίρνει δύο εικόνες ως εισαγωγή και παράγει μια τρίτη ως έξοδο).Με πιο απλά λόγια, παίρνει ένα σήμα εισόδου και εφαρμόζει ένα φίλτρο πάνω του, ουσιαστικά πολλαπλασιάζει το σήμα εισόδου με τον πυρήνα για να πάρει το τροποποιημένο σήμα.[28]

Η συγκέντρωση είναι μια διαδικασία διακριτοποίησης βάσει δείγματος. Ο στόχος της είναι να γίνει μια δειγματοληψία μιας αναπαράστασης εισόδου (εικόνα, πίνακας εξόδου κρυμμένου επιπέδου,

κ.λπ.), μειώνοντας τη διάστασή της και επιτρέποντας να γίνουν παραδοχές σχετικά με τα χαρακτηριστικά που περιέχονται στις υπό περιφέρειες. Υπάρχουν 2 βασικοί τύποι ομαδοποίησης που είναι κοινώς γνωστοί ως ομαδοποίηση μέγιστης και ελάχιστης ομαδοποίησης. Όπως υποδηλώνει το όνομα, η μέγιστη συγκέντρωση βασίζεται στην παραλαβή της μέγιστης τιμής από την επιλεγμένη περιοχή και η ελάχιστη συγκέντρωση βασίζεται στην παραλαβή της ελάχιστης τιμής από την επιλεγμένη περιοχή.[27]



Εικόνα 11: Δίκτυο CNN (https://www.pyimagesearch.com/2016/09/26/a-simple-neural-network-with-python-and-keras/)

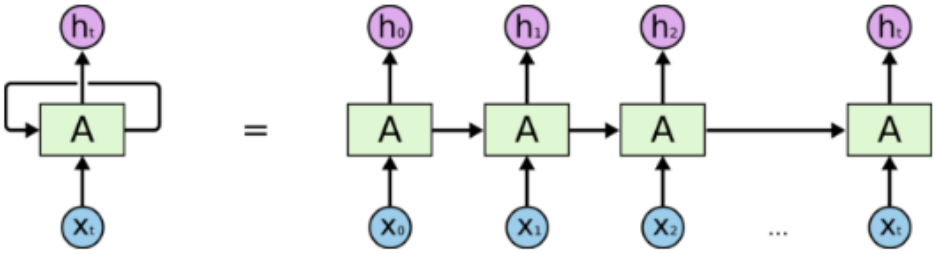
Recurrent Neural Networks (RNN)

Τα επαναλαμβανόμενα νευρωνικά δίκτυα ή RNN όπως λέγονται εν συντομία, είναι μια πολύ σημαντική παραλλαγή των νευρωνικών δικτύων που χρησιμοποιούνται πολύ στην επεξεργασία φυσικής γλώσσας. Σε ένα γενικό νευρωνικό δίκτυο, μια είσοδος υποβάλλεται σε επεξεργασία μέσω ενός αριθμού στρωμάτων και παράγεται μια έξοδος, με την υπόθεση ότι δύο διαδοχικές είσοδοι είναι ανεξάρτητες μεταξύ τους.[26]

Ωστόσο, αυτή η υπόθεση δεν ισχύει σε ορισμένα σενάρια πραγματικής ζωής. Για παράδειγμα, εάν κάποιος θέλει να προβλέψει την τιμή ενός αποθέματος σε μια δεδομένη στιγμή ή θέλει να προβλέψει την επόμενη λέξη σε μια ακολουθία, είναι επιτακτική ανάγκη να ληφθεί υπόψη η εξάρτηση από προηγούμενες παρατηρήσεις.[26]

Τα RNN ονομάζονται επαναλαμβανόμενα επειδή εκτελούν την ίδια εργασία για κάθε στοιχείο μιας ακολουθίας, με την έξοδο να

εξαρτάται από τους προηγούμενους υπολογισμούς. Ένας άλλος τρόπος να σκεφτείτε τα RNNs είναι ότι έχουν μια «μνήμη» που συλλαμβάνει πληροφορίες σχετικά με αυτό που έχει υπολογιστεί μέχρι τώρα. Θεωρητικά, τα RNN μπορούν να κάνουν χρήση πληροφοριών σε αυθαίρετα μεγάλες ακολουθίες, αλλά στην πράξη, περιορίζονται στο να κοιτάζουν πίσω μόνο μερικά βήματα.[25]



An unrolled recurrent neural network.

Εικόνα 12: Δίκτυο RNN (https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14

3.3 Python

Εικόνα 13: Python Logo (https://www.cleanpng.com/png-programming-python-logo-programming-language-compu-6805560/)

Για ανάπτυξη των συστημάτων στην παρούσα πτυχιακή χρησιμοποιήθηκε η γλώσσα Python. Πρόκειται για μια διερμηνευόμενη (interprited) γλώσσα προγραμματισμού, γενικού σκοπού (general-purpose). Είναι γλώσσα υψηλού επιπέδου, από την άποψη ότι ένα πρόγραμμα γραμμένο σε python μπορεί να μεταφερθεί εύκολα από έναν υπολογιστή, σε έναν άλλον και οι εντολές είναι απλές και κατανοητές αφού παραπέμπουν σε φυσική γλώσσα. Η φιλοσοφία

σχεδιασμού της Python δίνει έμφαση στην αναγνωσιμότητα του κώδικα με την αξιοσημείωτη χρήση σημαντικής εσοχής. Οι γλωσσικές δομές και η αντικειμενοστραφής προσέγγιση στοχεύουν να βοηθήσουν τους προγραμματιστές να γράψουν σαφή, λογικό κώδικα για μικρά και μεγάλα έργα.[1]

Θεωρείτε επίσης γλώσσα προστακτικού προγραμματισμού (Imperative programming) και υποστηρίζει τόσο τον διαδικαστικό (procedural) όσο και τον αντικειμενοστραφή προγραμματισμό (object-oriented programming). Είναι δυναμική γλώσσα (dynamically typed) παρέχοντας εύκολη χρήση δυναμικών χαρακτηριστικών και κάνοντας λιγότερους ελέγχους κατά την μετάφραση του κώδικα της στιγμή της εκτέλεσης του προγράμματος. [2]

Η Python δημιουργήθηκε στα τέλη της δεκαετίας του 1980 και κυκλοφόρησε για πρώτη φορά το 1991, από τον Guido van Rossum ως διάδοχο της γλώσσας προγραμματισμού ABC. Η έκδοση Python 2.0, που κυκλοφόρησε το 2000, εισήγαγε νέες δυνατότητες, όπως κατανοητές λιστών, και ένα σύστημα συλλογής απορριμμάτων με καταμέτρηση αναφοράς και διακόπηκε με την έκδοση 2.7 το 2020. [3]

Η επόμενη έκδοση Python 3.0, που κυκλοφόρησε το 2008, επέφερε σημαντικές αλλαγές αφού έγινε μια ουσιαστική αναθεώρηση της γλώσσας και δεν είναι εντελώς συμβατή προς τα πίσω, έτσι πολύς κώδικας Python 2 δεν εκτελείται χωρίς τροποποίηση στο Python 3. Σημαντικό γεγονός που έφερε έξαρση της φήμης και της χρήσης της γλώσσας αυτής. Με το τέλος του κύκλου ζωής της έκδοσης Python 2 (και το pip έχασε την υποστήριξη το 2021 [4]), υποστηρίζονται μόνο Python 3.6.x [5] και μεταγενέστερες εκδόσεις, με παλαιότερες εκδόσεις να υποστηρίζουν π.χ. Windows 7 (και παλιά προγράμματα εγκατάστασης που δεν περιορίζονται σε Windows 64-bit).

Οι διερμηνευτές της Python υποστηρίζονται για τα κλασσικά λειτουργικά συστήματα ενώ διατίθενται και σε λίγα επιπλέον (στο παρελθόν υποστηρίζονταν αρκετά περισσότερα). Μια παγκόσμια κοινότητα προγραμματιστών αναπτύσσει και διατηρεί το CPython, μια εφαρμογή αναφοράς δωρεάν και ανοιχτού κώδικα.[6]

3.4 Χρήσιμες Βιβλιοθήκες της Python

Scikit-Learn (sklearn)

Μία από τις πιο διαδομένες βιβλιοθήκες που υπάρχουν στο χώρο της μηχανικής μάθησης και συγκεκριμένα με την γλώσσα Python είναι η Scikit-Learn. Είναι μια βιβλιοθήκη που χρησιμοποιείται ευρέως και γίνεται αναφορά στην πτυχιακή αυτή γιατί χρησιμοποιήθηκε για την υλοποίηση τριών μοντέλων. Παρέχει αποτελεσματικές υλοποιήσεις αλγορίθμων τελευταίας τεχνολογίας, προσβάσιμους σε ειδικούς μη-μηχανικής μάθησης και επαναχρησιμοποιήσιμους σε επιστημονικούς κλάδους και τομείς εφαρμογών. [7]Εκμεταλλεύεται επίσης τη διαδραστικότητα και τη λειτουργικότητα της Python για να παρέχει γρήγορο και εύκολο πρωτότυπο.

Στην βιβλιοθήκη αυτή για χρήση των αλγορίθμων και τον αντικειμένων, γίνεται εισαγωγή των δεδομένων με την μορφή δισδιάστατων (2D) στοιχείων δειγμάτων μορφής ‘’μέγεθος x χαρακτηριστικό’’. Αυτός ο τρόπος το καθιστά γενικό και ανεξάρτητο από τον τομέα. Τα αντικείμενα μοιράζονται ένα ομοιόμορφο σύνολο μεθόδων που εξαρτώνται από τον σκοπό τους. Οι εκτιμητές (estimators) μπορούν να δώσουν στο μοντέλο τα δεδομένα, οι προβλεπτές (predictors) μπορούν να κάνουν προβλέψεις για νέα δεδομένα και οι μετασχηματιστές (transformers) να μετατρέπουν δεδομένα από τη μία αναπαράσταση στην άλλη. [8]

Numpy

Πρόκειται και εδώ για άλλη μια σημαντική βιβλιοθήκη της Python η οποία χρησιμοποιείται συστηματικά για τον διαχειρισμό για μεγάλες, πολυδιάστατες συστοιχίες (multi-dimensional arrays) και πίνακες (matrices), μαζί με μια μεγάλη συλλογή μαθηματικών συναρτήσεων υψηλού επιπέδου για λειτουργία σε αυτές τις συστοιχίες. [10] Νωρίτερα από το NumPy, η Numeric είχε δημιουργηθεί πρώτη από τον Jim Hugunin με την βοήθεια πολλών άλλων προγραμματιστών. Το 2005, ο Travis Oliphant έφτιαξε την NumPy ενσωματώνοντας χαρακτηριστικά και στοιχεία του ανταγωνιστικού Numarray στο

Numeric, με εκτεταμένες τροποποιήσεις. Το NumPy είναι λογισμικό ανοιχτού κώδικα και έχει πολλούς συντελεστές.

Παρέχει δυο βασικά αντικείμενα:

- Ένα αντικείμενο διαστάσεων N-διαστάσεων (ndarray)
- Ένα αντικείμενο καθολικής λειτουργίας (ufunc).

Ένας N-διάστατος πίνακας είναι μια ομοιογενής συλλογή ‘’αντικειμένων’’ που προσπελαύνει με την χρήση N integers. Υπάρχουν δύο βασικά στοιχεία που ορίζουν μια N-διάσταση, το σχήμα του πίνακα και το είδος του στοιχείου στο οποίο αποτελείται ο πίνακας. Το σχήμα του πίνακα είναι μια πλειάδα (Tuple) των N ακέραιων (Integers), μια για κάθε μια από τις διαστάσεις, που παρέχονται πληροφορίες σχετικά με το πόσο αναπάντεχα μπορεί να ποικίλλει το ευρετήριο κατά τη διάρκεια αυτής της διάστασης. Οι άλλες σημαντικές πληροφορίες που απαρτίζουν έναν πίνακα, είναι το είδος του στοιχείου στο οποίο αποτελείται ο πίνακας. Κάθε πίνακας N διαστάσεων είναι μια ομοιογενής συλλογή ακριβώς του ίδιου τύπου δεδομένων και μόνο, έτσι κάθε στοιχείο καταλαμβάνει το ίδιο μέγεθος σε μπλοκ μνήμης. [9]

Pandas

Μια άλλη σχεδόν ακόμα και απαραίτητη βιβλιοθήκη για την επεξεργασία κατά κύριο λόγο του συνόλου δεδομένων είναι η Pandas της Python. Βρίσκεται υπό ανάπτυξη από το 2008, με σκοπό να καλύψει το χάσμα στον πλούτο των διαθέσιμων εργαλείων ανάλυσης δεδομένων μεταξύ της Python, συστημάτων γενικής χρήσης και επιστημονικής γλώσσας υπολογιστών, και των πολυάριθμων πλατφορμών στατιστικών υπολογιστών για συγκεκριμένους τομείς και γλωσσών βάσεων δεδομένων [11].

Δεν στοχεύει μόνο στην παροχή ισοδύναμης λειτουργικότητας, αλλά και στην εφαρμογή πολλών δυνατοτήτων, όπως η αυτόματη ευθυγράμμιση δεδομένων και η ιεραρχική ευρετηρίαση, οι οποίες δεν είναι εύκολα διαθέσιμες με τόσο στενά ενσωματωμένο τρόπο σε άλλες βιβλιοθήκες ή υπολογιστικά περιβάλλοντα που γνωρίζουμε. Ενώ

αρχικά αναπτύχθηκε για εφαρμογές χρηματοοικονομικής ανάλυσης δεδομένων, προδιαθέτει την επιστημονική κοινότητα και την κάνει ένα πιο ελκυστικό και πρακτικό περιβάλλον στατιστικής πληροφορικής για ακαδημαϊκούς και επαγγελματίες του χώρου. Το όνομα της βιβλιοθήκης προέρχεται από δεδομένα πίνακα, ένας κοινός όρος για πολυδιάστατα σύνολα δεδομένων που συναντώνται στα στατιστικά και την οικονομετρία [11].

Ο προγραμματιστής McKinney ξεκίνησε να εργάζεται πάνω στην ανάπτυξη της βιβλιοθήκης αυτής το 2008, ενώ εργαζόταν στην AQR Capital Management λόγω της ανάγκης για την εύρεση ενός υψηλής απόδοσης εργαλείο για την εκτέλεση ποσοτικής ανάλυσης χρηματοοικονομικών δεδομένων. Πριν αποχωρήσει από την AQR κατάφερε να πείσει τον διευθυντή της εταιρείας να του επιτρέψει να ανοίξει την βιβλιοθήκη αυτή ως ένα πρότζεκτ ανοιχτού κώδικα. Μια άλλη υπάλληλος της AQR η Chang She, συνέβαλε στην προσπάθεια μαζί με τον McKinney ως η δεύτερη πιο σημαντική συνεισφέρουσα προγραμματίστρια στο πρότζεκτ αυτό. Το 2015 η pandas υπέγραψε συμβόλαιο σαν χρηματοδοτούμενο έργο της NumFOCUS, ενός κερδοσκοπικού οργανισμού στις Ηνωμένες Πολιτείες [12].

3.5 Tensorflow & Keras

3.5.1 Tensorflow

Η Βιβλιοθήκη Tensorflow μελετάται στην πτυχιακή αυτή, είναι μια βιβλιοθήκη της Python κατά κύριο λόγο, για γρήγορους αριθμητικούς υπολογισμούς. Δημιουργήθηκε και εκδόθηκε από την Google η οποία είναι υπεύθυνη για όλες τις αναβαθμίσεις τις και τις βελτιώσεις παρότι αποτελεί μια βιβλιοθήκη ανοιχτού κώδικα (open source library). Επίσης πρέπει να σημειωθεί ότι εκδόθηκε σύμφωνα με την άδεια του Apache 2.0 . Η διεπαφή χρονοπρογραμματισμού εφαρμογών (Application Programming Interface-API) όπως αναφέραμε είναι κατά κύριο λόγο για την γλώσσα Python αλλά μπορεί να χρησιμοποιηθεί και σε μερικές περιπτώσεις από το C++ API.[14]

Σε αντίθεση με άλλες βιβλιοθήκες που δημιουργήθηκαν για τα βαθέα νευρωνικά δίκτυα, όπως π.χ. η Theano, η Tensorflow σχεδιάστηκε για την έρευνα και την ανάπτυξη για τα συστήματα που κατασκευαζόντουσαν, το RankBrain του Google Search και το DeepDream. Σημαντικό προτέρημα που την καθιστά ναυαρχίδα, αποτελεί το γεγονός πως μπορεί να τρέξει εκμεταλλευόμενη τον επεξεργαστή CPU ενός συστήματος αλλά ιδίως της κάρτας γραφικών GPU όταν υποστηρίζει CUDA, καθώς όμως όπως θα δούμε και στην πτυχιακή αυτή χρησιμοποιείται αρκετά και σε μικρά επεξεργαστικά υπολογιστικά συστήματα, δηλαδή στο προσωπικό μας κινητό ή σε ένα Raspberry. [14]

Αξίζει να σημειωθεί, πως χρησιμοποιείται για υπολογιστικά γραφήματα (computational graphs) για να αναπαραστήσει την ροή δεδομένων (data flow) και για αριθμητικούς υπολογισμούς. Πιο αναλυτικά, λέξεις, δεδομένα ή αισθητήρες (tensors), περνάνε μέσα από αυτά τα γραφήματα (flow through the graph), γεγονός που εμπνεύστηκε και το όνομα της Tensorflow. Το γράφημα έχει κόμβους (nodes) οι οποίοι επιτρέπουν κάθε αριθμητικό υπολογισμό και συνεπώς είναι κατάλληλη για οποιαδήποτε βαθιάς μάθησης διεργασία. Όλες οι διεργασίες αυτές, η κλιμάκωση και η βελτιστοποίηση τους γίνεται σε backend επίπεδο.[15]

Αυτή η αρχιτεκτονική του Tensorflow, δίνει ευελιξία στον προγραμματιστή και τους επιτρέπει να πειραματιστούν με νέες βελτιστοποιήσεις και να εκπαιδεύσουν αλγορίθμους, ενώ άλλες βιβλιοθήκες παλιότερα παρείχαν την διαχείριση της κοινής κατάστασης ενσωματωμένη στο σύστημα. Η Tensorflow υποστηρίζει μια ποικιλία εφαρμογών, με ιδιαίτερη έμφαση στο training ενός μοντέλου και στα συμπεράσματα που προκύπτουν από τα βαθιά νευρωνικά δίκτυα. Αρκετές υπηρεσίες της Google χρησιμοποιούν το Tensorflow στην παραγωγή, έτσι και στην πτυχιακή αυτή παρουσιάζουμε την συναρπαστική απόδοση που επιτυγχάνει για πολλές πραγματικές εφαρμογές.[17]

Συμπεραίνουμε λοιπόν ότι ένα από τα μεγαλύτερα προτερήματα της Tensorflow είναι ή αφαίρεση. Αντί να ασχολείται με τις μικροσκοπικές λεπτομέρειες της εφαρμογής αλγορίθμων, ή να βρεί

κατάλληλους τρόπους για να προσαρμόσει την έξοδο μιας λειτουργίας στην είσοδο μιας άλλης, ο προγραμματιστής μπορεί να επικεντρωθεί στη συνολική λογική της εφαρμογής, γιατί φροντίζει την υλοποίηση των εφαρμογών ‘’behind the scenes’’. Επίσης προσφέρει επιπλέον ευκολίες για προγραμματιστές που πρέπει να κάνουν εντοπισμό σφαλμάτων και να αποκτήσουν ενδοσκόπηση στις εφαρμογές. Η λειτουργία πρόθυμης εκτέλεσης, μας επιτρέπει να αξιολογήσουμε και να τροποποιήσουμε κάθε λειτουργία γραφήματος ξεχωριστά και με διαφάνεια, αντί να κατασκευάζετε ολόκληρο το γράφημα ως ένα αδιαφανές αντικείμενο και να το αξιολογείτε ταυτόχρονα. Το TensorBoard Visualization μας επιτρέπει να ελέγχουμε και να σχεδιάζουμε τον τρόπο λειτουργίας των γραφημάτων μέσω ενός δια δραστικού πίνακα ελέγχου που βασίζεται στον ιστό[16]

Μέχρι στιγμής είδαμε πόσο χρήσιμη είναι η βιβλιοθήκη Tensorflow και τι πλεονεκτήματα μας προσφέρει για την κατασκευή διάφορων νευρωνικών δικτύων. Όμως είναι αρκετά δύσκολη στο να δημιουργήσει εξ’ ολοκλήρου μόνη της μοντέλα σαν και αυτά. Σε αυτό το σημείο έρχεται και τα διευκολύνει όλα η βιβλιοθήκη Keras που θα αναλύσουμε στην συνέχεια.

3.5.2 Keras

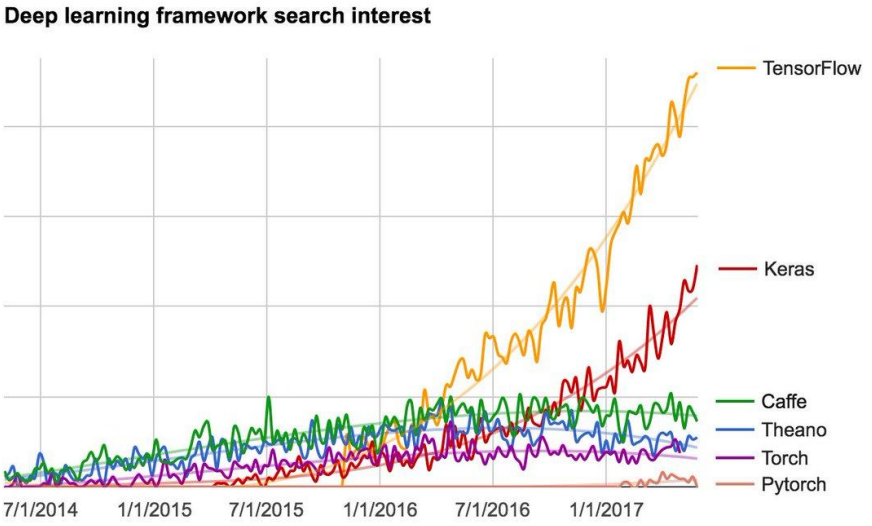
Η Keras είναι μια μινιμαλιστική βιβλιοθήκη της Python για την κατασκευή βαθέων νευρωνικών δικτύων η οποία τρέχει-διαχειρίζεται λίγο πιο low-level βιβλιοθήκες όπως η Tensorflow και η Theano. Δημιουργήθηκε με σκοπό της υλοποίηση γρήγορων, εύκολων και αξιόπιστων μοντέλων. Μπορούμε να την χρησιμοποιήσουμε και στην παλιότερα έκδοση της Python την 2 αλλά είναι ιδιαίτερα αποδοτική σε εκδόσεις 3.6 και άνω, και μπορεί και αυτή με την σειρά της όπως και η Tensorflow να τρέξει και σε GPUs και σε CPUs με δεδομένα τα απαραίτητα frameworks. Εκδόθηκε με την αδειοδότηση του MIT, και αναπτύχθηκε και διατηρείται από τον μηχανικό της Google Francois Chollet χρησιμοποιώντας τις εξής 4 κατευθυντήριες αρχές: [14]

- **Modularity:** Ένα μοντέλο μπορεί να κατανοηθεί σαν μια σειρά ή σαν ένα γράφημα ξεχωριστά. Όλες οι πτυχές των βαθέων νευρωνικών μοντέλων είναι διακριτά συστατικά τα οποία μπορούν να συγχωνευθούν με αυθαίρετους τρόπους.

- **Minimalism:** Η βιβλιοθήκη παρέχει αρκετά για να επιτευχθεί ένα αποτέλεσμα, χωρίς διακοσμητικά στοιχεία και μεγιστοποίηση της αναγνωσιμότητας.
- **Extensibility:** Νέα συστατικά είναι εκ προθέσεως εύκολα να συμπεριληφθούν και να χρησιμοποιηθούν σε ένα framework με σκοπό να δοκιμαστούν και να εξερευνηθούν από τους προγραμματιστές σε νέες ιδέες.
- **Python:** Δεν υπάρχουν ξεχωριστά μοντέλα αρχείων με συνηθισμένη μορφή. Όλα αποτελούν μέρος της Python. [14]

Επίσης έχει ενσωματωμένη υποστήριξη για επαναλαμβανόμενα δίκτυα (Convolutional Neural Networks) για computer vision, για επεξεργαστικά δίκτυα (Recurrent Neural Networks) για ακολουθιακή επεξεργασία και οποιονδήποτε συνδυασμό και των δύο. Υποστηρίζει αυθαίρετες αρχιτεκτονικές δικτύου: μοντέλα πολλαπλών εισόδων ή πολλαπλών εξόδων, κοινή χρήση επιπέδων, κοινή χρήση μοντέλων κ.α. Όλα αυτά λοιπόν σημαίνουν ότι η Keras είναι κατάλληλη για να φτιάξει οποιοδήποτε μοντέλο βαθιάς μάθησης, από ένα γενετικό δίκτυο αντιπαραθέσεων σε μια νευρική μηχανή Turing. [20]

Πληροφορικά, το Keras έχει πάνω από 200.000 χρήστες, που κυμαίνονται από ακαδημαϊκούς ερευνητές και μηχανικούς σε νεοσύστατες και σε μεγάλες εταιρείες εως και απόφοιτους ή μη φοιτητές και χομπίστες. Χρησιμοποιείται σε εταιρείες όπως η Google, Netflix, Amazon, Uber, CERN, Yelp, Square και εκατοντάδες νεοσύστατες επιχειρήσεις που λειτουργούν σε ένα ευρύ φάσμα προβλημάτων. Είναι επίσης πάρα πολύ δημοφιλές, αν όχι το πιο δημοφιλές framework, σε διαγωνισμούς μηχανικής μάθησης, όπως αυτούς που διοργανώνει το Kaggle, καθώς αποτελεί την πιο συνηθισμένη βιβλιοθήκη που κερδίζει. [20]



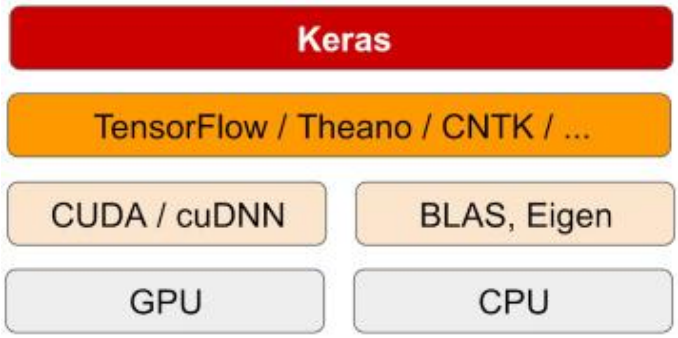
Εικόνα 14: Αναζήτηση Βιβλιοθηκών για ML και DL (https://twitter.com/fchollet/status/871089784898310144?lang=cs)

Όπως αναφέραμε διαχειρίζεται low-level βιβλιοθήκες αφού πρόκειται για μια high-level που παρέχει δομικά στοιχεία υψηλού επιπέδου. Δεν χειρίζεται όμως λειτουργίες χαμηλού επιπέδου όπως χειρισμό των tensors ή διαφοροποίηση. Αντ' αυτού, βασίζεται σε έναν εξειδικευμένο, καλά βελτιστοποιημένο tensor βιβλιοθήκη για να το κάνει, π.χ. Tensorflow, χρησιμεύοντας ως μηχανή υποστήριξης του. Αντί να χρησιμοποιήσουμε μια μόνο βιβλιοθήκη, με τον συνδυασμό αυτό των δύο, η Keras χειρίζεται το πρόβλημα με έναν αρθρωτό τρόπο και έτσι μπορούν να χρησιμοποιηθούν αρκετοί διαφορετικοί backend engines. Επί του παρόντος, οι τρεις υπάρχουσες εφαρμογές backend είναι το Tensorflow, το Theano και το Microsoft Cognitive Toolki(CNTK). Στο μέλλον, είναι πιθανό το Keras να επεκταθεί και να συνεργαστεί με περισσότερες backend εφαρμογές.[20]

Ιδιαίτερα χρήσιμο σε μια διαδικασία υλοποίησης ενός μοντέλου με την χρήση των βιβλιοθηκών αυτών αποτελεί το γεγονός πως ότι κομμάτι κώδικα γράφετε με βάση την Keras μπορεί να εκτελεστεί με οποιοδήποτε backend χωρίς να πρέπει να αλλάξουμε οτιδήποτε στο εκτελέσιμο αρχείο. Μπορούμε να αλλάξουμε απρόσκοπτα μεταξύ τους κατά της διάρκεια της ανάπτυξης, το οποίο συχνά αποδεικνύεται χρήσιμο-αν για παράδειγμα αποδειχθεί ότι κάποιο άλλο backend

frameworks είναι πιο γρήγορο από αυτό που χρησιμοποιούμε για την συγκεκριμένη εργασία. Εμείς στην εργασία αυτή προτείνουμε την επιλογή της Tensorflow ως προεπιλογή για την κάλυψη των περισσότερων αναγκών στην βαθιά-μάθηση, επειδή είναι η πιο ευρέως αποδεκτή, επεκτάσιμη και έτοιμη για παραγωγή.[20]

Ένα γράφημα που θα μας βοηθήσει να κατανοήσουμε πως λειτουργεί η Keras σε συνδυασμό με την Tensorflow είναι το παρακάτω:



Εικόνα 15: Keras workflow (https://towardsdatascience.com/my-journey-into-deeplearning-using-keras-part-1-67cbb50f65e6)

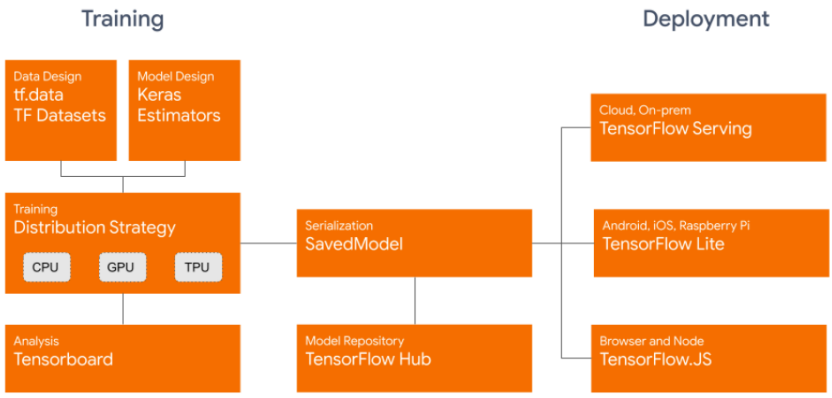
3.5.3 TensorflowLite

Κατά την διάρκεια του σχεδιασμού η Google ανέπτυξε και το TensorflowLite για να μπορεί να λειτουργεί σε ετερογενή συστήματα όπως τα κινητά και το Raspberry. Αυτό οφειλόταν στο πρόβλημα ανταλλαγής δεδομένων μεταξύ των συσκευών και των κέντρων δεδομένων , όταν η διαδικασία έπρεπε να γίνει σε μια συσκευή. Το TFL επέτρεψε στους προγραμματιστές να δημιουργήσουν διαδραστικές εφαρμογές χωρίς την ανάγκη καθυστερήσεων round-trip για υπολογισμούς ML.[21]

Καθώς οι εργασίες ML είναι υπολογιστικά ακριβές, η βελτιστοποίηση μοντέλου χρησιμοποιείται για τη βελτίωση της απόδοσης. Οι ελάχιστες απαιτήσεις υλικού του TFL όσον αφορά το μέγεθος της μνήμης τυχαίας προσπέλασης (RAM) και η ταχύτητα της CPU είναι χαμηλές και το κύριο σημείο συμφόρησης είναι η ταχύτητα υπολογισμού των υπολογισμών καθώς η επιθυμητή καθυστέρηση για

εφαρμογές για raspberry είναι χαμηλή. Για παράδειγμα, μια φορητή συσκευή με δυνατότητα υλικού λειτουργίας 10Giga Floating Point Operations Per Second (FLOPS) περιορίζεται στο μοντέλο runa 5 GFLOPS σε 2 FPS, το οποίο μπορεί να εμποδίσει την επιθυμητή απόδοση της εφαρμογής.

Το TFL είναι η εξέλιξη του TFM, το οποίο υποστηρίζει ήδη την ανάπτυξη σε κινητά και ενσωματωμένες συσκευές. Δεδομένου ότι υπάρχει μια τάση ενσωμάτωσης εφαρμογών για κινητά MLin και καθώς οι χρήστες έχουν υψηλότερες προσδοκίες για τις κινητές τους εφαρμογές όσον αφορά την κάμερα και τη φωνή, είναι πολύ ενθαρρυντικό να βελτιστοποιήσει περαιτέρω το TFM για ελαφριά χρήση κινητών [22]. Μερικές από τις βελτιστοποιήσεις που περιλαμβάνονται στο TFL είναι εξοπλισμός εξοπλισμού μέσω του στρώματος πυριτίου, πλαίσια όπως το Android NeuralNetwork API και ANN βελτιστοποιημένα για κινητά όπως MobileNets [22] και SqueezeNet [23]. Τα εκπαιδευμένα μοντέλα TF μετατρέπονται αυτόματα σε μορφή TFLmodel με TF [24].



Εικόνα 16: Tensorflowlite workflow stack (https://www.admin-magazine.com/Articles/The-TensorFlow-AI-framework/(offset)/6)

Βασικά χαρακτηριστικά και πλεονεκτήματα ενός μοντέλου Deep Learning σε Edge συσκευή:

- **Ελαφρύ (Light-weight):** Οι συσκευές Edge έχουν περιορισμένους πόρους όσον αφορά την ικανότητα αποθήκευσης και υπολογισμού. Τα μοντέλα βαθιάς μάθησης είναι πλούσια σε πόρους, επομένως τα μοντέλα που αναπτύσσουμε σε συσκευές edge θα πρέπει να είναι ελαφριά με μικρότερα binary size.
- **Χαμηλός λανθάνων χρόνος (Low Latency):** Τα μοντέλα στις συσκευές αυτές πρέπει να παράγουν γρηγορότερα συμπεράσματα ανεξάρτητα από τη συνδεσιμότητα του δικτύου. Καθώς τα συμπεράσματα γίνονται στην συσκευή, ένα round trip από την συσκευή στον διακομιστή θα εξαλειφθεί, καθιστώντας τα συμπεράσματα ταχύτερα.
- **Ασφαλές (Secure):** Από την στιγμή που το μοντέλο αναπτύσσεται στην συσκευή αυτή, τα συμπεράσματα εξάγονται σε αυτή, κανένα στοιχείο δεν φεύγει από την συσκευή ή κοινοποιείται σε ολόκληρο το διαδίκτυο, την καθιστά απόλυτα ασφαλής και δεν τίθεται περαιτέρω ανησυχία για το απόρρητο των δεδομένων.
- **Βέλτιστη κατανάλωση ενέργειας (Optimal Power):** Το δίκτυο χρειάζεται υψηλή ισχύ, οι συσκευές όμως δεν είναι συνδεδεμένες σε αυτό, επομένως η κατανάλωση ενέργειας είναι αρκετά χαμηλή.
- **Προ εκπαιδευμένο (Pre-Trained):** Τα μοντέλα μπορούν να εκπαιδευτούν on-prem ή cloud για διαφορετικές εργασίες βαθιάς μάθησης όπως ταξινόμηση εικόνας, ανίχνευση αντικειμένων, αναγνώριση ομιλίας κ.λπ. και μπορούν εύκολα να αναπτυχθούν για να εξάγουν συμπεράσματα.

[40]

4. Fake news Detection Model & Dataset

Είναι απαραίτητο να διευκρινίσουμε τι πρόβλημα αναλύουμε, τι κινδύνους αντιμετωπίζουμε καθημερινά, τι επίπτωση έχει στην κοινωνία μας, γιατί έχει λάβει τόσο μεγάλη έκταση αλλά και να τονίσουμε την ριζική αύξηση που έχει λάβει τα τελευταία χρόνια το πρόβλημα της παραπληροφόρησης. Αναφέρουμε την ιδέα που αναπτύσσουμε για την επίλυση του προβλήματος αυτού, τα μοντέλα που έχουν ήδη αναπτυχθεί και τα αποτελέσματα που έχουν επιφέρει αυτά με την σειρά τους. Δείχνουμε επίσης όλη την ροή της διαδικασίας για τα δεδομένα που συλλέξαμε και πως τα επεξεργαστήκαμε σε πρώτο στάδιο ώστε να μπορέσουμε να τα χρησιμοποιήσουμε για τη ανάπτυξη των μοντέλων μας.

4.1 Το πρόβλημα της παραπληροφόρησης

Σε αυτό το σημείο, αξίζει να σημειωθεί πως καθημερινά μπορούμε να πέσουμε ‘θύμα’ μιας απάτης ψεύτικων ειδήσεων καθώς οι διάφορες ιστοσελίδες στο διαδίκτυο μπορούν να δημοσιοποιούν ανεξέλεγκτα και χωρίς κανένα έλεγχο ότι άρθρο συντάσσουν. Αυτό αποτέλεσε ένα σημαντικό λόγο για την εύρεση μια λύσης σε ένα από τα πιο διαδεδομένα προβλήματα. Η ανάπτυξη του Fake News detection model που παρουσιάζεται σε αυτή την πτυχιακή μας αντικατοπτρίζει πόσο ορθά θα λειτουργούσε αν τον εκμεταλλευόμασταν για να φιλτράρουμε και για να αποτρέπουμε τέτοιες ιστοσελίδες καθημερινά.

Εικόνα 17: Παραπληροφόρηση από μέσα κοινωνικής δικτύωσης (https://asia.nikkei.com/Spotlight/Asia-Insight/Asia-s-war-on-fake-news-raises-real-fears-for-free-speech)

Στο πλαίσιο αυτό η παραπληροφόρηση ορίζεται ως ψευδείς, ανακριβείς ή ότι περιέχει παραπλανητικές πληροφορίες που

κατασκευάζονται, παρουσιάζονται και προωθούνται με σκοπό το κέρδος ή για να ζημιώσουν το κοινό συμφέρον.

Οι πληροφορίες αυτές μπορούν να υπονομεύσουν τις δημοκρατικές διαδικασίες και αξίες και να θέσουν στο στόχαστρο διάφορους τομείς, όπως η υγεία, η επιστήμη, η εκπαίδευση και ο χρηματοπιστωτικός τομέας. Τονίζεται η ανάγκη συμβολής όλων των ενδιαφερόμενων στοιχείων για κάθε πιθανή μέθοδο και συνιστά κατά κύριο λόγο την δημιουργία μιας προσέγγισης αυτορρύθμισης.[41]

Οι σύγχρονες προσεγγίσεις βασίζονται στον κλάδο του Machine Learning και των αλγορίθμων που χρησιμοποιούνται σε αυτόν με ποικίλα επιλογή χαρακτηριστικών για να αυξήσουν την επιτυχία στο αυτοματοποιημένο φιλτράρισμά τους. Πολλές φορές όμως δεν αποτελεί και τον πιο αποδοτικό τρόπο διότι έπονται περιορισμοί σχετικά με τον εντοπισμό πλαστών ειδήσεων νωρίς, γιατί οι πληροφορίες που απαιτούνται συχνά δεν είναι διαθέσιμες ή είναι ανακριβείς στο πρώιμο στάδιο διάδοσης των ειδήσεων αυτών. Ως αποτέλεσμα η ακρίβεια που επιτυγχάνουν σε συνάρτηση με την έγκαιρη ανίχνευση τους είναι αρκετά χαμηλή.[41]

Συνοψίζοντας, επισημαίνουμε ότι σύμφωνα με μια πρόσφατη μελέτη της Google, 2210 Αγγλικά άρθρα τα οποία χαρακτηρίστηκαν ως παραπληροφόρηση εμφανίστηκαν από την Γενάρη του 2017, σε σύγκριση με μόλις 73 εως το 2016. Όχι μόνο το πρόβλημα αυτό έχει εκτοξευθεί τα τελευταία 4 χρόνια αλλά έχει ξεπεράσει κατά πολύ σε σύγκριση με την παραπληροφόρηση της τηλεόρασης που καταμετρήθηκαν 329 άρθρα μέχρι το αντίστοιχο έτος. Ενώ στο διαδίκτυο καταμετρήθηκαν 1623 άρθρα όπου καταμετρήθηκαν από το Facebook, το Twitter και όλα τα υπόλοιπα social media, περιορίζοντας ρητά τις μελέτες για παραπληροφόρηση γύρω από μια συγκεκριμένη πλατφόρμα.[42]

4.2 Ιδέα για την επίλυση του προβλήματος

Η ιδέα λοιπόν που παρουσιάζεται σε αυτή την πτυχιακή για την έγκαιρη εύρεση τέτοιων ειδήσεων και συνεπώς αντιμετώπιση μιας πτυχής του προβλήματος είναι, αρχικά αν μπορούμε και υπάρχει η δυνατότητα να εκμεταλλευτούμε μια τέτοια πλατφόρμα για ακαδημαϊκούς σκοπούς εκμάθησης των νευρωνικών δικτύων. Πόσο εύκολη είναι στην χρήση της, ποια προβλήματα θα συναντήσουμε, γρήγορους και εύκολους τρόπους επίλυσης καθώς και αν θα μπορούσαμε από την στιγμή που το κόστος είναι αρκετά μικρό να την υλοποιήσουμε για να εκτελεί έναν τόσο συγκεκριμένο σκοπό, δηλαδή να ανιχνεύει αφού έχει εκπαιδευτεί όποια ιστοσελίδα ανοίγουμε η όποιο tweet-δημοσίευση που γίνεται στα social media post και να μας προειδοποιεί την πιθανότητα να είναι παραπληροφόρηση. Επίσης γίνεται σύγκριση από πλευράς hardware-κόστους ενός Raspberry και ενός μέσου Laptop επεξεργαστή αν συμφέρει για μικρά τέτοια task καθώς και θεωρητικά αν γινόντουσαν ένα δίκτυο πολλαπλών μικροσυσκευών αν θα μειωνόταν αισθητά ο χρόνος εκπαιδεύσεως τους.

Εικόνα 18: Δικτύωση Raspberry με Laptop
(https://www.weave.works/blog/kubernetes-raspberry-pi/)

4.3 Μοντέλο παραπληροφόρησης

Σε εννοιολογικό επίπεδο, τα ψεύτικα νέα έχουν ταξινομηθεί σε διαφορετικούς τύπους. Η γνώση επεκτείνεται για να γενικεύσει τα μοντέλα μηχανικής μάθησης (ML) για πολλούς τομείς. Διάφορες μελέτες περιλαμβάνουν την εξαγωγή γλωσσικών χαρακτηριστικών όπως n-grams από άρθρα κειμένου και εκπαίδευση πολλαπλών μοντέλων ML, όπως K-nearest neighbor (KNN), Support Vector Machine (SVM), Logistic Regression (LR), Lagnarian Support Vector Machine (LSVM), Decision tree (DT) και Stochastic Gradient Descent (SGD). Σύμφωνα με την έρευνα, καθώς ο αριθμός των αυξημένων in-gram υπολογίστηκε για ένα συγκεκριμένο άρθρο, η συνολική ακρίβεια μειώθηκε. Το φαινόμενο έχει παρατηρηθεί για μοντέλα μάθησης που χρησιμοποιούνται για ταξινόμηση. [42]

Καθώς λοιπόν οι μελέτες γύρω από πρόβλημα αυτό ποικίλουν πολλοί έχουν δώσει αποτελέσματα με τους απλούς αλγόριθμους ταξινόμησης σε σύντομο χρονικό διάστημα και με λίγες γραμμές κώδικα. Αφού παρουσιάσουμε και βγάλουμε τα αποτελέσματα από τους αλγόριθμους αυτούς και εμείς με την σειρά μας, στην συνέχεια αναπτύσσουμε ένα CNN και ένα RNN μοντέλο για να μελετήσουμε πόσο δύσκολη θα ήταν η ανάπτυξη τους, πόσο μεγαλύτερες απαιτήσεις έχουν από πλευράς Hardware για να εξάγουν γρήγορα αποτελέσματα και τι ενδείκνυται σε μια πλατφόρμα μικρή σαν το Raspberry. Διευρύνουμε τις δυνατότητες της χρήσης της βιβλιοθήκης Tensorflow και κάνουμε αναφορά στην Lite έκδοση που παρέχει η ίδια(TensorflowLite) και δείχνουμε με ποιον τρόπο μπορούμε να μεταφέρουμε τα μοντέλα αυτά που αναπτύξαμε σε μια συσκευή σαν το Raspberry η οποία λόγο του περιορισμένου δυνατοτήτων hardware ορισμένα εργαλεία δεν μας δίνει την δυνατότητα να τα χρησιμοποιήσουμε. Εφόσον γίνει η μετατροπή, στην συνέχεια παρουσιάζουμε τον τρόπο με τον οποίο ‘’ξαναχρησιμοποιείται’’ ένα τέτοιο μοντέλο στην lite πλέον έκδοσή του.

Όπως θα αναλύσουμε και στην συνέχεια η ανάπτυξη διαφέρει από πρόβλημα σε πρόβλημα γιατί τον σημαντικότερο ρόλο παίζει η επιλογή των χαρακτηριστικών. Τον τρόπο με τον οποίο θα παρουσιαστούν τα δεδομένα σε ένα νευρωνικό δίκτυο είναι σχετικά όμοια γιατί συνήθως

είναι με την μορφή πινάκων(n-d Tensors) με αριθμητικές τιμές για την γρηγορότερο εξαγωγή αποτελεσμάτων.

4.4 Εξόρυξη Δεδομένων

Το πρώτο πράγμα που θα απασχολήσει κάποιων που θα θελήσει να ασχοληθεί με την ανάπτυξη νευρωνικού δικτύου η γενικά ενός προβλήματος Machine Learning είναι ο χώρος καταστάσεων, ο όγκος και το είδος δεδομένων που θα χρησιμοποιήσει. Στην πτυχιακή αυτή και για την ανάπτυξη ενός μοντέλου Text Classification έγινε συνδυασμός 3 διαφορετικών dataset που βρέθηκαν στο διαδίκτυο τα οποία είχαν παρόμοια δομή μεταξύ τους. Στο τέλος δημιουργήθηκε ένα ενιαίο οπού περιέχει 3 στήλες με τον τίτλο κάθε άρθρου, το κείμενο του και για το αν πρόκειται για αληθινή ή ψευδής είδηση. [43]

Εγκυκλοπαιδικά σαν εξόρυξη δεδομένων(Data Mining) ορίζουμε την διαδικασία μη προφανούς εξαγωγής πληροφορίας από μεγάλες βάσεις δεδομένων, η οποία μπορεί να φανεί χρήσιμη για εξαγωγή συμπερασμάτων. Το επιστημονικό υποπεδίο της εξόρυξης δεδομένων βασίζεται σε προ υπάρχουσες έννοιες και επιστημονικές περιοχές, όπως η μηχανική μάθηση και οι βάσεις δεδομένων. Η εφαρμογή της, δεν είναι πάντοτε εφικτή σε μεγάλου όγκου και περίπλοκα δεδομένα, λόγω του ότι τέτοιοι μέθοδοι θα ήταν εξαιρετικά χρονοβόροι, όσο και επειδή ο αναλυτής δεν είναι σε θέση να καθοδηγήσει τη διαδικασία. Ένας ακόμη περιορισμός μπορεί να είναι οι νομικές διαδικασίες που προστατεύουν τα δεδομένα ή η κρυπτογράφηση τους. Η εξόρυξη δεδομένων παρουσιάζει ομοιότητες αλλά και διαφορές με την μηχανική μάθηση, ως προς τους στόχους, οι οποίες παρουσιάζονται παρακάτω[44]:

- Η εξόρυξη δεδομένων έχει στόχο την εύρεση πληροφορίας μέσα από δεδομένα μεγάλου όγκου. Στην συνέχεια, οι αναλυτές θα προσπαθήσουν να κατανοήσουν την πληροφορία που θα τους οδηγήσει σε πιο εύστοχες και αποδοτικές αποφάσεις. Αντίθετα, η μηχανική μάθηση συνήθως αποσκοπεί στην εύρεση πληροφορίας που θα βοηθήσει στην βελτίωση της απόδοσης,

αλγόριθμων, συστημάτων, όπως μοντέλων ταξινόμησης ή κάποιας τεχνικής οντότητας, π.χ. ενός ρομπότ.

- Η εξόρυξη δεδομένων εφαρμόζεται σε in-vivo(εν-ζωή, που επιδέχονται τροποποιήσεις) συλλογές δεδομένων, οι οποίες συχνά δεν σχεδιάστηκαν για την εξόρυξη τους, αλλά για την λειτουργικότητα συστημάτων ή ιστοσελίδων. Αντιθέτως, η μηχανική μάθηση εφαρμόζεται συνήθως σε in-vitro δεδομένα(σε αυστηρά ελεγχόμενες συνθήκες, εκτός ζωντανής επεξεργασίας) που συλλέχθηκαν ώστε να εφαρμοστεί σε αυτά κάποιος αλγόριθμος μηχανικής μάθησης.
- Η μηχανική μάθηση χρησιμοποιεί συνήθως μικρά δείγματα δεδομένων, σε αντίθεση με την εξόρυξη που ασχολείται με μεγαλύτερα δείγματα. Συνεπώς, η μηχανική μάθηση στοχεύει στην, όσο δυνατόν, καλύτερη απόδοση των συστημάτων, που έχουν σαν έξοδο προβλέψεις ή αποτελέσματα. Από την άλλη πλευρά, στόχο της εξόρυξης δεδομένων αποτελεί η επίτευξη της μείωσης της χρονικής πολυπλοκότητας των αλγορίθμων.[45]

Η εξόρυξη δεδομένων διέπεται από ορισμένους κανόνες και διαδικασίες. Αρχικά, πρέπει να γνωρίζουμε, πριν την έναρξη των διαδικασιών, καλά την φύση δεδομένων που στοχεύουμε να εξορύξουμε. Παράγοντες όπως ο θόρυβος ή ύπαρξη ιδιοτήτων που δεν σχετίζονται με την φύση και τον σκοπό της εκάστοτε εργασίας, στα δεδομένα, μπορεί να επηρεάσουν αρνητικά την εξαγόμενη πληροφορία. Για αυτό τον λόγο, απαιτείται η γνώση των δεδομένων και ίσως η προ επεξεργασία τους, ώστε να εκλείψουν, ή αν δεν είναι δυνατό, έστω να περιοριστούν οι αρνητικοί παράγοντες. Σύμφωνα με τον ορισμό των παραμέτρων και των στόχων που τίθενται, υλοποιούνται και οι αντίστοιχοι αλγόριθμοι. Τέλος, η αξία της πληροφορίας που εξορύχθηκε δεν γίνεται πάντοτε αντιληπτή, γι αυτό χρειάζεται η μελέτη και η αξιολόγηση των αποτελεσμάτων της διαδικασίας.[46]

4.5 Μεθοδολογία

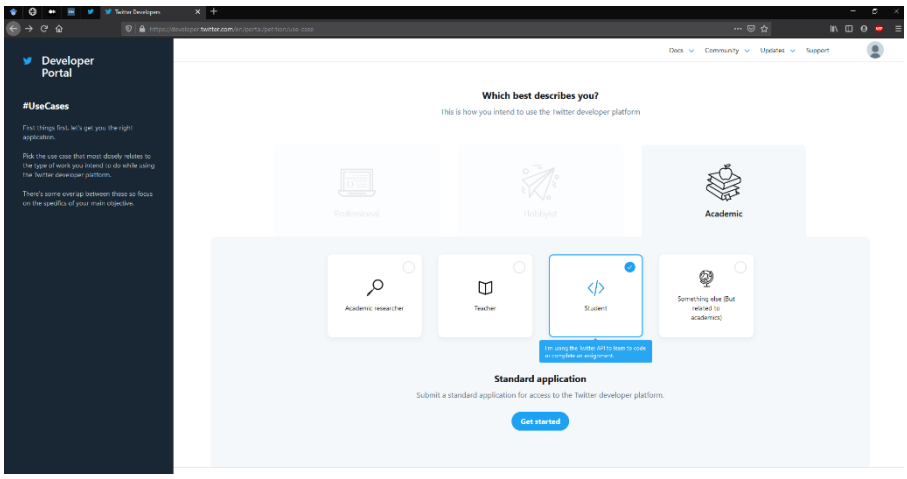
Στην πτυχιακή αυτή κάναμε ένα συνολικό σύνολο δεδομένων από 3 διαφορετικά, όμως του ίδιου τύπου για να συλλέξουμε αρκετά δεδομένα. Στο τελικό dataset όπως θα δούμε στην συνέχεια καταλήγουμε με 88.222 άρθρα. Από τα οποία 43458 ήταν παραπληροφόρηση ενώ τα 44764 ήταν αληθή. Τα δύο dataset τα οποία περιείχαν το 60% των άρθρων βρέθηκαν στο διαδίκτυο και με την κατάλληλη επεξεργασία όπως θα δείξουμε παρακάτω ήρθαν στην μορφή που τα θέλαμε. [53][54]

Το τρίτο σύνολο δεδομένων αποτελεί μέρος της πτυχιακής των Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, Huan Liu, και μπορεί να βρεθεί στον φάκελο τους στο GitHub με αναλυτικές οδηγίες εμείς θα δείξουμε βήμα βήμα πως κάναμε την ίδια δουλειά σε ένα Raspberry Pi4 (4Gb Ram).[52][55] Πρόκειται για ένα εργαλείο FakeNewsTracker το οποίο συνδυάζει ένα σύνολο πραγματικών και ψεύτικων άρθρων από τον ιστότοπο Politifact και τον Gossipcop που επισημάνθηκαν με μη αυτόματο τρόπο από ανθρώπους, με δεδομένα από το Twitter για χρήστες που έκαναν tweet στα άρθρα. Μπορούμε να συνδυάσουμε τα ανεπεξέργαστα δεδομένα για τα άρθρα με τα δεδομένα του Twitter για να κάνουμε έναν ακόμη καλύτερο ανιχνευτή ψευδών ειδήσεων από θα μπορούσαμε να κάνουμε μόνο με οποιαδήποτε πηγή. Ο τρόπος που επιτυγχάνεται αυτή η συλλογή είναι μέσω μιας εφαρμογής Flask η οποία με τα κατάλληλα API KEYS του Twitter μας δίνει την πρόσβαση για να συλλέξουμε αυτά τα δεδομένα. [52]

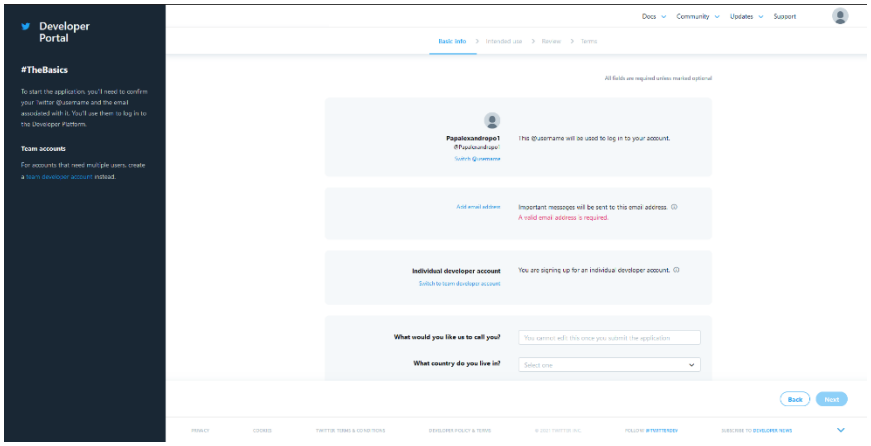
Ο λόγος που δεν παρέχεται κατευθείαν το πλήρες σύνολο δεδομένων είναι λόγω των πολιτικών απορρήτου του Twitter και των δικαιωμάτων αντιγραφής εκδότη ειδήσεων. Οι κοινωνικές δεσμεύσεις και οι πληροφορίες χρηστών δεν αποκαλύπτονται λόγω της Πολιτικής του Twitter επίσης. Θα δείξουμε και με ποιον τρόπο μπορούμε να δημιουργήσουμε ένα account στο Twitter Developer και πως μπορούμε να πάρουμε την έγκριση για την υλοποίηση παρόμοιων εφαρμογών. Αυτό είναι ιδιαίτερα σημαντικό γιατί μπορεί να χρησιμεύσει στην δημιουργία παρόμοιων εφαρμογών σαν αυτή.

4.5.1 Άδεια χρήσης από το twitter

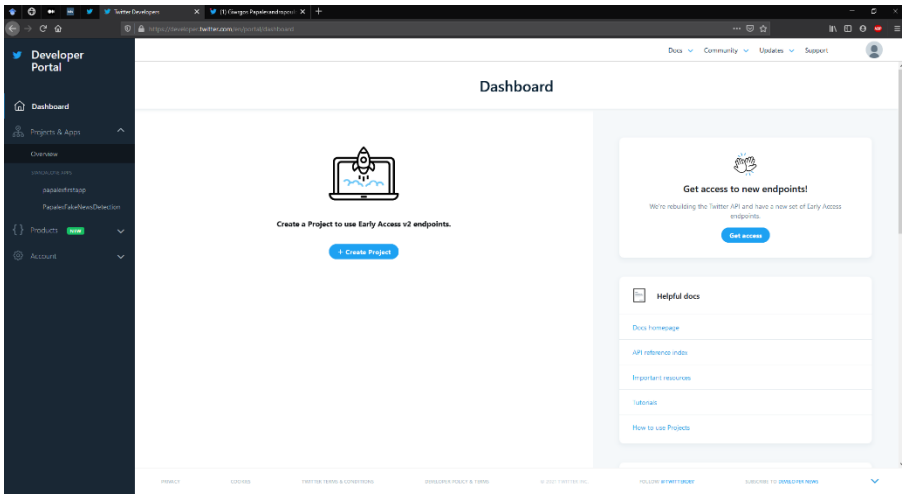
Βήμα 1: Δημιουργούμε ένα λογαριασμό με τα στοιχεία μας στο Twitter Developer Portal και επιλέγουμε σαν σκοπό του λογαριασμού ότι είμαστε μαθητές. Είναι ιδιαίτερα σημαντικό να προσέξουμε τα βήματα και να είμαστε ειλικρινείς γιατί σε πάρα πολλές περιπτώσεις και ανάλογα το σκοπό της εφαρμογής που θέλουμε να δημιουργήσουμε το Twitter μπορεί να μας απορρίψει.[60]



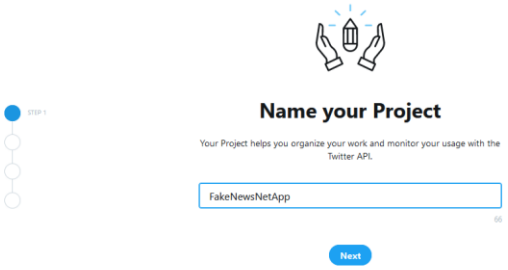
Βήμα 2: Εδώ συμπληρώνουμε τα επιπλέον στοιχεία μας και προσέχουμε να είναι το πεδίο σωστό σαν Individual Developer Account.



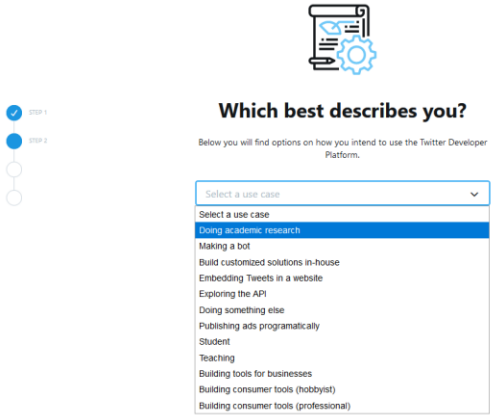
Βήμα 3: Στην συνέχεια δημιουργούμε μια νέα εφαρμογή.



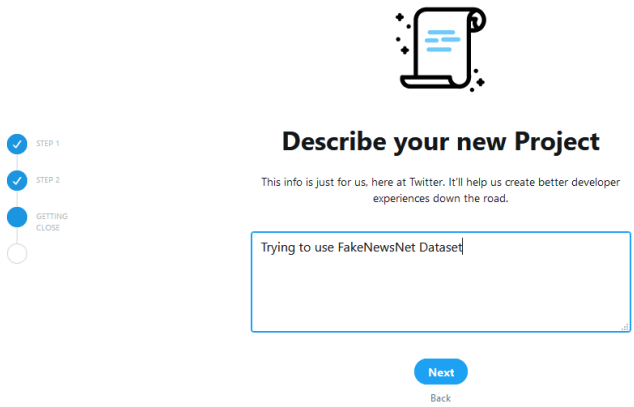
Βήμα 4: Επιλέγουμε ένα όνομα για το Προτζεκτ μας.



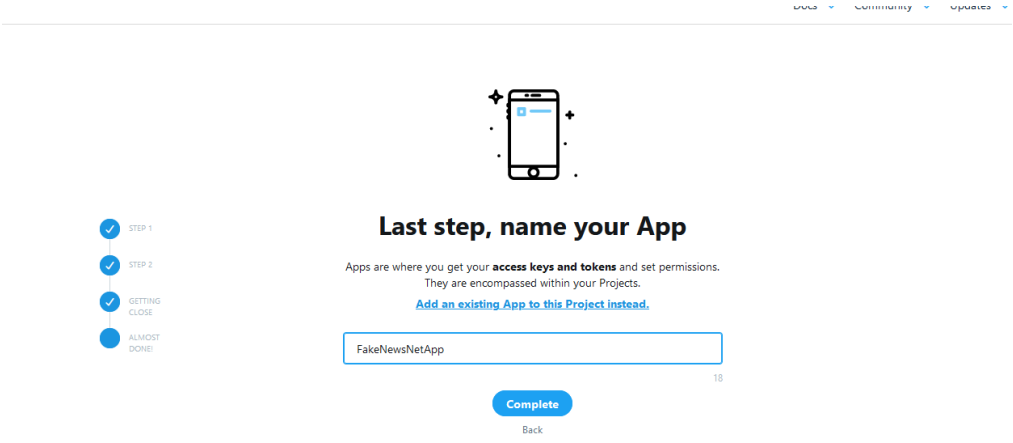
Βήμα 5: Επιλέγουμε για ποιον σκοπό χρησιμοποιούμε το API.



Βήμα 6: Στην συνέχεια δίνουμε τους λόγους αναλυτικά που δημιουργούμε την εφαρμογή αυτή για να εγκριθεί από το API του Twitter. Ξανά τονίζουμε είναι απαραίτητο να είμαστε αυστηροί και ειλικρινείς με τους λόγους, αρχικά διότι θα καθυστερήσει λιγότερο η έγκριση της εφαρμογής αλλά και γιατί υπάρχει πιθανότητα αν δεν τηρείται στα πλαίσια της πολιτικής της εταιρείας να μην εγκριθεί.



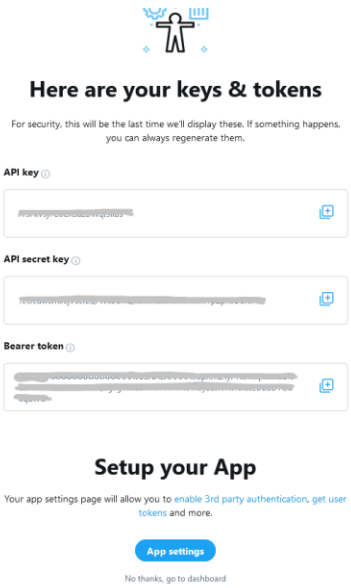
Βήμα 7: Δίνουμε ένα όνομα το οποίο δεν θα πρέπει να χρησιμοποιείται.



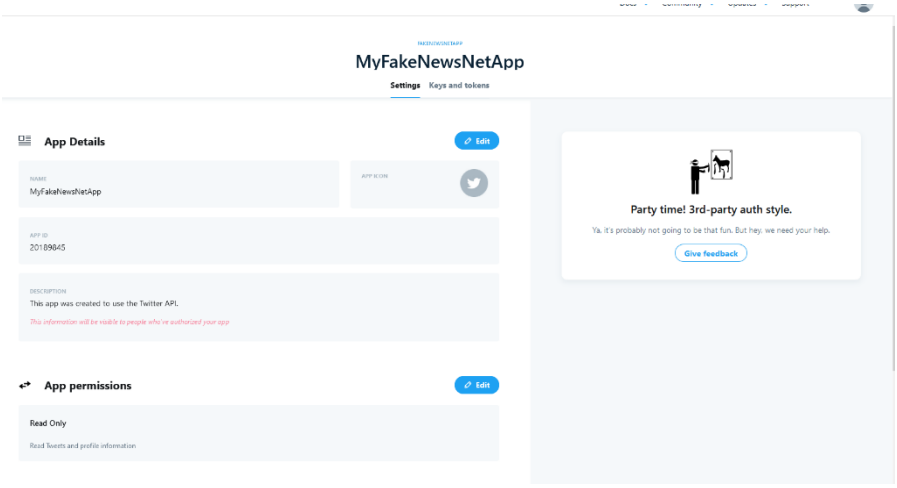
Εδώ είναι το σημείο όπου μπορεί να χρειαστεί να περιμένουμε, ειδικά στην περίπτωση όπου ο λογαριασμός είναι καινούριος. Την πρώτη φορά που τον δημιουργήσαμε περιμέναμε 36 ώρες αλλά αυτό ποικίλει

με τον φόρτο εργασίας αλλά και με το πόσο ξεκάθαροι ήμασταν στους λόγους υλοποίησης της εφαρμογής. Επίσης η εφαρμογή δεν θα πρέπει να υπονομεύει τα συμφέροντα της εταιρίας.

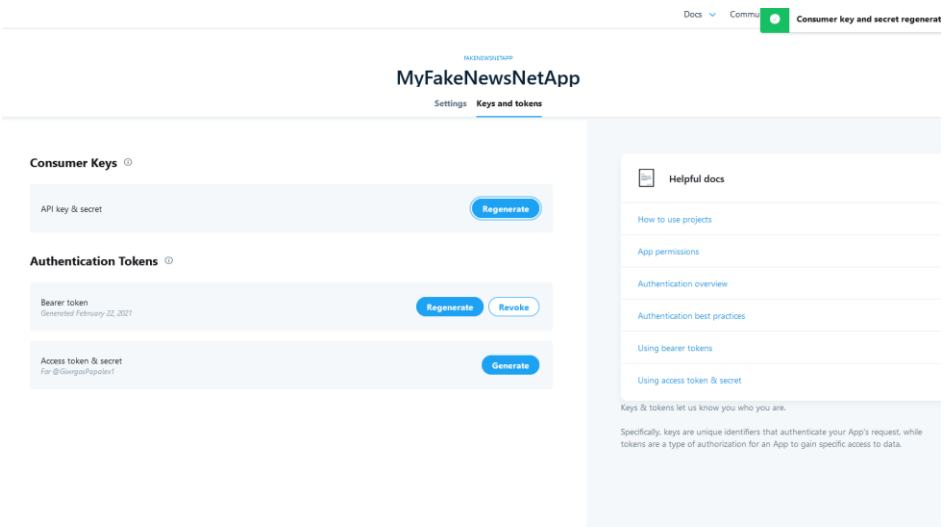
Βήμα 8: Στην συνέχεια όταν ολοκληρωθεί μας εμφανίζεται ένα παράθυρο με τα παρακάτω κλειδιά. Τα κλειδιά αυτά δεν πρέπει να τα δώσουμε σε κανέναν και να εμφανιστούν πουθενά. Είναι προσωπικά και οποιαδήποτε διαρροή τους μπορεί να μας δημιουργήσει προβλήματα.



Βήμα 9: Στην περίπτωση που κλείσαμε την ιστοσελίδα γιατί περιμέναμε την έγκριση και την ξανά ανοίξαμε για να βρούμε τα κλειδιά που χρειαζόμαστε, βρίσκουμε την εφαρμογή και επιλέγουμε στο κέντρο Keys and Tokens.



Βήμα 10: Στην συνέχεια καλό θα ήταν να κάνουμε παράγουμε (generate) ένα καινούργιο κλειδί. Κάθε φορά που παράγουμε καινούργιο κλειδί πρέπει να το αποθηκεύουμε και να το αλλάζουμε στην εφαρμογή μας προκειμένου να το χρησιμοποιήσουμε στην συνέχεια.

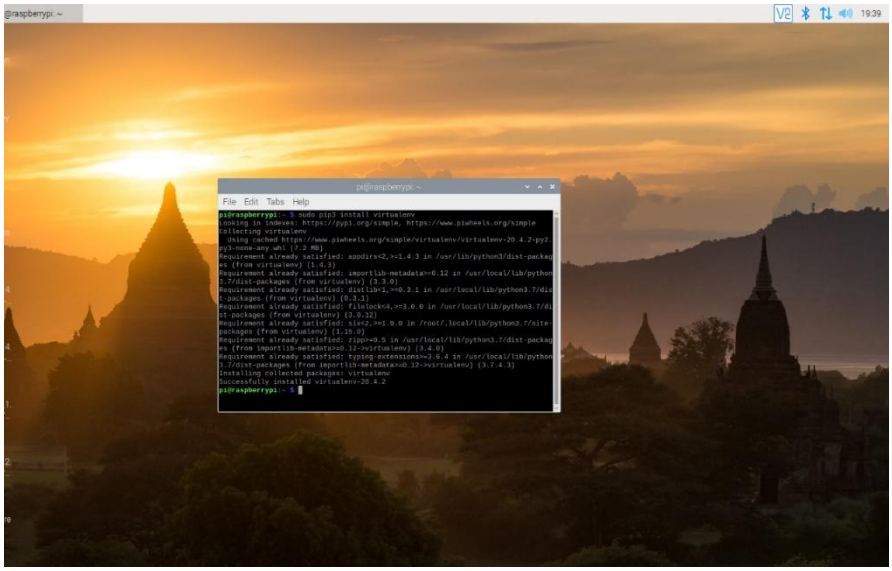


4.5.2 Πρόγραμμα συλλογής

Αφού λοιπόν δημιουργήσαμε τον λογαριασμό, πήραμε την έγκριση και στην συνέχεια αποθηκεύσαμε τα κλειδιά μας

Ένα από τα πιο απαραίτητα είναι η έκδοση την Python που χρησιμοποιούμε η οποία πρέπει να είναι τουλάχιστον στην έκδοση 3.6 και άνω. Επίσης προτείνουμε για να μην αλλάξουμε τις εκδόσεις στις βιβλιοθήκες όλου του συστήματος την εγκατάσταση και την χρήση ενός Virtual Enviroment η οποία είναι αρκετά απλή και γίνεται με χρήση της εντολής

-sudo pip3 install virtualenv

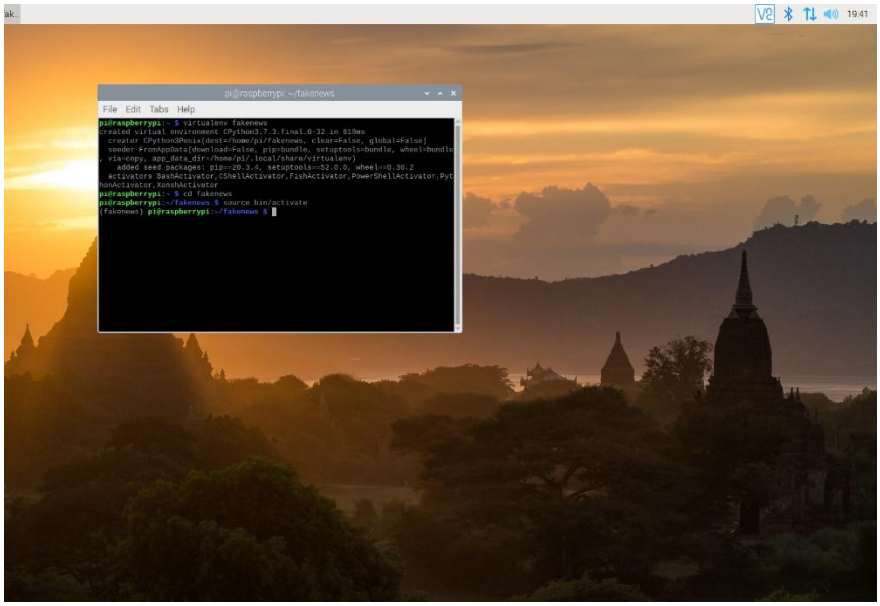


Δημιουργούμε ένα virtual environment με ότι όνομα θέλουμε π.χ. fakenews και στην συνέχεια εισερχόμαστε μέσα στον φάκελο αυτό για να τον ενεργοποιήσουμε με τις εντολές.

-virtualenv fakenews

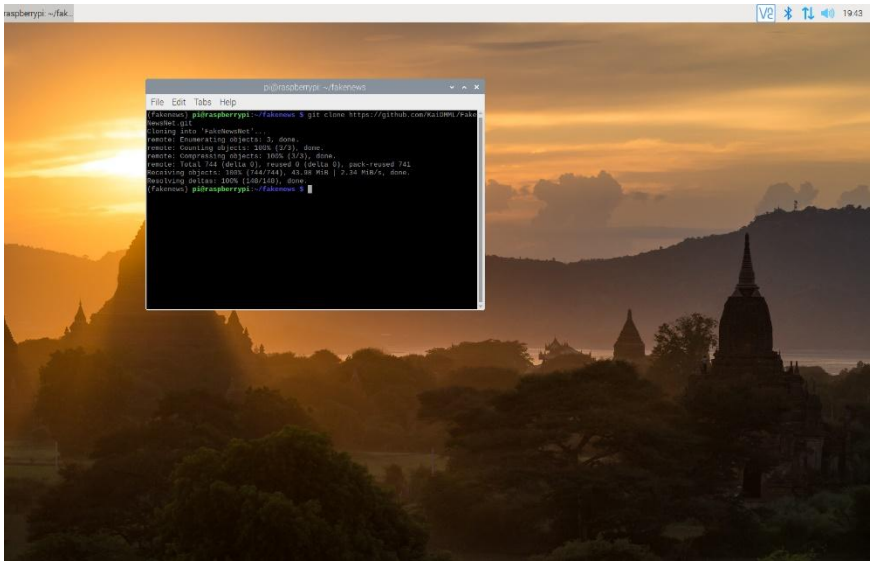
- cd fakenews

-source bin/activate



Στην συνέχεια κατεβάζουμε τον φάκελο από το github που υπάρχει για το σύνολο δεδομένων που θέλουμε.

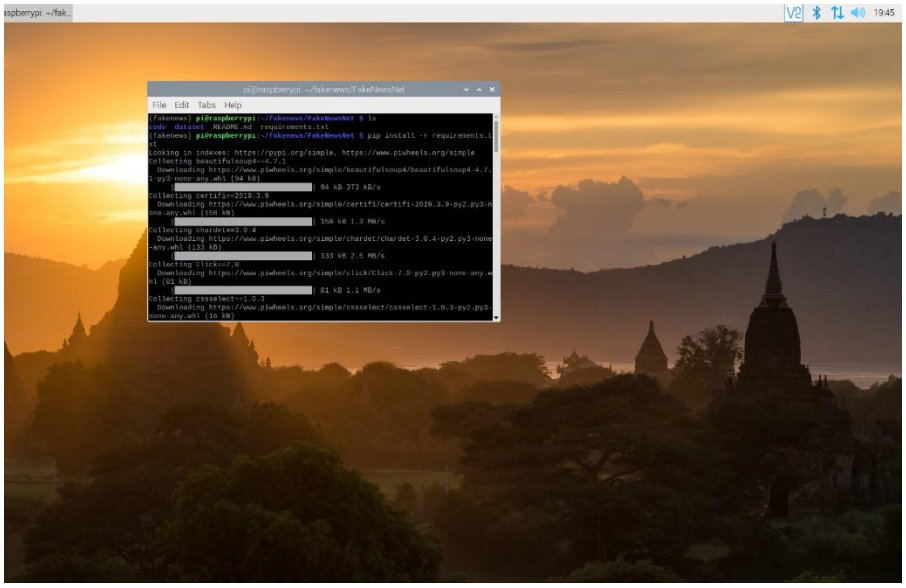
- git clone https://github.com/KaiDMML/FakeNewsNet.git



Ο λόγος όπως είπαμε που εγκαταστήσαμε και χρησιμοποιούμε ένα Virtual Enviroment είναι για να μην αλλάξουμε τις εκδόσεις και τις βιβλιοθήκες του συστήματος. Επομένως μπαίνουμε στον φάκελο που κατεβάσουμε και κάνουμε εγκατάσταση τα απαραίτητα.

-cd FakeNewsNet

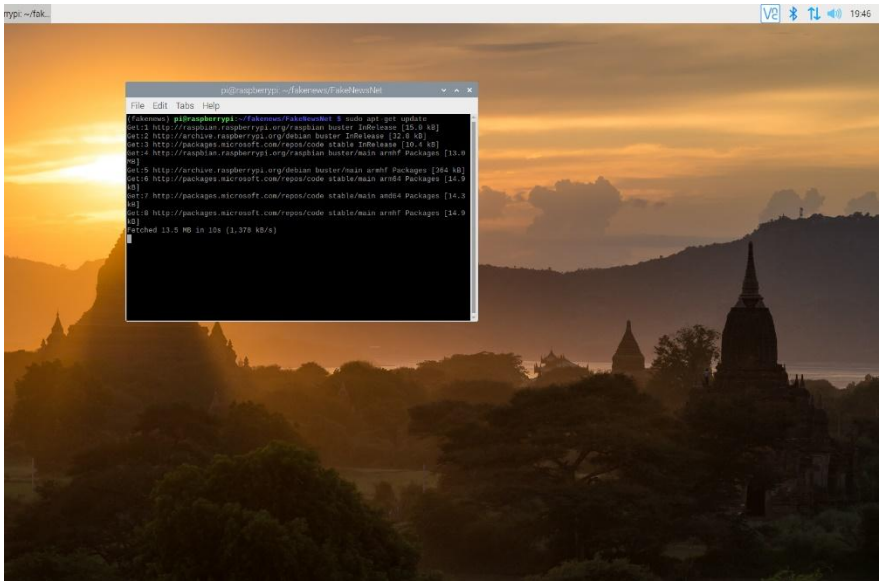
-pip install -r requirements.txt



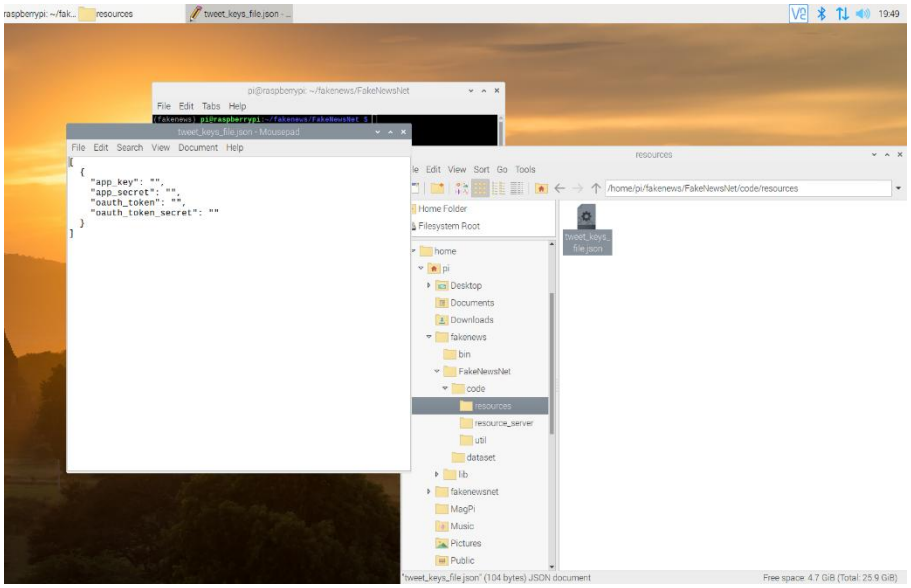
Καλό είναι σε αυτό το σημείο να κάνουμε ένα update το σύστημά μας με τις εντολές που χρησιμοποιούμε και στα συστήματα Linux αφού αποτελεί ανεπτυγμένο μέρος και προσαρμοσμένο στο Raspberry του Debian.

-sudo apt-get update

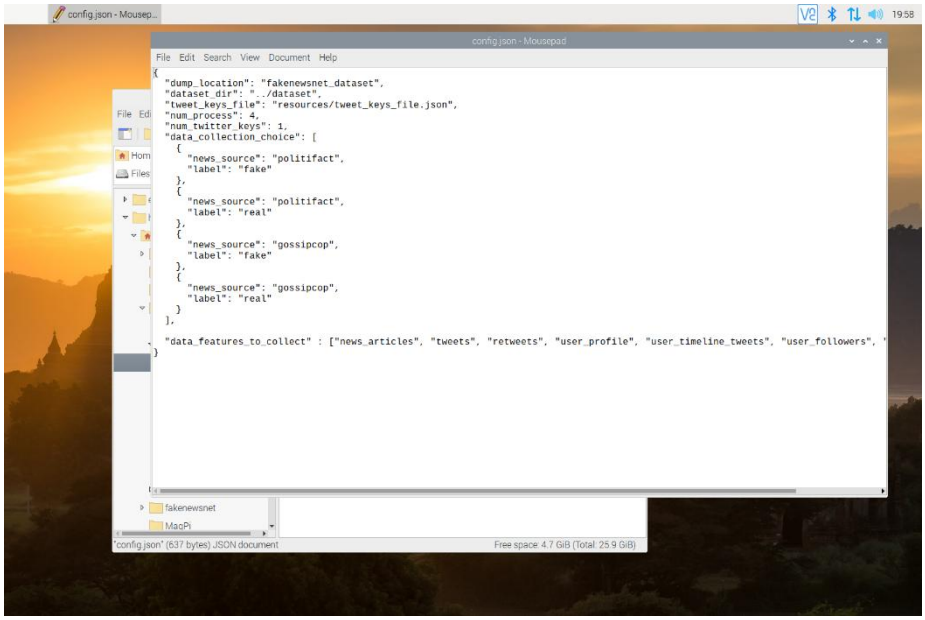
-sudo apt-get upgrade



Για να λειτουργήσει λοιπόν η εφαρμογή και για να συνδέσουμε τον λογαριασμό και το προτζεκτ που δημιουργήσαμε νωρίτερα στο twitter developer κάτω από τον φάκελο code/resource υπάρχει ένα αρχείο με όνομα tweet_keys_file.json και εκεί θα βάλουμε τα κλειδιά που αποθηκεύσαμε από το προτζεκτ μας από την ιστοσελίδα. Συνεπώς το API key & secret πηγαίνουν στα app_key και app secret και αντίστοιχα τα Access token & secret πηγαίνουν στα oauth_toke και oauth_token_secret.

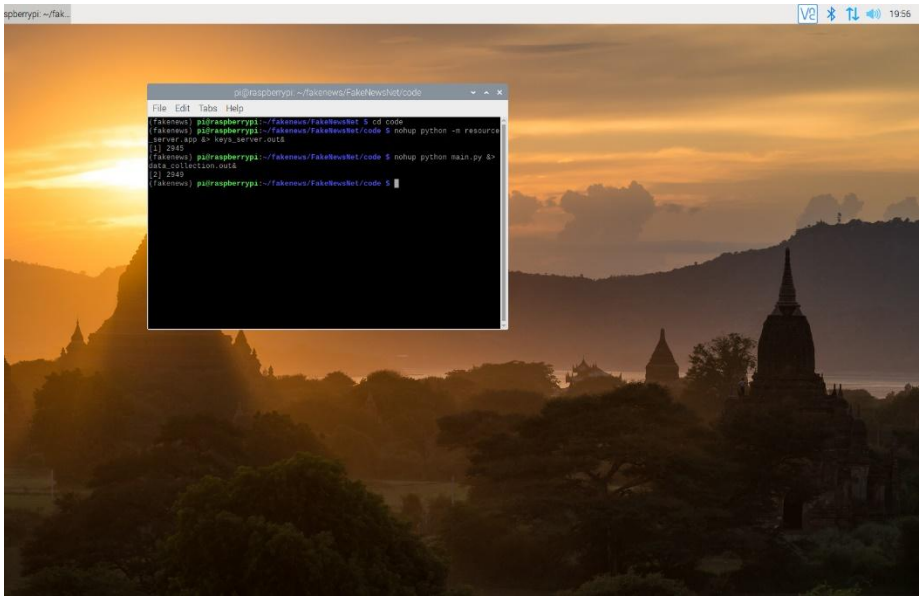


Αν θέλουμε να μην συλλέξουμε όλα τα δεδομένα για να κάνουμε πιο ελαφριά την εφαρμογή μας μπορούμε να αλλάξουμε ποια στοιχεία θα συλλέξει απλά πηγαίνοντας στο αρχείο code/config.json και αλλάζοντας τα πεδία στο data_features_to_collect. Εμείς επιλέγουμε να τα συλλέξουμε όλα.



Στην συνέχεια όλα είναι έτοιμα και για να ξεκινήσουμε την εφαρμογή μας εισερχόμαστε στο φάκελο code και εκτελούμε τις παρακάτω εντολές.

- nohup python -m resource_server.app &> keys_server.out&
- nohup python main.py &> data_collection.out&

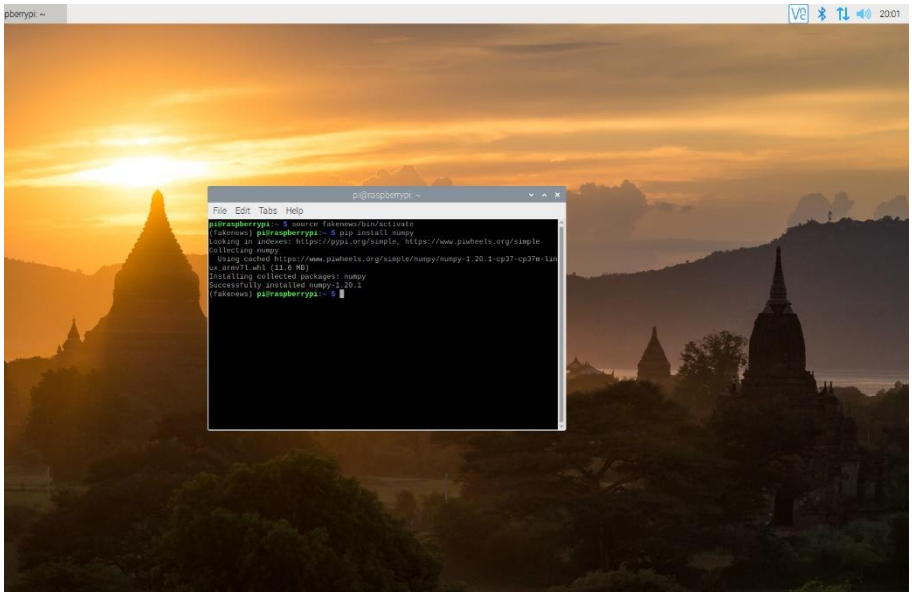


Στο αρχείο data_collection.out όπως του υποδείξαμε στην τελευταία εντολή (&> data_collecton.out&)μπορούμε να παρατηρούμε την κατάσταση την συλλογής δεδομένων μας.

Στην συνέχεια επειδή η εφαρμογή πήρε αρκετή ώρα και δεν τερματίστηκε επιλέξαμε να την σταματήσουμε έχοντας συλλέξει συνολικά 17.324 άρθρα και τα tweets τους από τις δύο ιστοσελίδες το Politifact και το Gossipcop, από τα οποία 4.893 ήταν ψευδής και τα υπόλοιπα 12.431 ήταν αληθής. Αφού λοιπόν τερματίσαμε την εφαρμογή κάνουμε ξανά ενεργοποίηση του εικονικού περιβάλλοντος και κάνουμε εγκατάσταση δύο πολύ σημαντικές βιβλιοθήκες , την Numpy και την Pandas.

-pip install numpy

-pip install pandas



4.5.3 Επεξεργασία δεδομένων tweet

Στην συνέχεια θέλουμε να εξάγουμε όλα τα απαραίτητα στοιχεία από κάθε αρχείο που συλλέξαμε. Ένας πολύ αποδοτικός τρόπος είναι αυτός που αναφέρεται από τον Matthew Whitehead και θα αναλύσουμε στην συνέχεια.[57]

Φτιάχνουμε μια συνάρτηση η οποία δέχεται σαν ορίσματα τον φάκελο, δηλαδή την ιστοσελίδα που συλλέχθηκαν και τον υπό φάκελο για το αν είναι αληθής η ψευδής. Στην συνέχεια κάνουμε για κάθε άρθρο και τα αντίστοιχα tweets τους εξαγωγή τον πληροφοριών που βρίσκονται στα αντίστοιχα json αρχεία και δημιουργούμε ένα τελικό dictionary με λίστες. Επίσης επειδή δεν υπάρχουν κάποια άρθρα και κάποια tweets απλά δεν θα συμπεριληφθούν στο τελικό σύνολο δεδομένων.

```
'''
**********************
This code is property of Matthew Whitehead
**********************

I show and i explain step by step of what he did because it is the same exact way i wanted to do
'''

import json
import os
import numpy as np
import pandas as pd


def load_data(news_source, truthfulness):
    temp = []

    for i in os.listdir('{}/{}/'.format(news_source, truthfulness)):
        try:
            with open('{}/{}/{}/news content.json'.format(news_source, truthfulness, i), 'rb') as file:
                article = json.load(file)

            tweets = []
            for j in os.listdir('{}/{}/{}/tweets/'.format(news_source, truthfulness, i)):
                with open('{}/{}/{}/tweets/{}'.format(news_source, truthfulness, i, j), 'rb') as file2:
                    tweet = json.load(file2)
                tweets.append(tweet)

        except FileNotFoundError:
            continue
        except NotADirectoryError:
            continue

        temp.append({'id': i, 'article': article, 'tweets': tweets})

    return temp
```

Κάθε άρθρο έχει πολλά Tweets που σχετίζονται με αυτό και κάθε άρθρο μπορεί να έχει διαφορετικό αριθμό Tweets. Επομένως, συνοψίζουμε μερικά από τα χαρακτηριστικά που υπάρχουν συνολικά.

```python
def gather_twitter_stats(tweets):
    follower_counts = []
    friends_counts = []
    favorite_counts = []
    retweet_counts = []
    statuses_counts = []
    verified_counter = 0

    for tweet in tweets:
        follower_counts.append(tweet['user']['followers_count'])
        friends_counts.append(tweet['user']['friends_count'])
        favorite_counts.append(tweet['favorite_count'])
        retweet_counts.append(tweet['retweet_count'])
        statuses_counts.append(tweet['user']['statuses_count'])
        if tweet['user']['verified']:
            verified_counter += 1

    return {
        'followers_mean': np.mean(follower_counts),
        'followers_std': np.std(follower_counts),
        'followers_median': np.median(follower_counts),
        'followers_sum': np.sum(follower_counts),
        'friends_mean': np.mean(friends_counts),
        'friends_std': np.std(friends_counts),
        'friends_median': np.median(friends_counts),
        'friends_sum': np.sum(friends_counts),
        'favorites_mean': np.mean(favorite_counts),
        'favorites_std': np.std(favorite_counts),
        'favorites_median': np.median(favorite_counts),
        'favorites_sum': np.sum(favorite_counts),
        'retweets_mean': np.mean(retweet_counts),
        'retweets_std': np.std(retweet_counts),
        'retweets_median': np.median(retweet_counts),
        'retweets_sum': np.sum(retweet_counts),
        'statuses_mean': np.mean(statuses_counts),
        'statuses_std': np.mean(statuses_counts),
        'statuses_median': np.median(statuses_counts),
        'statuses_sum': np.sum(statuses_counts),
        'verified_count': verified_counter
    }
```

Στην συνέχεια μια ακόμα συνάρτηση για να εξάγουμε τον τίτλο, το κείμενο και την ετικέτα (ψευδής η αληθής) που είναι τα χαρακτηριστικά που μας ενδιαφέρουν περισσότερο και τα επιστρέφουμε επίσης σε ένα dictionary.

```python

def process_example(ex, label):
    uid = ex['id']
    article_text = ex['article']['text']
    article_title = ex['article']['title']
    article_source = ex['article']['source']

    temp = {'id': uid, 'title': article_title, 'text': article_text, 'source': article_source, 'label': label}
    temp.update(gather_twitter_stats(ex['tweets']))

    return temp
```

Τέλος καλούμε όλες τις συναρτήσεις που φτιάξαμε και δημιουργούμε ένα dataframe της pandas βιβλιοθήκης και τα εξάγουμε σε ένα τελικό σύνολο δεδομένων.

```
91
92    pf_fake = load_data('politifact', 'fake')
93    pf_fake = [process_example(x, 'fake') for x in pf_fake]
94    pf_real = load_data('politifact', 'real')
95    pf_real = [process_example(x, 'real') for x in pf_real]
96    gc_fake = load_data('gossipcop', 'fake')
97    gc_fake = [process_example(x, 'fake') for x in gc_fake]
98    gc_real = load_data('gossipcop', 'real')
99    gc_real = [process_example(x, 'real') for x in gc_real]
```

```
100
101
102   df = pd.DataFrame(pf_fake)
103   df = df.append(pf_real)
104   df = df.append(gc_fake)
105   df = df.append(gc_real)df.reset_index(inplace=True, drop=True)df.to_csv('dataset.csv', index=False)
```

4.5.4 Προετοιμασία συνολικού συνόλου δεδομένων

Ο τρόπος με τον οποίο δημιουργήσουμε το τελικό alldataset.csv αρχείο με τα 88.222 άρθρα μαζί με αυτά που κατεβάσαμε από τις πηγές [53][54] είναι ο εξής.

Δημιουργούμε αρχικά μια συνάρτηση η οποία δέχεται τον φάκελο στόχο που περιέχονται τα csv αρχεία από τα άλλα δύο σύνολα δεδομένων και ανάλογα αν είναι δύο ή τρία αρχεία επιστρέφονται αντίστοιχα dataframes. –Τα Dataframes είναι ένα σύνολο δεδομένων που μας δίνεται με την βιβλιοθήκη pandas με την μορφή σαν ένα εικονικό excel αρχείο με στήλες, γραμμές και κελιά για κάθε εγγραφή. Χρησιμοποιήστε την και σε άλλες περιπτώσεις, όμως είναι μια βιβλιοθήκη που την ενσωματώνουν σχεδόν όλοι όσοι έχουν να τροποποιήσουν csv και excel.

```python
# A function that will open each csv file we have on our specific folders as pandas Dataframes
def load_data(targetfolder):
    # We first check if we are on the archive folder because its the only one that
    # will return 2 dataframes in result of the true and fake csv files it contains
    if (len([filesin for filesin in os.listdir('{}/'.format(targetfolder))])) == 2:
        for i in os.listdir('{}/'.format(targetfolder)):
            try:
                if i == 'True.csv':
                    df_true = pd.read_csv('{}/{}'.format(targetfolder, i))
                else:
                    df_fake = pd.read_csv('{}/{}'.format(targetfolder, i))
            except FileNotFoundError:
                continue
            except NotADirectoryError:
                continue
        return df_fake, df_true
    # The other 3 folders contains 3 csv files so we separate them to return 3 dataframes
    else:
        for i in os.listdir('{}/'.format(targetfolder)):
            try:
                if (i == 'train.csv'):
                    df_train = pd.read_csv('{}/{}'.format(targetfolder, i))
                elif (i == 'test.csv'):
                    df_test = pd.read_csv('{}/{}'.format(targetfolder, i))
                else:
                    df_dev = pd.read_csv('{}/{}'.format(targetfolder, i))
            except FileNotFoundError:
                continue
            except NotADirectoryError:
                continue
        return df_test, df_train, df_dev
```

Έπειτα αφού καλέσουμε την συνάρτηση και αποθηκεύσουμε τα dataframes εισάγουμε στήλες με βάση τις ετικέτες που θέλουμε, για κάθε ψευδής εισάγουμε 0 ενώ για κάθε αληθής 1.

```python
df_arc_fake, df_arc_true = load_data('archive')
df_test, df_train, df_sub = load_data('fake-news')
df_fakenews = pd.read_csv('dataset.csv')

# We create a new column at first dataset for their labels if its false it gets 0 and if its true it gets 1
# After thath we merge the two dataframes with specific columns we want and we get one last dataframe
df_arc_fake['label'] = 0
df_arc_true['label'] = 1
```

Φτιάχνουμε ένα τελικό dataframe με τις στήλες που χρειαζόμαστε.

```python
df_arc_true = pd.DataFrame(df_arc_true, columns=[
                           'title', 'text', 'label'])
df_arc_fake = pd.DataFrame(df_arc_fake, columns=[
                           'title', 'text', 'label'])
df_arc = pd.concat([df_arc_fake, df_arc_true], ignore_index=True)
```

Επειδή σε αυτό το σύνολο δεδομένων υπάρχουν 3 αρχεία και το ένα περιέχει εγγραφές που περιέχοντε ήδη στο αρχείο test.csv κάνουμε τις αντίστοιχες εργασίες ώστε να καταλήξουμε σε ένα τελικό dataframe με τις ίδιες στήλες όπως στα προηγούμενα.

```
61
62
63  # We do the same for the next datasets
64  df_test.sort_values('id')
65  df_sub.sort_values('id')
66  mylabellist = list(df_sub['label'])
67  for i in range(len(mylabellist)):
68      if df_test.loc[i, 'id'] == df_sub.loc[i, 'id']:
69          df_test.loc[i, 'label'] = mylabellist[i]
70
71  df_test = pd.DataFrame(df_test, columns=['title', 'text', 'label'])
72  df_train = pd.DataFr  df_test: DataFrame  s=['title', 'text', 'label'])
73  df_unk = pd.concat([df_test, df_train], ignore_index=True)
74  df_unk['label'] = df_unk['label'].astype(int)
75
76
```

Τέλος δημιουργούμε ένα τελικό κάνοντας συνδυασμό των τριών και το εξάγουμε στο τελικό alldataset.csv αρχείο που θα χρησιμοποιήσουμε στην συνέχεια για την κατασκευή των μοντέλων.

```
75
76
77  df_all = pd.concat([df_unk, df_arc, df_fakenews], ignore_index=True)
78  print(len(df_all))
79  df_all.to_csv(r'alldataset.csv', index=False)
80
```

5. Εγκατάσταση Tensorflow σε Raspberry

Σαν πλατφόρμα ανάπτυξης των μοντέλων μας επιλέχθηκε η υπολογιστική πλατφόρμα Raspberry. Μια ευρέως διαδεδομένη και αρκετά φθηνή λύση που προσφέρετε στην αγορά για ανάπτυξη μικρών συστημάτων αφού διακατέχεται από μικρά επιπέδου, ταχύτητας, υλικά και εργαλεία. Παρουσιάζουμε τον τρόπο με τον οποίο καταφέραμε να εγκαταστήσουμε το TensorflowLite που χρησιμοποιείται σε πλατφόρμες σαν αυτή αφού είναι μια ειδικά διαμορφωμένη μικρότερη έκδοση της κανονικής Tensorflow βιβλιοθήκης. Ο τρόπος με τον οποίο εγκαθίσταται αλλάζει συνεχώς για να είναι ευκολότερο ως προς τους χρήστες να το χρησιμοποιήσουν. Εμείς κάνουμε εγκατάσταση την έκδοση 2.0.0 που χρειαζόμαστε για τα μοντέλα μας, στην περίπτωση που κάποιος θέλει να εγκαταστήσει κάποια άλλη έκδοση πρέπει πρώτα να ψάξει αν υποστηρίζεται στην πλατφόρμα Raspberry και στην συνέχεια το μόνο που χρειάζεται είναι σε κάποιες εντολές που θα επισημάνουμε να αλλάξει και να βάλει τον αντίστοιχο αριθμό.

5.1 Οδηγός εγκατάστασης

Οι τρόποι με τους οποίους μπορούμε να εγκαταστήσουμε την Tensorflow στην Raspberry πλατφόρμα μας είναι αρκετοί. Το ίδιο το documentation της βιβλιοθήκης την παρούσα στιγμή που γράφτηκε αυτή η πτυχιακή μας προτείνει να εγκαταστήσουμε απλά κάνοντας

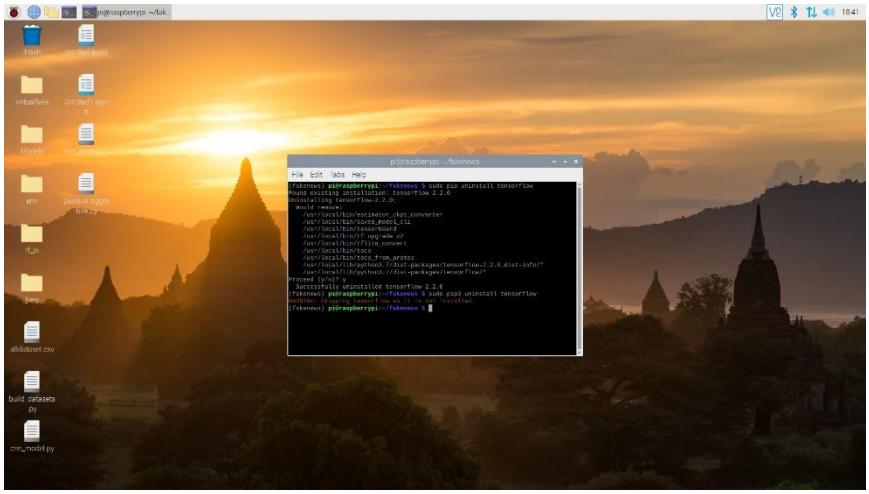
-sudo pip install tensorflow

Όμως στην δικιά μας περίπτωση μας εγκαθηστεί την έκδοση 1.14.0 η οποία δεν μπορεί να χρησιμοποιθεί στην συνέχεια για περαιτέρω εργασίες, γι αυτό λοιπόν προτείνουμε τον εξής τρόπο:

Αρχικά θα δούμε αν είναι ήδη εγκατεστημένο το Tensorflow και θα το απεγκαταστήσουμε.

-sudo pip uninstall tensorflow

-sudo pip3 uninstall tensorflow

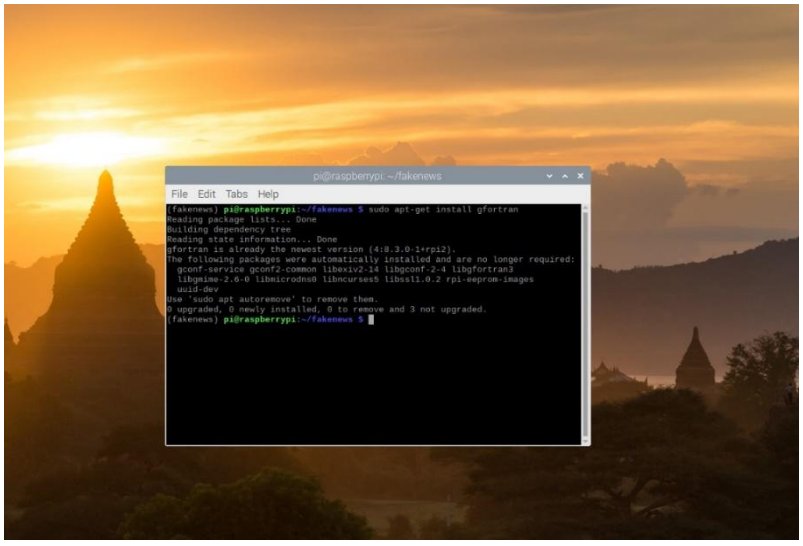


Και ξεκινάμε την εγκατάσταση όλων των απαραίτητων βιβλιοθηκών[58]. Αλλά πρώτα είναι ιδιαίτερα σημαντικό επίσης να κάνουμε απεγκατάσταση της βιβλιοθήκης wrapt γιατί αν είναι σε παλιότερη έκδοση δεν θα μας αφήνει να κάνουμε αργότερα εγκατάσταση στο tensorflow.

-sudo rm /usr/lib/python3/dist-packages/wrapt-1.10.11.egg-info

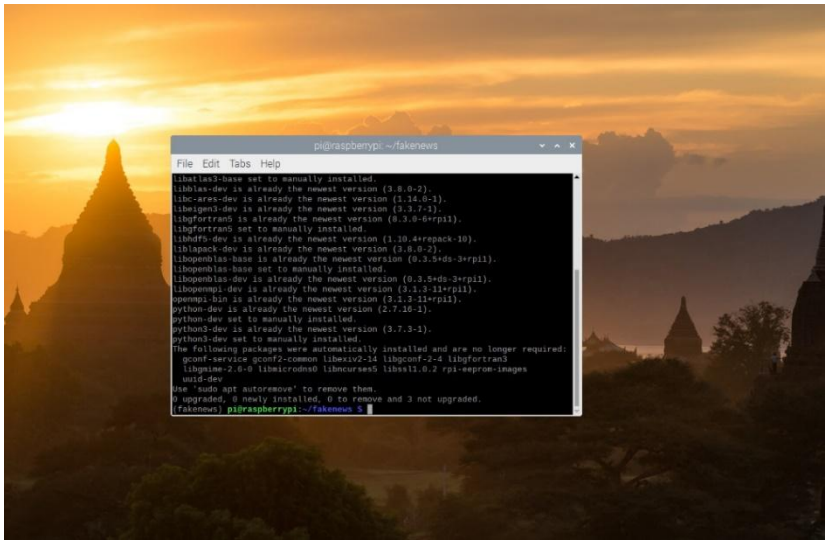
Ξεκινάμε λοιπόν κάνοντας εγκατάσταση την gfortran.

- sudo apt-get install gfortran



Επειδή όμως κατά την ολοκλήρωση υπάρχει περίπτωση να μην μας αφήνει το σύστημα να την χρησιμοποιήσουμε την Tensorflow επιστρέφοντας πρόβλημα HadoopFileSystemError κάνουμε εγκατάσταση τις παρακάτω βιβλιοθήκες. Μπορούμε να τις εγκαταστήσουμε και κάθε μια ξεχωριστά αλλά για δικούς μας λόγους ευκολίας προτιμάμε να τις κάνουμε όλες μαζί.

- sudo apt-get install -y libhdf5-dev libc-ares-dev libeigen3-dev gcc gfortran python-dev libgfortran5 \ libatlas3-base libatlas-base-dev libopenblas-dev libopenblas-base libblas-dev \ liblapack-dev cython libatlas-base-dev openmpi-bin libopenmpi-dev python3-dev



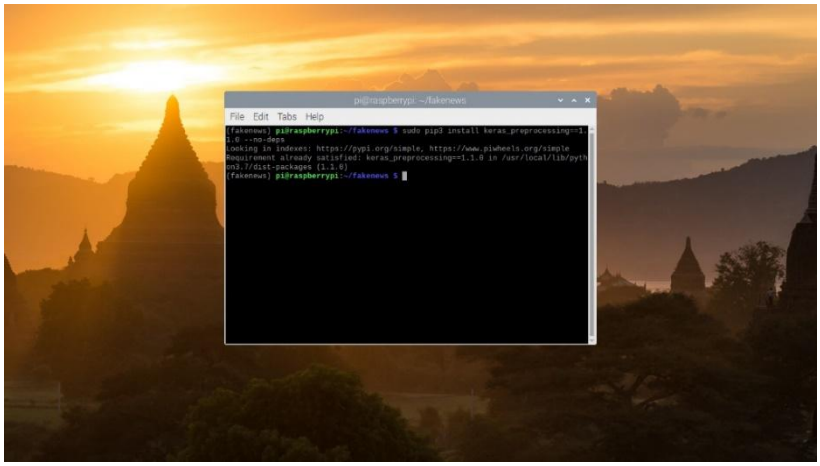
Και συνεχίζουμε:

-sudo pip3 install keras_applications==1.0.8 --no-deps

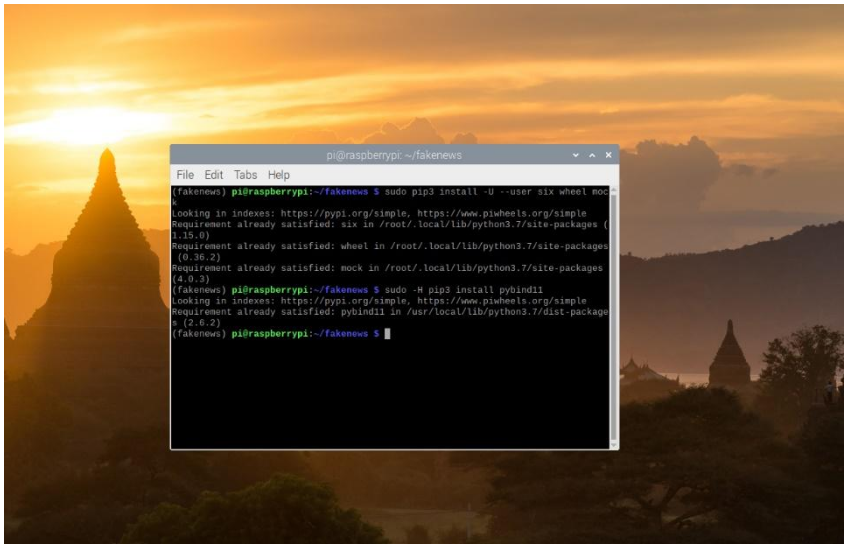
-sudo pip3 install keras_preprocessing==1.1.0 --no-deps

-sudo pip3 install h5py==2.9.0

-sudo pip3 install pybind11

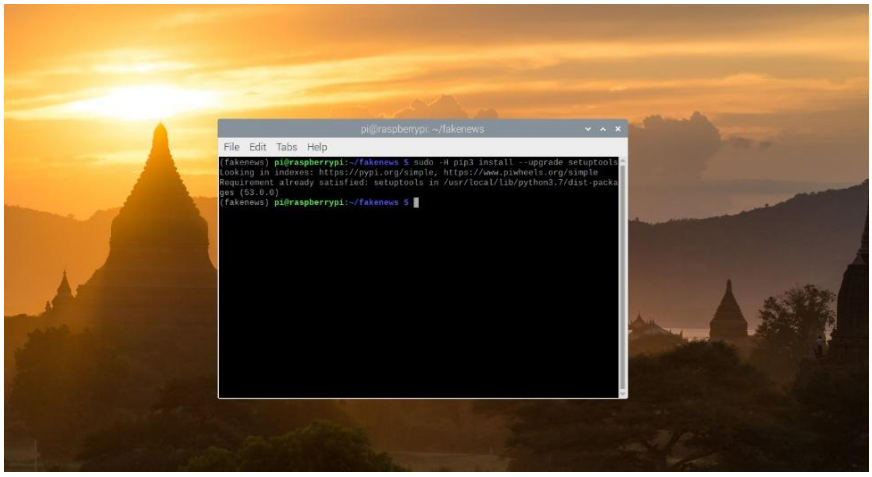


- sudo pip3 install -U --user six wheel mock



Και κάνουμε μια αναβάθμισή τα setuptools:

- sudo -H pip3 install --upgrade setuptools



Και τέλος κατεβάζουμε την βιβλιοθήκη. Μπορούμε να κατεβάσουμε και να περάσουμε όποια έκδοση θέλουμε αρκεί να υποστηρίζεται από το TensorflowLite. Την παρούσα χρονική στιγμή η τελευταία έκδοση ήταν η 2.2.0

-wget "https://raw.githubusercontent.com/PINTO0309/Tensorflow-bin/master/tensorflow-2.2.0-cp37-none-linux_armv7l_download.sh"

Αλλάζουμε τα δικαιώματα του αρχείου για να μπορέσουμε να κάνουμε αποσυμπίεση και εγκατάσταση.

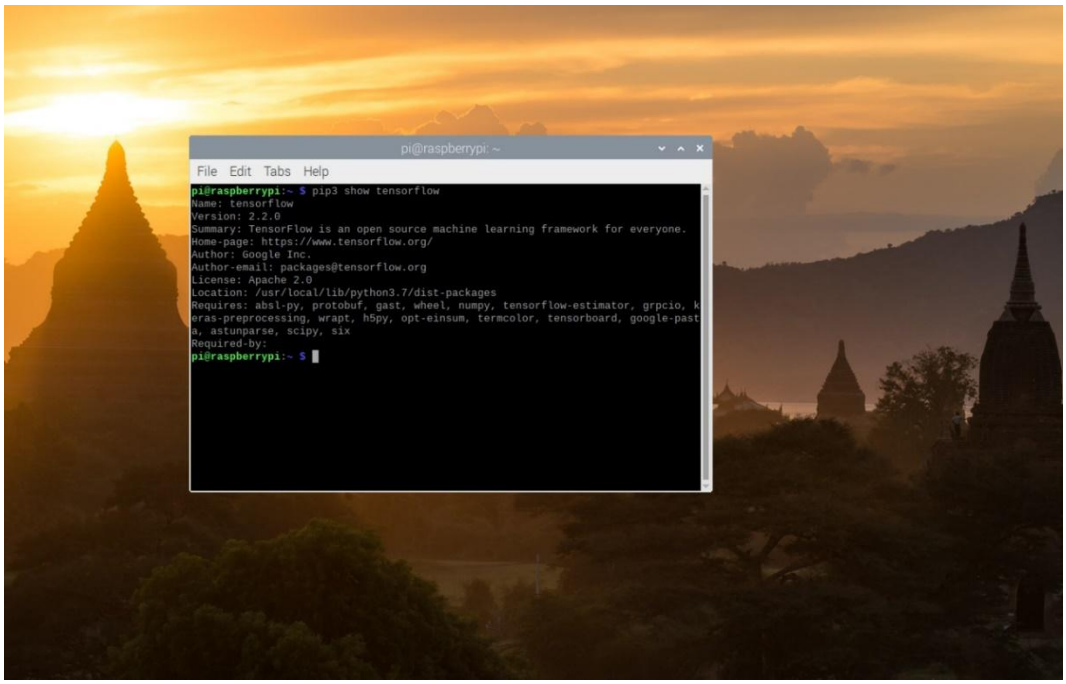
-chmod u+x tensorflow-2.2.0-cp37-none-linux_armv7l_download.sh

- ./tensorflow-2.2.0-cp37-none-linux_armv7l_download.sh

Και εγκαθιστούμε την έκδοση που κατεβάσαμε και στην συνέχεια κάνουμε έναν έλεγχο.

-sudo -H pip3 install tensorflow-2.2.0-cp37-none-linux_armv7l.whl

-pip3 show tensorflow



6. Δίκτυα που Υλοποιήθηκαν

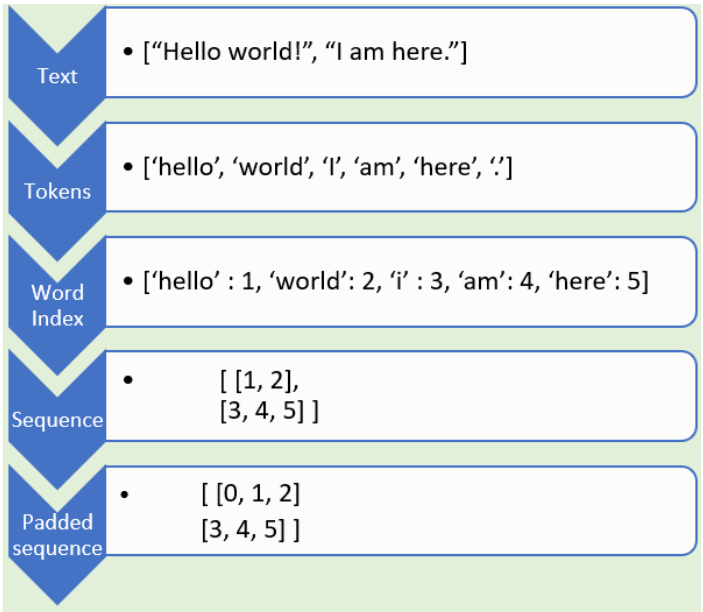
Σε αυτό το κεφάλαιο θα παραθέσουμε τμήματα από τους κώδικες των αρχείων μας και θα αναλύσουμε ποια μεθοδολογία ακολουθήσαμε για την υλοποίηση. Τον κοινό τρόπο που χρησιμοποιήσαμε σε όλα από την βιβλιοθήκη Tensorflow και τον τρόπο που τα μετατρέψαμε για να μπορούν να χρησιμοποιηθούν και στο Raspberry.

6.1 Προ επεξεργασία Κειμένου

Με παρόμοιο τρόπο όπως αναφέραμε νωρίτερα θα ανοίξουμε το σύνολο δεδομένων μας.

```
df = pd.read_csv('alldataset.csv')
df = df.fillna(' ')
```

Έπειτα στόχος μας είναι να μετατρέψουμε κάθε άρθρο σε ξεχωριστές λέξεις οι οποίες θα παρουσιαστούν μέσα στους κατηγοριοποιητές ή στα δίκτυα μας σαν αριθμοί. Για να επιτευχθεί αυτό αρχικά κάνουμε Tokenize δηλαδή κάθε λέξη παίρνει θέση σε μια λίστα σαν μια εγγραφή. Στην συνέχεια θέτουμε έναν αριθμό τυχαίο για κάθε λέξη ώστε να αναπαρασταθεί σαν αριθμός με την συνάρτηση text_to_sequences και στην συνέχεια με την pad_sequences προσθέτουμε μηδενικά ή στο τέλος η στην αρχή ώστε όλα τα άρθρα-λίστες να έχουν το ίδιο μήκος με βάση μια μεταβλητή ανάλογη του μέσου όρου του πλήθους των λέξεων στα άρθρα μας.



Εικόνα 19: Διαδικασία μορφοποίησης κειμένο

Η διαδικασία αυτή γίνεται προσαρμοσμένη στο σύνολο δεδομένων μας διαχωρίζοντας τις λίστες του τίτλου και του άρθρου.

```
MAX_TITLE = 7000
MAX_TEXT = 12000

tokenizer = Tokenizer(num_words=None, filters='!"#$%&()*+,-./:;<=>?@[\\]^_`{|}~\t\n',
                      lower=True, split=' ', char_level=False, oov_token=None, document_count=0)

tokenizer.fit_on_texts(df['text'])
df['text'] = tokenizer.texts_to_sequences(df['text'])
tokenizer.fit_on_texts(df['title'])
df['title'] = tokenizer.texts_to_sequences(df['title'])




X_train, X_test, y_train, y_test = train_test_split(df, labels)
X_train_title = X_train.pop('title')
X_test_title = X_test.pop('title')
X_train_text = X_train.pop('text')
X_test_text = X_test.pop('text')




X_train_title = return_pad_sequences(X_train_title, MAX_TITLE)
X_test_title = return_pad_sequences(X_test_title, MAX_TITLE)
X_train_text = return_pad_sequences(X_train_text, MAX_TEXT)
X_test_text = return_pad_sequences(X_test_text, MAX_TEXT)
```

Τελευταία διαδικασία για να ολοκληρωθεί η προ επεξεργασία είναι ο διαχωρισμός των δεδομένων σε σύνολο εκπαίδευσης και σύνολο δοκιμής. Όπως καταλαβαίνουμε και από τις ονομασίες τους, το σύνολο εκπαίδευσης θα είναι τα δεδομένα που θα τα χρησιμοποιήσουμε για να εκπαιδεύσουμε το δίκτυο μας και το σύνολο δοκιμής θα είναι τα δεδομένα που δεν θα έχει δει το δίκτυο, πολύ σημαντικό, για να κάνουμε αξιολόγηση. Υφίσταται η αναλογία να είναι 80% για το σύνολο εκπαίδευσης και 20% για το σύνολο δοκιμής, χωρίς να υπάρχει κάποιος αυστηρά δεσμευτικός όρος.

Προτείνουμε δύο τρόπους διαχωρισμού, με χρήση μια μεταβλητής (π.χ. traini_size) που υπολογίζεται από τον συνολικό αριθμό των δεδομένων και το μέγεθος που θέλουμε να γίνει ο διαχωρισμός (π.χ. training_portion = 0.8) και διαχωρίζουμε απο το μηδέν μέχρι την μεταβλητή και το υπόλοιπο από την μεταβλητή μέχρι το τέλος.

```
vocab_size = 12000
embedding_dim = 64
max_length = 200
trunc_type = 'post'
padding_type = 'post'
oov_tok = '<OOV>'
training_portion = .8

train_size = int(len(df) * training_portion)
labels = df.pop('label')
train_articles = articles[0: train_size]
train_labels = labels[0: train_size]

validation_articles = articles[train_size:]
validation_labels = labels[train_size:]
```

Αλλά μπορούμε και με χρήση της συνάρτησης train_test_split δίνοντας του σαν παραμέτρους τα δεδομένα και τους στόχους που θέλουμε.

```
X_train, X_test, y_train, y_test = train_test_split(df, labels)
X_train_title = X_train.pop('title')
X_test_title = X_test.pop('title')
X_train_text = X_train.pop('text')
X_test_text = X_test.pop('text')
```

6.2 RNN

Αφού κάνουμε την προ επεξεργασία των δεδομένων φτιάχνουμε ένα Βαθύ Νευρωνικό Δίκτυο με 4 επίπεδα, ένα Embedding, ένα Bidirectional και δύο Dense. [59]

```
model = tf.keras.Sequential([
    # Add an Embedding layer expecting input vocab of size 5000, and output embedding dimension of size 64 we set at the top
    tf.keras.layers.Embedding(vocab_size, embedding_dim),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(embedding_dim)),
    # use ReLU in place of tanh function since they are very good alternatives of each other.
    tf.keras.layers.Dense(embedding_dim, activation='relu'),
    # Add a Dense layer with 6 units and softmax activation.
    # When we have multiple outputs, softmax convert outputs layers into a probability distribution.
    tf.keras.layers.Dense(1, activation='sigmoid')
])
model.summary()
```

6.3 CNN

Εδώ παρουσιάζουμε το πιο υπολογιστικά βαρύ δίκτυο που δημιουργήσαμε. Δημιουργούμε στην αρχή δύο όμοια δίκτυα για τις δύο στήλες που θα χρησιμοποιήσουμε από το σύνολο δεδομένων μας, την title που περιέχει τους τίτλους των άρθρων και την text που περιέχει το συνολικό κείμενο τους. Στην συνέχεια τοποθετούμε ένα Embedding, ένα Convolution1D και ένα MaxPooling1D. Προσθέτουμε αυτά τα δύο στο τελικό μας και στην συνέχεια τα παρουσιάζουμε σε 3 Dense όπου το τελευταίο θα μας δίνει σαν έξοδο μια απάντηση, που αντιστοιχεί προφανώς στο αν είναι ψευδή ή αληθές το άρθρο.[57]

```
text_input = tf.keras.layers.Input(
    shape=(MAX_TEXT,), name='article_body_input')
text_embed = tf.keras.layers.Embedding(
    vocab_size + 1, 50, input_length=MAX_TEXT, name='article_body_embedding')(text_input)
text_conv = tf.keras.layers.Conv1D(
    256, 10, name='article_body_conv')(text_embed)
text_pool = tf.keras.layers.GlobalMaxPool1D(
    name='article_body_pooling')(text_conv)
title_input = tf.keras.layers.Input(
    shape=(MAX_TITLE,), name='article_title_input')
title_embed = tf.keras.layers.Embedding(
    vocab_size + 1, 50, input_length=MAX_TITLE, name='article_title_embedding')(title_input)
title_conv = tf.keras.layers.Conv1D(
    256, 3, name='article_title_conv')(title_embed)
title_pool = tf.keras.layers.GlobalMaxPool1D(
    name='article_title_pooling')(title_conv)
concat = tf.keras.layers.concatenate([text_pool, title_pool])
dense_100 = tf.keras.layers.Dense(100, activation='relu')(concat)
dense_50 = tf.keras.layers.Dense(50, activation='relu')(dense_100)
out_layer = tf.keras.layers.Dense(1, activation='sigmoid')(dense_50)
model = tf.keras.models.Model(
    inputs=[text_input, title_input], outputs=out_layer)
model.summary()
model.compile(optimizer=tf.keras.optimizers.Adam(0.0005),
              loss='binary_crossentropy', metrics=['accuracy'])
```

6.4 Αλγόριθμοι Κατηγοριοποίησης

Με ακριβώς ίδιο τρόπο και απλά κάνοντας διαφορετική κλήση στους αλγορίθμους μπορούμε εύκολα, γρήγορα και αρκετά αποδοτικά να έχουμε αποτελέσματα σε ένα πρόβλημα.[58]

```
# DataFlair - Initialize a PassiveAggressiveClassifier
pac = PassiveAggressiveClassifier(max_iter=70)
pac.fit(tfidf_train, y_train)
# DataFlair - Predict on the test set and calculate accuracy
y_pred = pac.predict(tfidf_test)
score = accuracy_score(y_test, y_pred)
print(f'Accuracy: {round(score*100,2)}%')

"""
# DataFlair - Build confusion matrix
confusion_matrix(y_test, y_pred, labels=['FAKE', 'REAL'])
"""
```

Καλούμε τον αλγόριθμο Support Vector Machine όπου η παράμετρος C είναι είναι ένα υπέρμετρο που έχει ρυθμιστεί πριν από το εκπαιδευτικό μοντέλο και χρησιμοποιείται για τον έλεγχο του σφάλματος, η παράμετρος degree είναι ο βαθμός του πολυωνύμου που χρησιμοποιείται για την εύρεση του hyperplane για το διαχωρισμό των δεδομένων, η Gamma είναι επίσης ένα υπέρμετρο που τίθεται πριν από το εκπαιδευτικό μοντέλο και χρησιμοποιείται για να δώσει το βάρος καμπυλότητας του ορίου απόφασης και η kernel είναι η συνάρτηση που παίρνει τα δεδομένα σαν είσοδο και τα μετατρέπει στην απαραίτητη μορφή.

```
27
28
29  model = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
30  model.fit(x_train, y_train)
31
```

Καλούμε τον αλγόριθμο Naïve Bayes στην μορφή Gauss.

```
27
28   model = GaussianNB()
29   model.fit(x_train, y_train)
30
31
```

6.5 Σύγκριση αποτελεσμάτων

Αλγόριθμος	Αποτέλεσμα
NAÏVE BAYES	76,99%
PASSIVE AGGRESSIVE	83.35%
SUPPORT VECTOR MACHINE	86.67%
RNN	94%
CNN	90%

Σύμφωνα με τα αποτελέσματα αλλά και όπως ήταν αναμενόμενο τα βαθιά νευρωνικά (CNN-RNN) δίκτυα έδωσαν καλύτερα αποτελέσματα από ότι οι κατηγοριοποιητές. Από τους οποίους την μικρότερη τιμή μας την έδωσε ο NB, ακολούθως ο PA και ο SVM. Στα δύο νευρωνικά δίκτυα πιο αποδοτικό κατά 4% ήταν το RNN δίκτυο με 94% με το CNN να μας δίνει εξίσου καλά αποτελέσματα στο 90%.

Μπορούμε εύκολα να κατανοήσουμε τον λόγο της αύξησης της φήμης των Βαθέων Νευρωνικών Δικτύων αφού στις περισσότερες περιπτώσεις που θα συναντήσουμε, με τις κατάλληλες παραμετροποιήσεις μας δίνουν τα καλύτερα αποτελέσματα.

6.6 Μετατροπή σε Tensorflow Lite

Ο τρόπος με τον οποίο μας δίνει την δυνατότητα η Tensorflow να χρησιμοποιήσουμε ένα μοντέλο σε κάποια edge συσκευή, όπως στην δική μας περίπτωση στο Raspberry, γίνεται μέσω μιας διαδικασίας αυτοματοποιημένης και αρκετά εύκολης. Καλούμε τον μετατροπέα είτε κατευθείαν από ένα μοντέλο keras που έχουμε εκπαίδευση αλλά μπορούμε και σε μοντέλα που τα έχουμε εκπαίδευση και τα έχουμε εξάγει και αποθηκεύσει σε μορφή pb.[63]

```
129
130
131   converter = tf.lite.TFLiteConverter.from_keras_model(model)
132   tflite_model = converter.convert()
133
```

Και στην συνέχεια το αποθηκεύομε στην νέα πλέον μορφή model.tflte ή κάνοντας απλά αποθήκευση το μοντέλο μας μπορούμε να το μετατρέψουμε αργότερα. Αυτός είναι γενικά ένας τρόπος ώστε να αποθηκεύομε την εκπαίδευση ενός μοντέλου.[63]

```
117
118   model.save('my_model')
119
120   new_model = tf.keras.models.load_model('saved_model/my_model')
121
122
123
```

```
model.save(tflite_model)

new_model = tf.keras.models.load_model('model.tflite')
```

Και για να δούμε ότι όλα όσα αναφέραμε ισχύουν εκτελούμε το παρακάτω πρόγραμμα στην πλατφόρμα που θέλουμε.[60]

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

TEST_CASES = 10


interpreter = tf.lite.Interpreter(model_content=tflite_model)
interpreter.allocate_tensors()
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()


prediction = []
for i in range(len(preds)):
    if preds[i].item() > 0.95:
        prediction.append(1)
    else:
        prediction.append(0)


accuracy = accuracy_score(list(validation_label_seq), prediction)

print("Model Accuracy : ", accuracy)
```

7. ΣΥΜΠΕΡΑΣΜΑΤΑ

Καταφέραμε αποδοτικά να μελετήσουμε τις δυνατότητες των βαθέων νευρωνικών δικτύων και να δείξουμε μέσω τον αποτελεσμάτων την αξιοπιστία τους σε σύγκριση με πιο συνηθισμένους αλγορίθμους κατηγοριοποίησης. Εμβαθύναμε στις δυνατότητες μέσω των συναρτήσεων που μας παρέχει η βιβλιοθήκη Tensorflow και πως μπορούμε να τις χρησιμοποιήσουμε για να κατασκευάσουμε και να κατανοήσουμε τον τρόπο λειτουργίας και ανάπτυξης μοντέλων νευρωνικών δικτύων . Επίσης, δείξαμε τον τρόπο που συνεργάζεται η Keras με άλλες βιβλιοθήκες και διαχωρίσαμε πως πιο αποδοτικό τρόπο αποτελεί η σύμπτυξη δύο ή περισσότερων βιβλιοθηκών ανάλογα με τον στόχο που θέτουμε. Αναλύσαμε το πρόβλημα της παραπληροφόρησης και εξάγαμε αποτελέσματα αποδοτικότητας των νευρωνικών δικτύων και των αλγορίθμων κατηγοριοποίησης πάνω σε σύνολο δεδομένων τους θέματος αυτού. Τέλος, καταφέραμε να τα μετατρέψουμε σε κατάλληλα αρχεία που θα εκτελεστούν από πλατφόρμες περιορισμένου hardware, όπως το Raspberry και να δείξουμε ότι με μικρό οικονομικό κόστος μπορεί κάποιος να αναλύσει όλα τα παραπάνω και να τα μελετήσει.

7.1 Αντικείμενο προς μελέτη

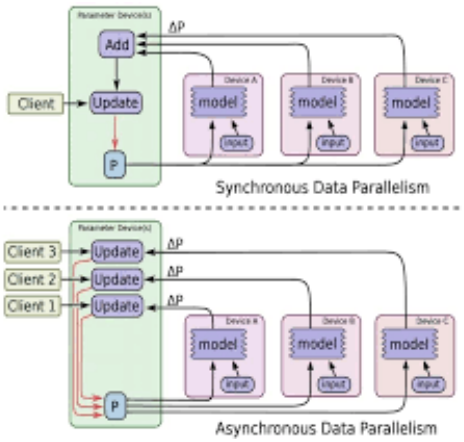
Αναφέραμε νωρίτερα πως ο κλάδος της Τεχνητής Νοημοσύνης είχε εμφανισθεί πολύ νωρίτερα αλλά λόγο της έλλειψης δεδομένων δεν μπορούσε να μελετηθεί σε βάθος. Με το πέρασμα των χρόνων ο όγκος των δεδομένων αυξήθηκε ραγδαία σε σημείο να εμφανισθεί η μελέτη ανάλυσης μεγάλων δεδομένων (Bid Data Analysis), που καθίσταται ολοένα και πιο απαραίτητη για πολλούς τομείς της επιστήμης. Συχνά, το τεράστιο μέγεθος των δεδομένων αυτών συχνά οδηγεί σε χαμηλό χρόνο εκτέλεσης με δύσκολους υπολογιστικά αλγόριθμους. Μια άλλη δυσκολία που προκύπτει είναι όταν το σύνολο δεδομένων είναι μεγαλύτερο από τη διαθέσιμη μνήμη RAM σε ένα υπολογιστικό σύστημα. Στην περίπτωση αυτή, θα χωριστεί το σύνολο δεδομένων σε μικρότερα και σε διαδικασίες ξεχωριστά, κάτι που συνεπάγεται ακόμη μεγαλύτερο χρόνο επεξεργασίας. Καθώς τα σύνολα αυξάνονται συνεχώς ο νόμος του Moore πλησιάζει τα όριά του [67], οι κλασσικοί

αλγόριθμοι τείνουν να είναι αρκετά αναποτελεσματικοί για να χειριστούν τις αναλυτικές αναλύσεις τελευταίας τεχνολογίας.

Μια εύλογη λύση που προτείνετε από τους μελετητές για την βελτίωση της αποτελεσματικότητας της επεξεργασίας είναι η ανάπτυξη παράλληλων και κατανεμημένων αλγόριθμων-συστημάτων όπως η κατανεμημένη μνήμη Fourier, distributed-memory Fast Fourier Transform [68] και η παράλληλη μετεγκατάσταση αντιστροφής ώρας, parallel Reverse Time Migration [69]. Συνεπώς ο παράλληλος και κατανεμημένος υπολογιστής γίνεται όλο και περισσότερο αντικείμενο προς μελέτη και μια οικονομική και αρκετά αξιόπιστη λύση θα ήταν η χρήση δικτύου Raspberry.

Το ενδιαφέρον για την κατασκευή ενός τέτοιο δικτύου έχει αυξηθεί και αυτό με την σειρά αρκετά τα τελευταία χρόνια αφού αποτελεί φυσικό επόμενο, όταν μπορεί να κατασκευαστεί ένα υπερ υπολογιστής με τη βοήθεια φθηνών ηλεκτρονικών εξαρτημάτων. Ένα σύμπλεγμα μπορεί να δημιουργηθεί με πολλούς διαφορετικούς τρόπους, με λίγους εώς εκατοντάδες τέτοιος κόμβους.[71]

Για να παραλληλιστεί η εκπαίδευση ενός NN, χρησιμοποιείται συνήθως παραλληλισμός δεδομένων. Αυτό το είδος παραλληλισμού εκδηλώνεται στο ότι πολλαπλές εργασίες (tasks) σε μια εργασία εργαζομένων μπορούν να εκπαιδεύσουν το ίδιο μοντέλο σε μικρές παρτίδες δεδομένων και να ενημερώσουν τις κοινές παραμέτρους στις εργασίες της εργασίας του βασικού worker. [70]
Συνεπώς ένα δίκτυο μπορεί να εκπαιδευτεί με δύο τρόπους, έναν σύγχρονο (sychronous) και έναν ασύγχρονο (asychronous). Η ασύγχρονη είναι η πιο συνηθισμένη από τις δύο μέθοδος εκπαίδευσης, η σύγχρονη χρησιμοποιείται κυρίως όταν όλα τα αντίγραφα γραφημάτων διαβάζουν ως είσοδο από το ίδιο σύνολο τρεχόντων τιμών παραμέτρων. Στην συνέχεια οι κλίσεις υπολογίζονται παράλληλα και τελικά εφαρμόζονται μαζί πριν ξεκινήσει ο επόμενος κύκλος εκπαίδευσης. Η ασύγχρονη εκπαίδευση χρησιμοποιείται όταν κάθε αντίγραφο του γραφήματος έχει επαναληπτικό κύκλο εκπαίδευσης, η οποίοι είναι ανεξάρτητοι ο ένας από τον άλλο και εκτελούνται ασύγχρονα. [70]



Εικόνα 23: Σύγχρονος και ασύγχρονος τρόπος εκπαίδευσης παράλληλων δεδομένων

7.2 Προβλήματα που αντιμετωπίστηκαν

Ένα από τα βασικότερα προβλήματα που αντιμετωπίστηκε κατά την διάρκεια υλοποίησης της πτυχιακής αυτής ήταν ο τρόπος εγκατάστασης της βιβλιοθήκης Tensorflow στην πλατφόρμα Raspberry. Σε αρκετά αρχικό στάδιο συνειδητοποιήσαμε ότι με τον προτεινόμενο τρόπο της βιβλιογραφίας του TensorflowLite αντιμετώπιζε την παρούσα χρονική στιγμή πρόβλημα, αφού όπως προτείνετε με αρκετά απλό τρόπο η εγκατάσταση, κάνει αυτόματο έλεγχο τους συστήματος και εγκαθιστάτε η κατάλληλη έκδοση, όμως στην δική μας περίπτωση δεν λειτούργησε για αυτό κάναμε έναν πιο πολύπλοκο τρόπο όπου εγκαταστήσαμε όλα τα απαραίτητα dependencies ξεχωριστά.

Στην συνέχεια επειδή η μετατροπή μοντέλων βαθέων νευρωνικών δικτύων μέσω του μετατροπέα είναι σε πρώιμο στάδιο, χρειαζόταν περισσότερη εμβάθυνση από ότι φαινόταν γιατί τα δεδομένα πρέπει να παρουσιάζονται με συγκεκριμένο τρόπο ώστε να επιτευχθεί σωστά η μετατροπή και να λειτουργήσει το μοντέλο στην συνέχεια στο Raspberry.

8. ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] *"index | TIOBE - The Software Quality Company". www.tiobe.com. Retrieved 2 February 2021.* Python has won the TIOBE programming language of the year award! This is for the fourth time in the history, which is a record! The title is awarded to the programming language that has gained most popularity in one year.

[2] *[a] [b] Venners, Bill (13 January 2003). "The Making of Python". Artima Developer. Artima. Retrieved 22 March 2007.*

[3] *Peterson, Benjamin (20 April 2020). "Python Insider: Python 2.7.18, the last release of Python 2". Python Insider. Retrieved 27 April 2020.*

[4] https://github.com/pypa/pip/issues/6148

Accessed: 2021-02-22.

[5] *"Python Developer's Guide — Python Developer's Guide". devguide.python.org. Retrieved 17 December 2019.*

[6]*"History and License". Retrieved 5 December 2016.* "All Python Releases are Open Source"

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* 12, 2825.

Pereira, F., Mitchell, T., and Botvinick, M. (2009). Machine learning classifiers and fMRI: a tutorial overview. *Neuroimage* 45, S199–S209. doi: 10.1016/j.neuroimage.2008.11.007

[8]V. Michel, A. Gramfort, G. Varoquaux, E. Eger, C. Keribin, and B. Thirion. A supervised clusteringapproach for fMRI-based inference of brain states.Patt Rec, page epub ahead of print, April2011. doi: 10.1016/j.patcog.2011.04.006

[9] Guide to NumPyTravis E. Oliphant, PhDDec 7, 2006

[10] *Charles R Harris; K. Jarrod Millman; Stéfan J. van der Walt; et al. (16 September 2020). "Array programming with NumPy" (PDF). Nature. 585 (7825): 357–362. doi:10.1038/S41586-020-2649-2. ISSN 1476-4687. PMID 32939066. Wikidata Q99413970.*

[11] pandas: a Foundational Python Library for DataAnalysis and StatisticsWes McKinney

[12] *"NumFOCUS – pandas: a fiscally sponsored project". NumFOCUS. Retrieved 3 April 2018.*

[13] Deep learning with Python

F Chollet - 2018 - silverio.net.br

[14] Deep Learning With Python: Develop Deep Learning Models on Theano and Tensorflow Using Keras

By Jason Brownlee

[15] Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras

R Shanmugamani - 2018 - books.google.com

[16] Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems

A Géron - 2019 - books.google.com

[17] TensorFlow: A System for Large-Scale Machine LearningMartín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, Google Brain November 2–4, 2016 • Savannah, GA, USAISB N 978 -1- 931971-33 -1

[18] Deep learning with Keras

A Gulli, S Pal - 2017 - books.google.com

[19] An introduction to deep learning and keras

J Moolayil - Learn Keras for Deep Neural Networks, 2019 – Springer

[20] Deep learning with Python

F Chollet - 2018 - silverio.net.br

[21]Google Inc.Tensorflow Text Classification. https://github.com/tensorflow/examples/tree/master/lite/examples/text_classification/android. Accessed: 2021-02-22.

Accessed: 2021-02-22.

[22] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko,Weijun Wang, Tobias Weyand, Marco Andreetto, and HartwigAdam. “MobileNets: Efficient Convolutional Neural Networksfor Mobile Vision Applications”. In:CoRRabs/1704.04861 (2017).arXiv:1704.04861.

[23] Google Inc. Tensorflow Lite. https://www.tensorflow.org/mobile/tflite/.

Accessed: 2021-02-22.

[24] Andrej Karpathy.CS231n Convolutional Neural Networks for Visual Recognition.http://cs231n.github.io/convolutional-networks/. Accessed: 2021-02-22.

[25]https://medium.com/hackernoon/challenges-in-deep-learning-57bbf6e73bb

Accessed: 2021-02-22.

[26]http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Accessed: 2021-02-22.

[27]https://www.cs.cornell.edu/courses/cs1114/2013sp/sections/S06_convolution.pdf

Accessed: 2021-02-22.

[28]https://medium.com/hackernoon/challenges-in-deep-learning-57bbf6e73bb

Accessed: 2021-02-22.

[29] https://wiki.pathmind.com/neural-network

Accessed: 2021-02-22.

[30]https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/

Accessed: 2021-02-22.

[31]https://www.hackster.io/news/passive-aggressive-classifier-for-embedded-devices-f97c3461fbee

Accessed: 2021-02-22.

[32] https://medium.com/swlh/demystifying-support-vector-machine-part-i-b5b083844c9a

Accessed: 2021-02-22.

[33]https://en.wikipedia.org/wiki/Naive_Bayes_classifier

Accessed: 2021-02-22.

[34]https://www.astroml.org/book_figures/chapter9/fig_simple_naivebayes.html

Accessed: 2021-02-22.

[35]https://blog.clairvoyantsoft.com/mlmuse-naivety-in-naive-bayes-classifiers-9c7f6ba952bf

Accessed: 2021-02-22.

[36] (https://el.wikipedia.org/wiki/%CE%A4%CE%B5%CF%87%CE%BD%CE%B7%CF%84%CE%AE_%CE%BD%CE%BF%CE%B7%CE%BC%CE%BF%CF%83%CF%8D%CE%BD%CE%B7)

Accessed: 2021-02-22.

[37] https://www.europarl.europa.eu/news/el/headlines/society/20200827STO85804/ti-einai-i-techniti-noimosuni-kai-pos-chrisimopoieitai

Accessed: 2021-02-22.

[38]https://el.wikipedia.org/wiki/Μηχανική_μάθηση

Accessed: 2021-02-22.

[39]https://www.ibm.com/blogs/systems/ai-machine-learning-and-deep-learning-whats-the-difference/

Accessed: 2021-02-22.

[40]https://towardsdatascience.com/a-basic-introduction-to-tensorflow-lite-59e480c57292

Accessed: 2021-02-22.

[41] Research ArticleFake News Detection Using Machine Learning Ensemble MethodsIftikhar Ahmad,1Muhammad Yousaf,1Suhail Yousaf,1and Muhammad Ovais Ahmad2

Received 4 September 2020; Revised 14 September 2020; Accepted 16 September 2020; Published 17 October 202

[42] https://towardsdatascience.com/fake-news-detector-with-deep-learning-approach-part-ii-modeling-42b9f901b12b

Accessed: 2021-02-22

[43]https://el.wikipedia.org/wiki/%CE%95%CE%BE%CF%8C%CF%81%CF%85%CE%BE%CE%B7_%CE%B4%CE%B5%CE%B4%CE%BF%CE%BC%CE%AD%CE%BD%CF%89%CE%BD

Accessed: 2021-02-22

[44] Data mining

DJ Hand, NM Adams - Wiley StatsRef: Statistics Reference, 2014 - Wiley Online Library

[45]
IANH.WITTEN,E.F.(2017).DataMining:practicalmachinelearningtoolsandtechniques, Fourth Edition

[46]
ΜΑΝΩΛΟΠΟΥΛΟΣ,Ν.Α.(2008).Εισαγωγήστηνεξόρυξηκαιτιςαποθήκεςδεδομένων.ΕΚΔΟΣΕΙΣ ΝΕΩΝ ΤΕΧΝΟΛΟΓΙΩΝ. ISBN

[47]
Sebastiani,F.(2001).MachineLearninginAutomatedTextCategorization. ACMComputingSurveys (CSUR)

[48] https://www.sciencedirect.com/topics/computer-science/text-mining

Accessed: 2021-02-20

[49] https://towardsdatascience.com/using-deep-learning-for-end-to-end-multiclass-text-classification-39b46aecac8

Accessed: 2021-02-22

[50] Handbook of natural language processing

N Indurkhya, FJ Damerau - 2010 - books.google.com

[51] Natural language processing

ED Liddy - 2001 - surface.syr.edu

[52] https://github.com/KaiDMML/FakeNewsNet.git

[53] https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset

[54] https://www.kaggle.com/samrat96/fake-news-detection

[55] FakeNewsNet: A Data Repository with News Content, Social Context and Spatialtemporal Information for Studying Fake News on Social Media

Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, Huan Liu

[56] https://developer.twitter.com/en/portal/petition/use-case

[57] https://medium.com/better-programming/how-to-use-artificial-intelligence-and-twitter-to-detect-fake-news-a-python-tutorial-75a4132acf7f

[58] https://qengineering.eu/install-tensorflow-2.1.0-on-raspberry-pi-4.html

[59] https://www.tensorflow.org/tutorials/keras/text_classification

[60] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.PassiveAggressiveClassifier.html

[61] https://scikit-learn.org/stable/modules/svm.html

[62] https://scikit-learn.org/stable/modules/naive_bayes.html

[63] https://www.tensorflow.org/lite/tutorials/model_maker_image_classification

[64] https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn

[65] https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34

[66] Yilmaz, O. (2001). Seismic data analysis: processing, inversion and interpretation of seismic data.SEG.

[67] Kumar, S. (2015). Fundamental Limits to Moore's Law.arXiv:1511.05956

[68] Araya-Polo, M., Cabezas, J., Hanzich, M., & Pericas, M. (2011). Assessing accelerator-based HPC reverse time migration. IEEE Transactions on Parallel and Distributed Systems, 22 (1), 147-162.

[69] Gholami, A., Hill, J., Malhotra, D., & Biros, G. (2016). AccFFT: A library for distributed-memory 3-D FFT on CPU and GPU architectures.arXiv:1506.07933

[70] Ellen-Louise Bleeker,Magnus Reinholdsson (2017) Creating a Raspberry Pi-Based BeowulfCluster

[71] Joshua Kiepert. Creating a raspberry pi-based beowulf cluster.Boise State, 2013

[72]Distributed TensorFlow documentation. https://www.tensorflow.org/deploy/distributed.

Accessed: 2021-02-12

[73] https://digitalcommons.mtech.edu/cgi/viewcontent.cgi?article=1198&context=grad_rsch

Accessed: 2021-02-12