ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΜΕΛΕΤΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ DEEP LEARNING ΤΕΧΝΙΚΩΝ ΣΤΟΝ ΤΟΜΕΑ ΤΗΣ ΑΥΤΟΚΙΝΗΣΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βαλουξής Δημήτρης

Δράκος Γιώργος

ΕΠΙΒΛΕΠΟΝΤΕΣ ΚΑΘΗΓΗΤΕΣ

Τσουκαλάς Ελευθέριος, Καθηγητής

Χαρίκλεια Τσαλαπάτα, Ε.ΔΙ.Π

Βόλος, 2018

UNIVERSITY OF THESSALY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

RESEARCH AND IMPLEMENTATION OF DEEP LEARNING TECHNIQUES IN THE AUTOMOBILE AREA

DIPLOMA THESIS

Valouxis Dimitrios

Drakos Georgios

SUPERVISORS

Eleftherios Tsoukalas, Professor

Harikleia Tsalapata, E.DI.P

Volos, 2018

Στις οικογένειές μας,

Ευχαριστίες

Με την ολοκλήρωση της παρούσας διπλωματικής εργασίας, θα θέλαμε να ευχαριστήσουμε θερμά, τον επιβλέποντα καθηγητή μας, κύριο Τσουκαλά Ελευθέριο και την συνεπιβλέπουσα καθηγήτρια μας Τσαλαπάτα Χαρίκλεια για την υποστήριξη τους.

Επιπλέον, θα θέλαμε να ευχαριστήσουμε θερμά τον κύριο Καρκάνη Σταύρο, καθηγητή του τμήματος Μηχανικών Πληροφορικής Τ.Ε.Ι Στερεάς Ελλάδας, για την παραχώρηση των οδηγικών δεδομένων, καθώς και για τις πολύτιμες επιστημονικές του γνώσεις, που ήταν καθοριστικές για την ολοκλήρωση της παρούσας διπλωματικής. Επίσης θα θέλαμε να ευχαριστήσουμε τον κύριο Σαβελώνα Μιχάλη επίκουρο καθηγητή στο τμήμα Μηχανικών Πληροφορικής στο Τ.Ε.Ι. Στερεάς Ελλάδας και τον κύριο Σπύρου Ευάγγελο επίκουρο καθηγητή στο τμήμα Μηχανικών Πληροφορικής στο Τ.Ε.Ι. Στερεάς Ελλάδας.

Ιδιαίτερα θα θέλαμε να ευχαριστήσουμε τον Δρόσο Αναστάσιο, Βαλουξή Φώτη, Αλεξίου Μαρία και τον Κυριάκου Δημήτριο για την ψυχολογική συνεισφορά που μας προσέφεραν σε όλο αυτό το διάστημα ανάπτυξης της παρούσας εργασίας.

Τέλος, θα θέλαμε να ευχαριστήσουμε την οικογένειά μας και τους φίλους μας, που είναι δίπλα μας όλα αυτά τα χρόνια και μας στηρίζουν αδιάκοπα.

Περίληψη

Στην παρούσα διπλωματική εργασία, το ζήτημα που τίθεται προς αντιμετώπιση είναι η αξιολόγηση της οδηγικής συμπεριφοράς του οδηγού. Ο τρόπος με τον οποίο επιλέγουμε να προσεγγίσουμε το συγκεκριμένα θέμα είναι με τη χρήση της μηχανικής μάθησης και πιο συγκεκριμένα με το deep learning και την εκμετάλλευση των νευρωνικών δικτύων. Βασικός στόχος μας είναι, έχοντας λάβει δεδομένα από διαφορετικά είδη οχήματος (ΙΧ, φορτηγό, σκούπα) να υλοποιήσουμε την δυαδική κατηγοριοποίηση του οχήματος με βάση τον τρόπο οδήγησης. Δίχως όμως να μένουμε μόνο σε αυτή την περίπτωση αναπτύξαμε περαιτέρω το νευρωνικό μας δίκτυο, ώστε να μπορεί να προσαρμοστεί σε διαφορετικές εισόδους και να εξάγει διαφορετικά συμπεράσματα ανάλογα με το τί θέλουμε να προβλέψει το μοντέλο. Παράλληλα δημιουργήθηκαν από τα ήδη υπάρχοντα δεδομένα, χαρακτηριστικά δηλαδή νέα δεδομένα με σκοπό την αύξηση της αποδοτικότητας και ακρίβειας του νευρωνικού δικτύου. Επιπρόσθετα, έγιναν οι κατάλληλοι έλεγχοι για την εύρεση συσχέτισης μεταξύ των δεδομένων που λειτουργούν έως είσοδοι στο νευρωνικό και των αποτελεσμάτων που παράγει. Ολοκληρώνοντας, είναι σημαντικό να τονιστεί πως ο συγκεκριμένος κλάδος της μηχανικής μάθησης απασχολεί όλο και περισσότερο στις ημέρες μας και εφαρμόζεται σε πληθώρα επιστημονικών πεδίων.

Abstract

In this diploma thesis, the issue to be addressed is to evaluate the driving behavior of the driver. The way we choose to approach this subject is through the use of mechanical learning and more specifically through deep learning and exploitation of neural networks. Our main goal is to get the binary classification of the vehicle based on driving mode, having received data from different types of vehicle (LC, truck, vacuum). But without living only in this case we further developed our neural network so that it can adapt to different inputs and draw different conclusions depending on what the model is intended to predict. At the same time they were created from the existing data, namely new data in order to increase the efficiency and accuracy of the neural network. Additionally, appropriate checks were made to find a correlation between the data that works to input to the neuronal and the results it produces. In conclusion, it is important to emphasize that this particular field of engineering learning is becoming more and more important in our days and is applied in a variety of scientific fields.

Πίνακας Περιεχομένων

Κεφάλαιο 1

Εισαγωγή

Στο παρόν κεφάλαιο, αναλύεται συνοπτικά ο σκοπός της παρούσας εργασίας. Επίσης γίνεται μια νίξη στο τι έχει επιτευχθει εώς σήμερα στον τομέα της αυτοκινητοβιομηχανίας και της σχέσης της με τη μηχανική μάθηση. Τελικά, δίνεται έμφαση στην προσέγγιση που ακολουθήθηκε και τη διαφοροποίηση της με τις υπάρχουσες.

1.1 Σκοπός της Εργασίας

Σκοπός της εργασίας μας είναι η αξιολόγηση της οδηγικής συμπεριφοράς με διαφορετική προσέγγιση. Συγκεκριμένα, εστιάζουμε την προσέγγιση μας αρχικά στην εξαγωγή νέων χαρακτηριστικών , με πρωτότυπο τρόπο, από τα ήδη υπάρχοντα δεδομένα που έχουμε από το όχημα και στην συνέχεια στην προσαρμογή του μοντέλου μας σε διαφορετικά οδηγικά σενάρια. Επίσης κάτι που αξίζει να σημειωθεί, είναι η ανάλυση συσχέτισης που αναπτύχθηκε με σκοπό την εύρεση συσχέτισης μεταξύ των δεδομένων που λειτουργούν ως είσοδοι στο δίκτυο και στο τελικό αποτέλεσμα που παράγεται από το δίκτυο μας. Η εν λόγω ανάλυση δίνει την δυνατότητα περαιτέρω βελτίωσης του δικτύου που οδηγεί σε βελτιστοποίηση των προβλέψεων. Παράλληλα, η πρόσληψη των δεδομένων μας πραγματοποιείται σε πραγματικό χρόνο και σε περιβάλλοντα πραγματικών συνθηκών. Αυτό οδηγεί σε μεγαλύτερη εγκυρότητα και αξιοπιστία αφού το μοντέλο έχει δομηθεί και δοκιμαστεί μέσα από πραγματικά δεδομένα και όχι σε πειραματικά και προσεγγιστικά στοιχεία όπως είθισται.

1.2 Τι έχει επιτευχθεί έως Σήμερα

Σήμερα, στον ευρύτερο κλάδο της αυτοκινητοβιομηχανίας πραγματοποιούνται σημαντικές επενδύσεις με έμφαση στην τεχνητή νοημοσύνη για την βελτιστοποίηση της αυτόνομης οδήγησης. Συγκεκριμένα η Uber δοκιμάζει ήδη αυτόνομα ταξί [1], η Tesla εξελίσσει περαιτέρω το σύστημα Autopilot [2] και η Google τρέχει προγράμματα ανάπτυξης αυτόνομων οχημάτων [3]. Στο μέλλον, η τεχνολογία αυτόνομων οχημάτων θα φέρει ριζικές αλλαγές τόσο στο ίδιο προϊόν – το όχημα – όσο και στο ευρύτερο οικοσύστημα.

Ένας σημαντικός τομέας όπου η τεχνητή νοημοσύνη βρίσκει εφαρμογές τα τελευταία χρόνια, είναι ο τομέας της αξιολόγησης της οδηγικής συμπεριφοράς. Με τον όρο οδηγική συμπεριφορά νοείται ένας συνοπτικός όρος ο οποίος χρησιμοποιείται για να αναπαρασταθούν διαφορετικά πλάνα τα οποία σχετίζονται με οδηγικές ενέργειες και τους τρόπους οδήγησης στον οποίο όρο έχουμε άπειρες συσχετιζόμενες μεταβλητές. Υπάρχουν αρκετές εφαρμογές που έχουν υλοποιηθεί για τον προσδιορισμό της οδηγικής συμπεριφοράς των οδηγών[4]. Στη συνέχεια θα περιγράφουμε μια σχετική εφαρμογή για την πλήρη κατανόηση της.

Αρχικά η κατηγοριοποίηση(πλάνα) των οδηγών ορίζεται ως εξής:

1. Ασφαλής οδήγηση
2. Επιθετική οδήγηση
3. Απρόσεκτη οδήγηση
4. Οδήγηση υπό την επήρεια αλκοόλ

Συνεπώς η τεχνητή νοημοσύνη θα χρησιμοποιηθεί για να γίνει η κατηγοριοποίηση των οδηγών με βάση τα δεδομένα της οδήγησης τους. Τα δεδομένα που χρειαζόμαστε σχετίζονται με την επιτάχυνση, την ταχύτητα, την μορφολογία του εδάφους, το στρίψιμο του τιμονιού και την εμπέδηση. Τροφοδοτώντας το δίκτυο τεχνητής νοημοσύνης με τα δεδομένα, το δίκτυο "αποφασίζει" σε ποια από τις τέσσερις κατηγορίες ανήκει ο οδηγός.

1.3 Η προσέγγιση μας

Σε αντίθεση με τις έως τώρα προσεγγίσεις οι οποίες παρουσιάζουν ως βασικό όχημα το ΙΧ για την αξιολόγηση της οδηγικής συμπεριφοράς, η παρούσα εργασία καλείται να υλοποιήσει την αξιολόγηση ερευνώντας ταυτόχρονα διαφορετικά είδη οχήματος όπως ΙΧ, φορτηγό, σκούπα και απορριμματοφόρο. Η μέθοδος αυτή ισχυροποιεί το μοντέλο καθώς του δίνει την δυνατότητα να κατηγοριοποιήσει ένα ευρύτερο φάσμα οχημάτων αλλά και να ταυτίσει ένα όχημα με μια οδηγική συμπεριφορά όπως για παράδειγμα την οδήγηση απορριμματοφόρου μέσα στην πόλη με την οδήγηση ενός επιθετικού οδηγού ΙΧ..

Κεφάλαιο 2

Ο Τομέας της Τεχνητής Νοημοσύνης

Στο παρόν κεφάλαιο θα γίνει εισαγωγή στις έννοιες της τεχνητής νοημοσύνης, μηχανικής μάθησης (machine learning), βαθιάς μάθησης (deep learning), νευρωνικών δικτύων καθώς και στην σχέση μεταξύ τους. Παράλληλα θα καταγραφούν σύγχρονα παραδείγματα χρήσης τεχνητής νοημοσύνης και θα γίνει ανάλυση της σύγχρονης μορφής της.

2.1 Τεχνητή Νοημοσύνη

Η τεχνητή νοημοσύνη (AI) εξελίσσεται για δεκαετίες και αποτελεί ακόμα και σήμερα ένα από τα πιο σημαντικά ζητήματα στον τομέα του Computer Science. Αυτό συμβαίνει αφού το αντικείμενο εκτείνεται σε πάρα πολλά θέματα ξεκινώντας από κάτι πολύ απλό , όπως αλγόριθμοι για να παιχτούν επιτραπέζια παιχνίδια και φτάνοντας σε πολύπλοκες εφαρμογές που χρησιμοποιούμε καθημερινά στις σύγχρονες κοινωνίες.

Πρακτικά τεχνητή νοημοσύνη είναι : "Υπολογιστικά συστήματα με ανθρώπινη συμπεριφορά". Αφού πιστεύουμε ότι οι άνθρωποι είναι ευφυείς, πρέπει να φτιάξουμε μοντέλα με έξυπνη συμπεριφορά. Υπάρχει μία εξαιρετική έρευνα από τον Turing ο οποίος πρότεινε την ιδέα να φτιαχτούν μοντέλα με ανθρώπινη συμπεριφορά. Η συγκεκριμένη επιστήμη μπορεί να προσεγγιστεί με ποικίλους τρόπους. Αρχικά, μια ιδέα θα ήταν να γίνουν πειράματα πάνω σε ανθρώπους , να δούμε πως συμπεριφέρονται υπό κάποιες συνθήκες και να προσπαθήσουμε να κάνουμε τους υπολογιστές να συμπεριφέρονται με τον ίδιο τρόπο. Φανταστείτε ένα πρόγραμμα που θα έπαιζε πόκερ. Αντί να φτιάχναμε το πρόγραμμα που θα έπαιζε πόκερ με τον καλύτερο δυνατό τρόπο , θα φτιάχναμε ένα που θα έπαιζε όπως κάποιος άνθρωπος.

Μία άλλη προσέγγιση θα ήταν να φτιάχναμε μοντέλα με ανθρώπινο τρόπο σκέψης. Δεν αρκεί να δημιουργήσουμε προγράμματα που θα μοιάζουν να συμπεριφέρονται όπως οι άνθρωποι, αλλά να συμπεριφέρονται πραγματικά με τον ίδιο τρόπο. Η στρατηγική έρευνας είναι να υιοθετήσουμε τη διαδικασία σκέψης των ανθρώπων και στη συνέχεια να κατασκευάσουμε υπολογιστικά μοντέλα που αντανακλούν αυτόν τον τρόπο σκέψης. Ένα σημαντικό ζήτημα είναι να αποφασίσουμε μέχρι ποιο στάδιο θα υιοθετήσουμε αυτό που βρίσκεται στη σκέψη κάθε ανθρώπου. Κάποιος μπορεί να προσπαθήσει να το μοντελοποιήσει σε πολύ υψηλό επίπεδο, για παράδειγμα, διαιρώντας την επεξεργασία σε οθόνες υψηλού επιπέδου, μνήμη και γνωστικές λειτουργίες. Έπειτα θα προσπαθήσουν να

καταστήσουν ακριβή την λειτουργικότητα αλλά δε θα ανησυχούν πάρα πολύ για τον τρόπο υλοποίησης των ενοτήτων.

Ορισμένοι ερευνητικοί χώροι που δραστηριοποιείται η τεχνητή νοημοσύνη με στόχο την δημιουργία πιο ευφυών συστημάτων και μηχανών είναι οι εξής:

1. Η Ρομποτική. Ο συγκεκριμένος τομέας ασχολείται το σχεδιασμό , τη λειτουργία και τον έλεγχο ρομποτικών συστημάτων δηλαδή μηχανών που μπορούν εκτελέσουν εξειδικευμένες εργασίες όπως κίνηση, χειρισμός και αναγνώριση αντικειμένων.

2. Επίλυσης προβλημάτων. Στον συγκεκριμένο τομέα μελετώνται ευφυείς αλγόριθμοι εύρεσης λύσεων.

3. Μηχανική Μάθηση που έχει σκοπό να δοθεί σε ένα σύστημα η δυνατότητα να αυξήσει την απόδοση του μέσω επαγωγικών μεθόδων, όπως τα δέντρα απόφασης, να επιδεικνύει δυνατότητες ελέγχου προτύπων αυτό-βελτιώνοντας τη γνώση του και την απόδοση χωρίς την παρέμβαση του ανθρώπου.

4. Τα Νευρωνικά Δίκτυα, μέσω των οποίων γίνεται προσομοίωση της λειτουργίας του ανθρώπινου εγκεφάλου .

Από τη Siri, την Alexa ή τα bots που υπάρχουν σε όλο το διαδίκτυο μέχρι την αυτόνομη οδήγηση, η Τεχνητή Νοημοσύνη παρουσιάζει συνεχή πρόοδο. Αντίθετα με τις παλιότερες πεποιθήσεις, η Τεχνητή Νοημοσύνη δεν είναι μόνο ρομπότ με ανθρώπινα χαρακτηριστικά αλλά είναι πολύ περισσότερο οι αλγόριθμοι, όπως αυτοί που κρύβονται πίσω από κάθε εργαλείο αναζήτησης διαδικτύου καθώς και πίσω από το σύστημα IBM Watson. [5]

2.2 The Turing Test

Το 1950 ο Alan Turing, πρότεινε ένα τεστ, το οποίο μπορεί να χρησιμοποιηθεί για να διαπιστώσουμε πόσο καλό είναι ένα πρόγραμμα Τεχνητής Νοημοσύνης. Έχουμε λοιπόν τρία χωριστά δωμάτια, δύο ανθρώπους και έναν υπολογιστή. Οι άνθρωποι και ο υπολογιστής βρίσκονται σε τρία ξεχωριστά δωμάτια. Ο ένας από τους ανθρώπους έχει το

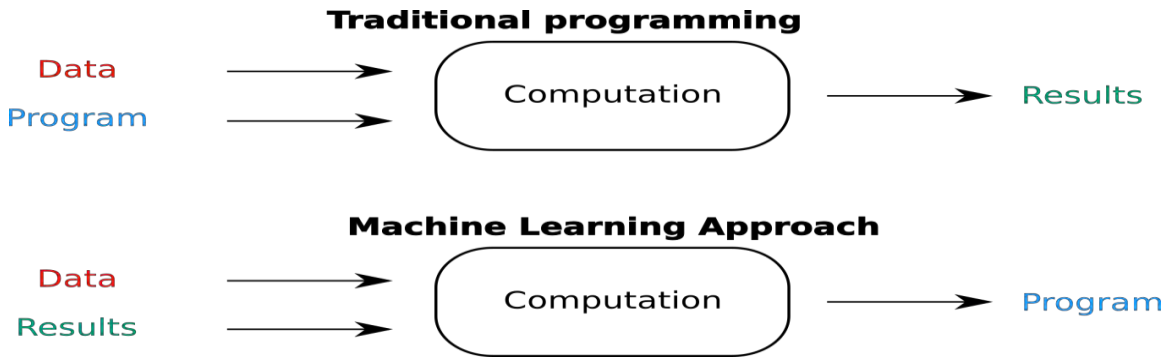
ρόλο του ανακριτή, ενώ ο δεύτερος άνθρωπος και ο υπολογιστής είναι οι εξεταζόμενοι. Απαραίτητη προϋπόθεση είναι ο ανακριτής να μην γνωρίζει σε ποιο δωμάτιο είναι το μηχάνημα και σε ποιο ο άνθρωπος. Ο ανακριτής θέτει διάφορα ερωτήματα στον άνθρωπο και στον υπολογιστή. Με βάση τις ξεχωριστές απαντήσεις που παίρνει, προσπαθεί να διακρίνει τον άνθρωπο από τη μηχανή. Εάν ο ανακριτής δεν καταφέρει να διαχωρίσει τους δύο, τότε μπορεί να θεωρηθεί ότι ο υπολογιστής κατέχει ευφυΐα. Μέχρι στιγμής ,το ρωσικό λογισμικό Eugene Goostman θεωρείται ότι κατάφερε να περάσει το τεστ Turing, αφού ξεπέρασε το όριο του 30% που είχε θέσει ο μαθηματικός. Συγκεκριμένα, σε ένα πεντάλεπτο διάλογο που είχε με εθελοντές ανακριτές, μπόρεσε να τους ξεγελάσει στο 33% των περιπτώσεων, αφού τους έπεισε με επιτυχία ότι ήταν ένα 13χρονο αγόρι από την Ουκρανία. Και αν μας φαίνεται μικρό το ποσοστό του 33%, είναι καλό να αναλογιστούμε τη πραγματική φύση ενός υπολογιστή. Είναι αναμφισβήτητο πως κάθε υπολογιστής υπερέχει όσον αφορά μαθηματικές πράξεις, και στη μνήμη με την έννοια της αποθήκευσης πληροφοριών. Όμως, μέχρι και τα πλέον προηγμένα υπολογιστικά συστήματα αδυνατούν να αναπαράγουν ακριβώς την διαδικασία της ανθρώπινης σκέψης, ακόμα και σε κάτι τόσο καθημερινό όσο ένας διάλογος. Πόσο μάλλον για ακόμα πιο περίπλοκες δραστηριότητες όπως η δημιουργικότητα και η πονηριά. [6]

Σήμερα, υπάρχει πληθώρα πειραμάτων τεχνητής νοημοσύνης και μπορεί ο καθένας μας να τα χρησιμοποιήσει. Ενδεικτικά αναφέρουμε τα πειράματα τεχνητής νοημοσύνης της Google(A.I. Experiments) όπως : Chrome Experiments, Handwriting with a Neural Net, What Neural Networks See και Visualizing High-Dimensional Space.

2.3 Machine Learning

Το Machine Learning είναι η επιστήμη η οποία κάνει τους υπολογιστές να δρουν χωρίς να έχουν προηγουμένως προγραμματιστεί. Την περασμένη δεκαετία η μηχανική μάθηση συνέβαλε σημαντικά στη δημιουργία αυτό-οδηγούμενων αυτοκινήτων, στην ανάπτυξη του speech recognition, στην αποτελεσματικότερη αναζήτηση στον παγκόσμιο ιστό, αλλά και στην βαθύτερη κατανόηση του ανθρώπινου γονιδιώματος. Το machine learning είναι τόσο διαδεδομένο στις μέρες μας, που πιθανώς χρησιμοποιείται πολλές φορές τη μέρα χωρίς να το καταλαβαίνουμε. Πολλοί ερευνητές θεωρούν πως είναι ο καλύτερος τρόπος για να σημειωθεί πρόοδος στην επικοινωνία ανθρώπου-μηχανής.

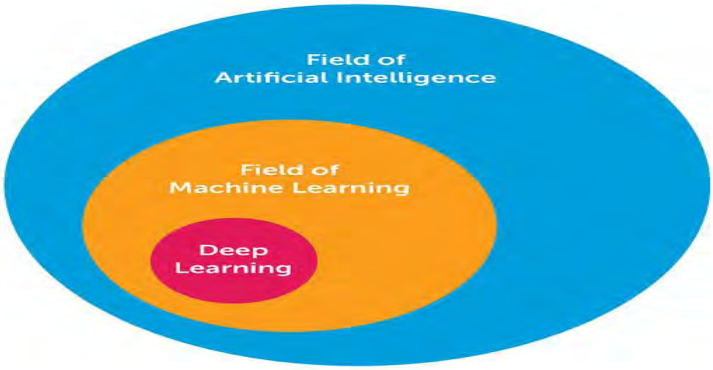
Ουσιαστικά μέσω του machine learning ένας υπολογιστής δέχεται ως είσοδο τα δεδομένα (input) αλλά και τα προσδοκώμενα αποτελέσματα και εξάγει τον αλγόριθμο (Εικόνα 1). Το μοντέλο αυτό μπορεί μελλοντικά να χρησιμοποιηθεί σε νέα δεδομένα και να εξάγει αληθινά αποτελέσματα αντίθετα με τον παραδοσιακό τρόπο προγραμματισμού όπου χρησιμοποιούνται τα δεδομένα και ο αλγόριθμος για την εξαγωγή αποτελεσμάτων.[7]



Εικόνα 1.Traditional programming VS Machine Learning Approach

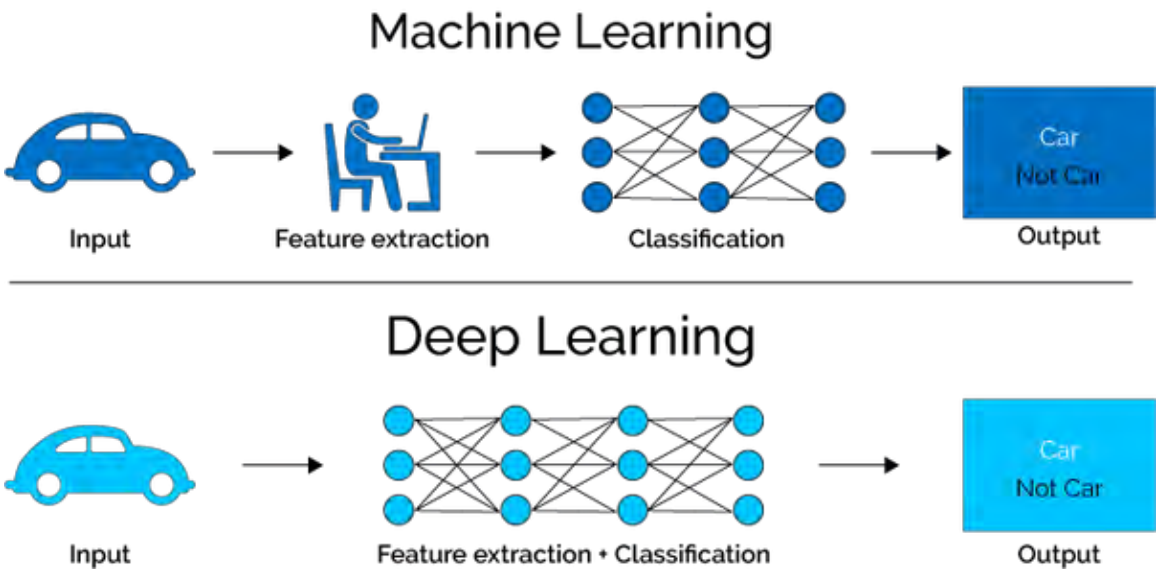
2.4 Deep Learning

Το Deep Learning είναι μια συγκεκριμένη υπό ενότητα του Machine Learning και της Τεχνητής Νοημοσύνης (Εικόνα 2). Στην ουσία είναι μια τεχνική που δίνει την ικανότητα στους υπολογιστές να προσεγγίσουν τον τρόπο σκέψης των ανθρώπων. Στο deep learning το υπολογιστικό μοντέλο μαθαίνει να κατηγοριοποιεί διεργασίες απευθείας από εικόνες, κείμενο και ήχο. Τα μοντέλα deep learning μπορούν να επιτύχουν τεράστια ακρίβεια και ακόμα και να ξεπεράσουν την ανθρώπινη επίδοση. Η λέξη deep στην φράση Deep Learning δεν σχετίζεται σε καμία περίπτωση με κάποιου είδους βαθύτερη κατανόηση. Αντιθέτως σχετίζεται με την ιδέα διαδοχικών στρώσεων(layers) αναπαραστάσεων. Το σύνολο των στρώσεων που συμβάλουν σε ένα μοντέλο δεδομένων είναι το βάθος του μοντέλου. Σήμερα, το deep learning συχνά αποτελείται από δεκάδες ή και εκατοντάδες επίπεδα στρωμάτων τα οποία έχουν την ικανότητα να εκπαιδευτούν αυτόματα από την έκθεση σε δεδομένα εκπαίδευσης.



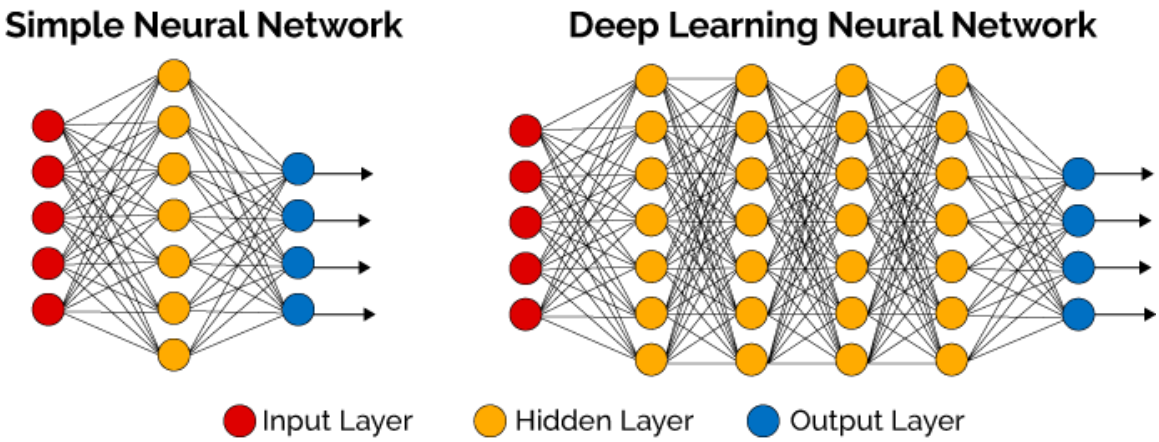
Εικόνα 2. Τεχνητή Νοημοσύνη, Machine Learning, Deep Learning.

Η κύρια διαφορά μεταξύ machine και deep leaning είναι η νευρωνική δομή που αποτελεί το deep learning. Στο machine learning υπάρχει ένα επίπεδο δεδομένων ενώ στο deep learning τα επίπεδα είναι πολυάριθμα (Εικόνα 3).



Εικόνα 3. Machine Learning VS Deep Learning.

Στο deep learning, αυτές οι πολυεπίπεδες αναπαραστάσεις (σχεδόν πάντα) μαθαίνονται απο μοντέλα που ονομάζονται νευρωνικά δίκτυα, δομημένα σε στρώματα στοιβαγμένα το ένα πάνω στο άλλο. Ο όρος νευρωνικό δίκτυο είναι μια αναφορά στη νευροβιολογία, αλλά αν και μερικές από τις κεντρικές έννοιες του deep learning αναπτύχθηκαν εν μέρει με την έμπνευση από την κατανόηση του εγκεφάλου, τα μοντέλα deep learning δεν είναι μοντέλα του εγκεφάλου. Δεν υπάρχουν αποδείξεις ότι ο εγκέφαλος εφαρμόζει οτιδήποτε όπως οι μηχανισμοί μάθησης που χρησιμοποιούνται στα σύγχρονα μοντέλα deep learning. Για τους σκοπούς μας, το deep learning είναι ένα μαθηματικό αντικείμενο για τη μάθηση παραστάσεων από δεδομένα. Το νευρωνικό δίκτυο μπορεί να έχει απλή μορφή με μικρό αριθμό νευρώνων ή να είναι ¨βαθύ¨ και να παρουσιάζουνε υψηλού πολυπλοκότητα και πολλούς νευρώνες (Εικόνα 4).



Εικόνα 4.Simple Neural Network VS Deep Learning Neural Network

Το deep learning παρουσιάζεται χρήσιμο σε πληθώρα εφαρμογών. Αρχικά στον τομέα της μηχανικής όρασης βρίσκει εφαρμογή επιχειρώντας την κατασκευή των συστημάτων αλλά και των ψηφιακών εικόνων – βίντεο με σκοπό την αλγοριθμική αναπαράσταση της αίσθησης της όρασης. Μερικές εφαρμογές είναι ο έλεγχος διαδικασιών (ή βιομηχανικό ρομπότ), ανίχνευση συμβάντων και οργάνωση πληροφοριών δηλαδή ευρετηριοποίηση βάσεων δεδομένων και ακολουθιών εικόνων. Επιπρόσθετα το deep learning έχει συμβάλει στην περαιτέρω ανάπτυξη της βιοπληροφορικής, της ανάκτησης πληροφορίας, της βελτιστοποίησης ακόμα και στον τομέα της οικονομίας. [8]

Το machine learning, και συνεπώς και το deep learning, χωρίζεται στις εξής τρείς κατηγορίες:

1. Supervised Learning

Αποτελεί την πιο συνηθισμένη μέθοδο. Ειδικά, στοχεύει στην αντιστοίχιση των δεδομένων που εισέρχονται σας είσοδοι, με τους αντίστοιχους στόχους που επίσης δίνονται.Το regression και το classification αποτελούν χαρακτηριστικά παραδείγματα Supervised learning

2. Unsupervised Learning

Ο εν λόγω τομέας της μηχανικής μάθησης επιχειρεί να εντοπίσει συσχετίσεις μόνο έχοντας δεδομένα τις εισόδους χωρίς την βοήθεια των στόχων. Ενδεικτικά παραδείγματα unsupervised learning είναι η οπτικοποίηση δεδομένων, συμπίεση

δεδομένων και η ανάλυση δεδομένων. Το unsupervised learning χωρίζεται σε dimensionality reduction και σε clustering.

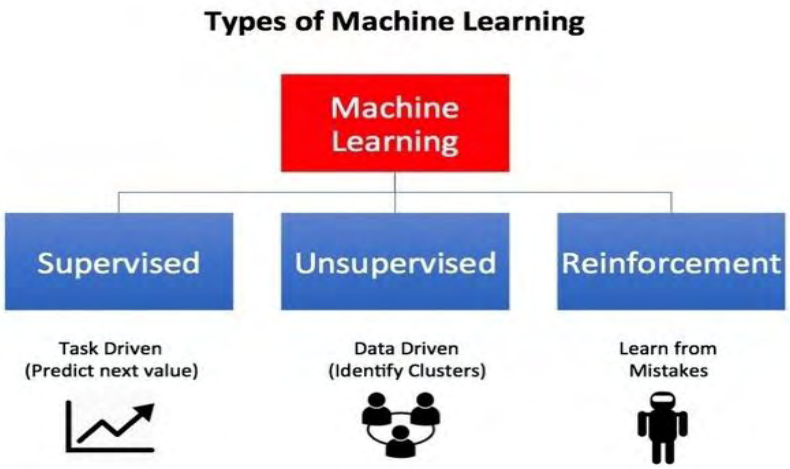
3. Self-Supervised

Η κατηγορία αυτή είναι υποπερίπτωση supervised learning παρουσιάζοντας μια διαφορά σχετικά με τα labels των δεδομένων. Συγκεκριμένα τα labels δεν τα ορίζει ο προγραμματιστής αλλά εξάγονται από τα δεδομένα εισόδου με ευρετικούς αλγορίθμους.

4. Reinforcement

Ο συγκεκριμένος τομέας του machine learning αναπτύχθηκε πρόσφατα με αφορμή την επιτυχή εφαρμογή του στην εκμάθηση video games. Συγκεκριμένα, ένας πράκτορας δέχεται πληροφορίες από το περιβάλλον και μαθαίνει να επιλέγει πράξεις οι οποίες θα μεγιστοποιήσουν την ανταμοιβή. Για παράδειγμα ένα νευρωνικό δίκτυο το οποίο παρακολουθεί μια οθόνη βιντεοπαιχνιδιού και αποφασίζεις τις κατάλληλες ενέργειες με σκοπό να επιτύχει την καλύτερη βαθμολογία. Μακροπρόθεσμος σκοπός του συγκεκριμένου τομέα είναι η υλοποίηση του σε αυτοοδηγούμενα αυτοκίνητα, στην ρομποτική και στην εκπαίδευση. [9]

Οι κατηγορίες παρουσιάζοντας στην Εικόνα 5.



Εικόνα 5.Οι κατηγορίες του Machine Learning.

2.5 Σύγχρονη μορφή της Τεχνητής Νοημοσύνης

Η μάθηση είναι ένα από τα βασικά χαρακτηριστικά της ανθρώπινης νοημοσύνης. Στον τομέα της τεχνητής νοημοσύνης, η μάθηση νοείται ως η ικανότητα χρήσης της εμπειρίας μέσα από τα δεδομένα, για τη βελτίωση της γνωστικής συμπεριφοράς.

Η νευροεπιστήμη έχει δείξει ότι οι ανθρώπινες νοητικές ικανότητες βασίζονται στην ενεργοποίηση σύνθετων δικτύων νευρώνων στον εγκέφαλο μας. Αυτά τα νευρωνικά δίκτυα είναι σε θέση να αποθηκεύουν πληροφορίες και γνώσεις και κατά συνέπεια να παρέχουν τις ικανότητες μάθησης. Περαιτέρω ανάπτυξη των νευρωνικών δικτύων θα γίνει στα κεφάλαια που ακολουθούν.

Αυτή η νέα γενιά AI έχει απόλυτη ανάγκη τα δεδομένα. Τα δικά μας δεδομένα και στον τομέα που μελετάμε οδηγικά δεδομένα σε πραγματικές συνθήκες.

Αν λόγου χάρη θέλουμε να «διδάξουμε» μία μηχανή να αναγνωρίζει την αλλαγή λωρίδας σε έναν αυτοκινητόδρομο τότε θα πρέπει να τροφοδοτήσουμε την μηχανή μας με χιλιάδες δεδομένα, όπως φωτογραφικά, για να είναι σε θέση να αναγνωρίσει ή και να ξεχωρίσει την πραγματική λωρίδα από μια λωρίδα έκτακτης ανάγκης ή λωρίδα έργων. Εάν, η έξοδος είναι σωστή, το δίκτυο θα ενισχύσει τις εσωτερικές του αλληλεπιδράσεις. Εάν, η έξοδος είναι λανθασμένη, θα πρέπει να τροποποιήσει τις αλληλεπιδράσεις του για να λάβει υπόψη τις σωστές πληροφορίες.

Η ικανότητα εκμάθησης των μηχανών βασίζεται στην ικανότητα του αλγορίθμου να βρει στατιστικά στοιχεία και συσχετίσεις στα δεδομένα που αναλύει.

Ως περιορισμένη τεχνητή νοημοσύνη αναφέρεται η τεχνητή νοημοσύνη που βασιζόμενη στα δεδομένα, μπορεί να κάνει πολύ καλά μία μόνο εργασία: να αναγνωρίσει

Η ουσία είναι ότι οι δεξιότητες που μπορούν να αποκτήσουν αυτές οι μηχανές για να εκτελέσουν μια εργασία δεν είναι μεταβιβάσιμες σε μια άλλη εργασία, κάτι που συνιστά μια σημαντική πτυχή της ανθρώπινης νοημοσύνης.

Η τεχνητή νοημοσύνη που βασίζεται σε χρήση δεδομένων, μπορεί να θεωρηθεί ως μια νέα μορφή νοημοσύνης, διαφορετική από αυτή του ανθρώπινου εγκεφάλου, που επιτρέπει στις μηχανές να εκτελούν καθήκοντα, όπως και οι άνθρωποι, αλλά πολύ γρηγορότερα. Χρησιμοποιώντας στατιστική συσχέτιση προερχόμενη από μια τεράστια ποσότητα δεδομένων, τα μηχανήματα είναι σε θέση να εκτελούν εργασίες που απαιτούν ευφυΐα όταν εκτελούνται από τον άνθρωπο. Ωστόσο, οι μηχανές το κάνουν με μη ανθρώπινες μεθόδους.

Ένας βασικός περιορισμός της τεχνητής νοημοσύνης είναι ότι δεν διαθέτει κοινή λογική και βούληση. Δεν είναι ακόμα δυνατό για τα μηχανήματα να καταλάβουν τι θα ακολουθήσει στη συνέχεια σε μια αλληλουχία από εικόνες, ούτε να κατανοήσουν το ευρύτερο πλαίσιο μιας σκηνής σε μια εικόνα. Αυτή η ικανότητα κατανόησης των «κανόνων του κόσμου», αποτελεί μία βασική ικανότητα του ανθρώπου που αναπτύσσεται σε νεαρή ηλικία, και είναι αυτό που λέμε με απλά λόγια «κοινή λογική». Τα μηχανήματα

δεν έχουν κοινή λογική και δεν κατανοούν τι συνιστά τη λειτουργία ενός αντικειμένου, σε τι χρησιμεύει κλπ.

Η μη εποπτευόμενη μάθηση (unsupervised) μπορεί να φέρει τις μηχανές πιο κοντά στην απόκτηση κοινής λογικής, αλλά και στην εκμάθηση του τρόπου διαχείρισης της αβεβαιότητας. Η μη εποπτευόμενη μάθηση χρησιμοποιείται όλο και περισσότερο, ώστε οι μηχανές να μάθουν τα καθήκοντα τους χωρίς να τα διδάσκονται: να μάθουν να παίζουν, δηλαδή, παιχνίδια παρακολουθώντας τα απλώς.

Επίσης, οι μηχανές AI παραμένουν ακόμα ανίκανες να μοιραστούν ένα στόχο, έναν σκοπό με έναν άνθρωπο. Είναι προγραμματισμένες να εκτελούν μια εργασία που μπορεί να βοηθήσει τους ανθρώπους να επιτύχουν ένα στόχο, αλλά δεν μπορούν να μοιραστούν αυτόν τον στόχο ως νόημα, ως αξία ή να τον κατανοήσουν.

Ένα αυτόνομο αυτοκίνητο είναι ένα εργαλείο για να φτάσει κάποιος από το σημείο Α στο σημείο Β, δεν μπορεί, όμως, να μοιραστεί το στόχο ενός επιβάτη να ακολουθήσει τη διαδρομή που προσφέρει τα καλύτερα αξιοθέατα.

Οι μηχανές δεν μοιράζονται την σκοπιμότητα με τον χειριστή τους. Είναι εργαλεία που δεν μπορούν να συνεργαστούν, ανεξάρτητα από το πόσο έξυπνα εμφανίζονται, αλλά να αλληλοεπιδράσουν. Αυτό έχει σημαντικές επιπτώσεις στην υπό εκκόλαψη σχέση μεταξύ ανθρώπων και συστημάτων τεχνητής νοημοσύνης. Οι άνθρωποι πρέπει να είναι σε θέση να αναλάβουν τον έλεγχο και να διορθώσουν το σύστημα όταν εμφανιστεί κάτι απρόσμενο ή όταν υπάρξει ανάγκη.[10]

Κεφάλαιο 3

Ανάλυση των Μεθόδων που Εφαρμόστηκαν

Στο παρον κεφάλαιο θα αναπτυχθούν όλα τα εργαλεία και οι μέθοδοι που χρησιμοποιήθηκαν για την ανάπτυξη του μοντέλου μας. Παράλληλα θα πραγματοποιηθεί ανάλυση των δεδομένων και των τρόπων επεξεργασίας τους και τελικά θα παρουσιαστούν τα είδη των νευρικών δικτύων πάνω στα οποία αναπτύχθηκαν οι κώδικες.

3.1 TensorFlow

Το TensorFlow είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα για αριθμητικούς υπολογισμούς χρησιμοποιώντας γραφήματα ροής δεδομένων. Αρχικά αναπτύχθηκε από την ομάδα Google Brain σε συνεργασία με τον ερευνητικό οργανισμό 'Machine Intelligence research organization for machine learning and deep neural networks'.Παρόλα αυτά το σύστημα έχει την δυνατότητα να εφαρμοστεί σε ένα ευρύ φάσμα άλλων τομέων.[11]

Το TensorFlow είναι ένα σύστημα cross-platform δηλαδή εχει την ικανότητα να εκτελείται σχεδόν παντού: Επεξεργαστές και Κάρτες Γραφικών κινητών και ενσωματωμένων πλατφόρμων και ακόμη και σε μονάδες επεξεργασίας tensor(TPU),οι οποίες είναι εξειδικευμένες για την εκτέλεση tensor math.

Τα πλεονεκτήματα του TensorFlow σε σχέση με τα υπόλοιπα backend όπως είναι το Theano και το CNTK είναι αρχικά ότι υποστηρίζεται από την Google,η οποία αναβαθμίζει συνεχώς την πλατφόρμα με νέα χαρακτηριστικά και επίσης παρέχει πολλούς οδηγούς χρήσης της πλατφόρμας. Ακόμα ακολουθεί τις ανάγκες της αγοράς και έχει την δυνατότητα να χρησιμοποιήσει GPUs και Google TPUs.

3.2 Keras

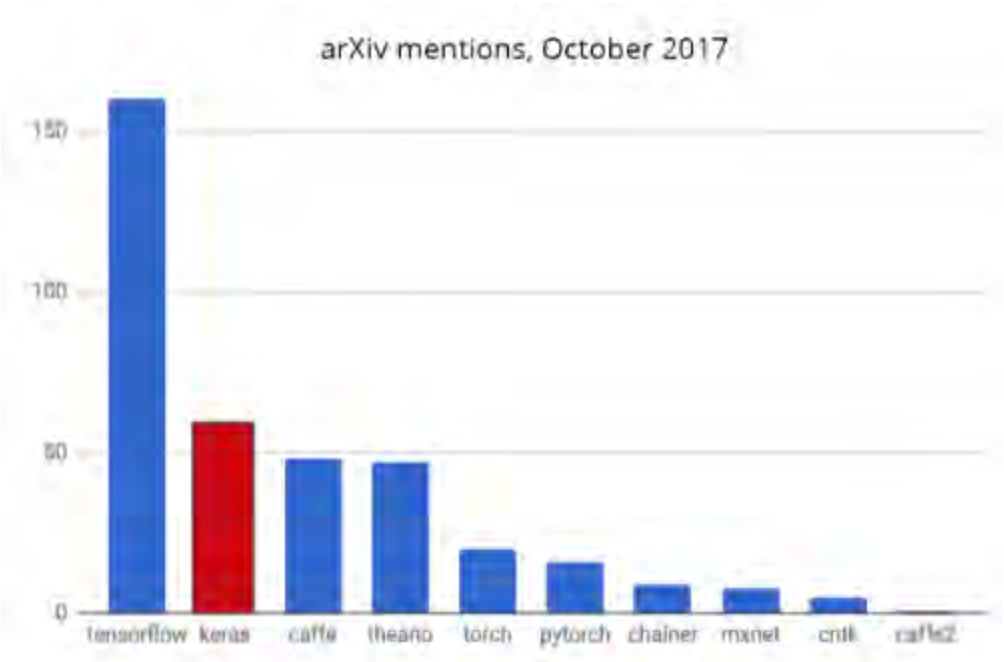
Το Keras είναι ένα programming framework για βαθιά εκμάθηση το οποίο απλοποιεί την διαδικασία κατασκευής εφαρμογών βαθιάς μάθησης. Αντί να παρέχει όλες τις λειτουργίες από μόνο του, χρησιμοποιεί είτε το TensorFlow είτε το Theano και προσθέτει μια τυπική, απλοποιημένη διεπαφή προγραμματισμού(interface).[12]

Η κυριότερη διαφορά μεταξύ TensorFlow και Keras είναι η εξής: τα πλαίσια βαθιάς μάθησης λειτουργούν σε δύο επίπεδα. Το πρώτο και χαμηλότερο επίπεδο είναι αυτό που

ανήκει και το TensorFlow. Είναι το επίπεδο που εφαρμόζονται μαθηματικές πράξεις όπως οι πολλαπλασιασμοί Matrix-Matrix. Το δεύτερο και υψηλότερο επίπεδο είναι και αυτό που ανήκει το Keras. Σε αυτό το επίπεδο χρησιμοποιούνται τα αποτελέσματα του πρώτου επιπέδου με σκοπό να γίνουν τροποποιήσεις στο νευρωνικό δίκτυο όπως layers και models. Γενικότερα στο υψηλότερο επίπεδο εφαρμόζονται διαδικασίες για την εκπαίδευση του μοντέλου.

Σήμερα, υπάρχουν χιλιάδες framework για βαθιά μάθηση (Εικόνα 6). Μέσα λοιπόν από ένα πλήθος επιλέχθηκε το Keras .Οι βασικοί λόγοι της επιλογής αυτής είναι οι εξής:

1) Το Keras έχει ευρεία υιοθέτηση από τον χώρο της βιομηχανίας καθώς και στην ερευνητική κοινότητα. Συγκεκριμένα το Keras χρησιμοποιείται από περισσότερους από 200 χιλιάδες χρήστες καθώς και από μεγάλες εταιρίες όπως Netflix, Uber, Yelp, Instcart, ZocDoc, Square και αρκετές ακόμα. Είναι επίσης ιδιαίτερα δημοφιλές σε startup εταιρίες που έχουν την βαθιά μάθηση ως το κύριο προϊόν τους.



Εικόνα 6. Τα framework για βαθιά μάθηση.

2)Το keras δίνει την δυνατότητα της εύκολης μετατροπής του μοντέλου σε προϊόν όπως εφαρμογές android ή ios, raspberry pi, java environment , εφαρμογές φυλλομετρητή κ.α.

3)Το Keras υποστηρίζει αρκετές μηχανές backend και δεν δεσμεύει το χρήστη του σε ένα οικοσύστημα. Συνεπώς η χρήση και η προσαρμογή σε ένα backend διαφορετικού του TensorFlow μπορεί να γίνει εύκολα. Η Amazon αναπτύσσει δικό της backend , το

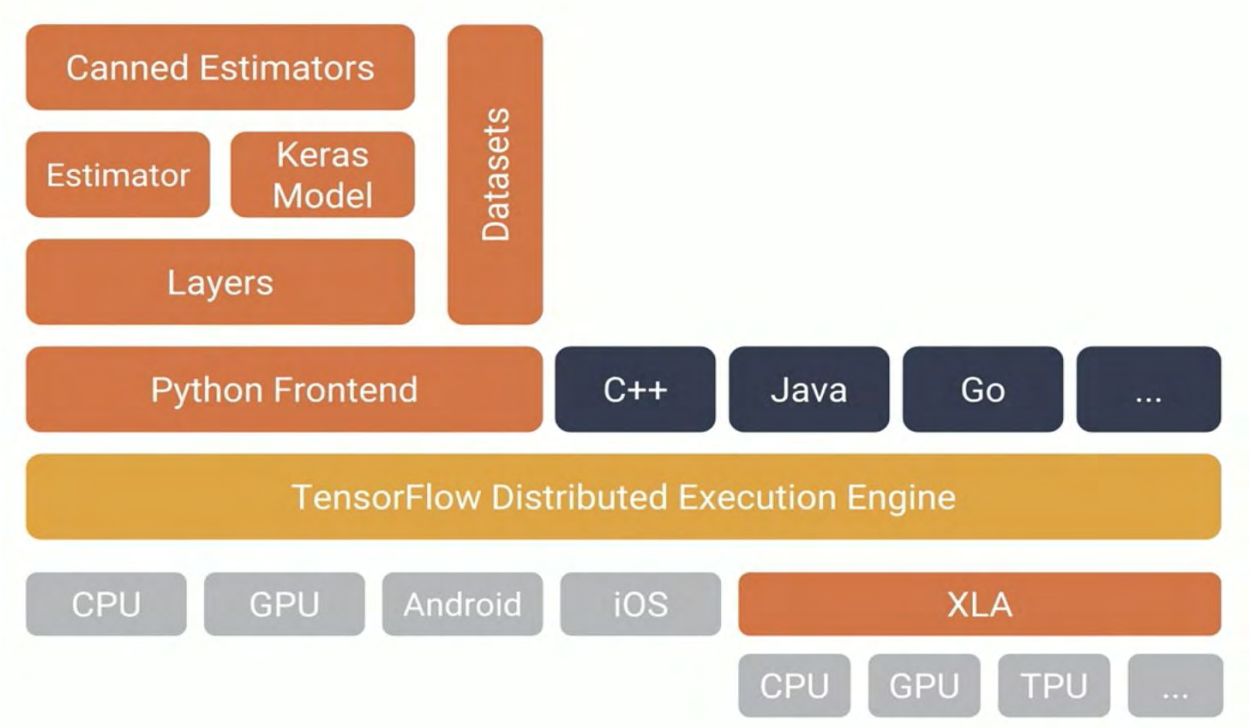
MXNET backend [13], για να το χρησιμοποιήσει σε συνδυασμό με το Keras. Επιπρόσθετα τα μοντέλα του Keras έχουν την δυνατότητα να εκπαιδευτούν σε διαφορετικές πλατφόρμες όπως NVIDIA GPU, OPENCL-enabled GPUs και Google TPUs.

3.3 TensorFlow σε Συνδυασμό με το Keras

Στην εργασία και στην υλοποίηση επιλέχθηκε ως backend το TensorFlow σε συνδυασμό με ένα framework υψηλότερου επιπέδου όπως είναι το Keras . Οι λόγοι που οδήγησαν στην επιλογή αυτή είναι οι εξής:

1)Το Keras προσφέρει ένα πιο απλό περιβάλλον από το σχετικά περίπλοκο του TensorFlow. Παρέχει ένα υψηλότερου επιπέδου API, το οποίο γράφεται σε Python και τρέχει πάνω από το TensorFlow (Εικόνα 7).

2)Σε συνέχεια του 1, το Keras δίνει τη δυνατότητα να αφιερωθεί περισσότερος χρόνος για τα πράγματα που βρίσκονται στην επιφάνεια παρά για όλα αυτά που γίνονται στο παρασκήνιο.



Εικόνα 7. Τα API του TensorFlow

Συνοψίζοντας, για τη δημιουργία ενός αποτελεσματικού νευρωνικού δικτύου χρειάστηκαν τρία εργαλεία. Αρχικά το TensorFlow, το οποίο παρέχει το κατάλληλο Back-End που

γίνονται όλες οι μαθηματικές πράξεις και παράλληλα θα "κουμπώσουν" όλα τα υπόλοιπα layers της δομής όπως το programming framework και το API. Το Keras παρέχει όλες τις κατάλληλες βιβλιοθήκες για την κατασκευή του μοντέλου μας. Η γλώσσα προγραμματισμού που επιλέξαμε είναι η Python καθώς έχει ευκολία στην χρήση της, μιας και επιτρέπει να περιοριστεί το μέγεθος του κώδικα σημαντικά σε σχέση με C+ και Java, έχει αρκετές βιβλιοθήκες και έτοιμες λειτουργίες και τέλος μπορεί να πακεταριστεί σε εκτελέσιμα προγράμματα που τρέχουν αυτούσια στα πιο δημοφιλή λειτουργικά συστήματα.

3.4 Ανάλυση Νευρωνικού Δικτύου

3.4.1 Η ανατομία του Νευρωνικού Δικτύου – Πλεονεκτήματα Νευρωνικών

Ένα νευρωνικό δίκτυο είναι ουσιαστικά ένα κύκλωμα διασυνδεδεμένων νευρώνων το οποίο έχει ως σκοπό να προβλέψει (prediction) τις επιθυμητές εξόδους (targets) από τις εισόδους (input data) που θα έχει τροφοδοτηθεί. Ο πρωταρχικός σκοπός της λειτουργίας ενός νευρωνικού δικτύου είναι να μπορεί να λύνει συγκεκριμένα προβλήματα ή να πραγματοποιεί συγκεκριμένες διεργασίες χωρίς τη δική μας παρέμβαση. Για να επιτευχθεί όμως αυτό θα πρέπει το νευρωνικό δίκτυο να εκπαιδευτεί κατάλληλα. Αυτό είναι και το κύριο χαρακτηριστικό των νευρωνικών δικτύων δηλαδή η ικανότητα τους να μαθαίνουν ή να εκπαιδεύονται.

Η διαδικασία της μάθησης γίνεται μέσω κάποιων παραδειγμάτων (training data όπου θα αναφερθούμε διεξοδικά στο κεφάλαιο 3.5.1) καθώς και ενός συγκεκριμένου αλγορίθμου εκπαίδευσης όπου έχει ως στόχο τη βελτίωση της απόδοσης του δικτύου. Στην ουσία ο εν λόγω αλγόριθμος είναι μια επαναληπτική διαδικασία όπου σε κάθε επανάληψη αλλάζουν οι παράμετροι (οι ελεύθεροι παράμετροι) του νευρωνικού δικτύου με σκοπό σταδιακά να μειωθεί το σφάλμα μεταξύ της επιθυμητής εξόδου και της εξόδου του δικτύου. Υπάρχουν πολλά είδη μάθησης, καθένα από τα οποία αλλάζει διαφορετικά τις παραμέτρους. Οι κυριότεροι και βασικότεροι τρόποι μάθησης είναι η μάθηση με επίβλεψη (supervised learning) ,η βαθμολογημένη μάθηση (graded learning), η μάθηση χωρίς επίβλεψη (unsupervised learning) καθώς και reinforcement.

Αξίζει να αναφερθεί μια ικανότητα που θα πρέπει να έχει το νευρωνικό δίκτυο, την ικανότητα της γενίκευσης. Η ικανότητα της γενίκευσης αναφέρεται στο γεγονός ότι το νευρωνικό δίκτυο μπορεί να εκτιμά με επιτυχία τους στόχους για εισόδους που δεν εχει εκπαιδευτεί και όχι μόνο για τις εισόδους που χρησιμοποιήθηκαν για την εκπαίδευση του. Η ικανότητα της γενίκευσης αποτελεί ένα από τα σημαντικότερα χαρακτηριστικά ενός νευρωνικού δικτύου καθώς στο μεγαλύτερο αριθμό προβλημάτων δεν μπορούμε να γνωρίζουμε από πριν τις καταστάσεις που θα αντιμετωπίσει το νευρωνικό δίκτυο. Η ικανότητα της γενίκευσης επηρεάζεται από την πολυπλοκότητα του προβλήματος, το

μέγεθος του νευρωνικού δικτύου καθώς και από τα δεδομένα εκπαίδευσης που έχουν δοθεί ως είσοδο στο νευρωνικό και στο κατά πόσο αυτά αναπαριστούν ικανοποιητικά το περιβάλλον από τον οποίο προέρχονται.

Βασικά πλεονεκτήματα της προσέγγισης των νευρωνικών δικτύων αποτελούν, η ικανότητα των νευρώνων να δέχονται ρυθμιζόμενες παραμέτρους και η δυνατότητα παράλληλης επεξεργασίας των παραμέτρων αυτών με αποτελέσματα την αύξηση της αποδοτικότητας. Αξίζει να αναφερθούν τα πλεονεκτήματα στην ικανότητα της γενίκευσης όπως αναλύθηκε προηγουμένως, τη λειτουργία σε πραγματικό χρόνο με πραγματικές συνθήκες και τέλος την ικανότητα διόρθωσης σφαλμάτων.

3.4.2 Layers

Αποτελούν τα δομικά στοιχεία ενός νευρωνικού δικτύου. Πιο συγκεκριμένα δέχονται σαν είσοδο κάποια tensors (πίνακες) και δημιουργούν νέα. Ανάλογα με το είδος της εισόδου χρησιμοποιούνται διαφορετικά είδη layer. Για παράδειγμα για απλά διανύσματα χρησιμοποιούμε 2D tensors (samples,features) τα οποία ονομάζονται dense connected layers ή fully connected layers. Αν έχουμε αλληλουχίες δεδομένων (sequence data) τότε χρησιμοποιούμε 3D tensors (samples,timesteps,features) δηλαδή recurrent layers. Τέλος για εικόνες χρησιμοποιούμε 4D tensors δηλαδή convolution layers.

Αφού υλοποιήσουμε την αρχιτεκτονική οφείλουμε να επιλέξουμε τις κατάλληλες παραμέτρους για να ολοκληρώσουμε την εκπαίδευση του νευρωνικού μας δικτύου.

3.4.3 Loss Function (objective function)

Είναι μία συνάρτηση η οποία έχει το ρόλο της ανατροφοδότησης κατά τη διάρκεια της εκπαίδευσης του μοντέλου μας. Αναλυτικά, παίρνει την πρόβλεψη η οποία έχει εξαχθεί και τον πραγματικό στόχο που έχουμε δώσει εμείς και υπολογίζει ένα distance score για να παρατηρήσουμε την αποτελεσματικότητα του μοντέλου στην πρόβλεψη του.

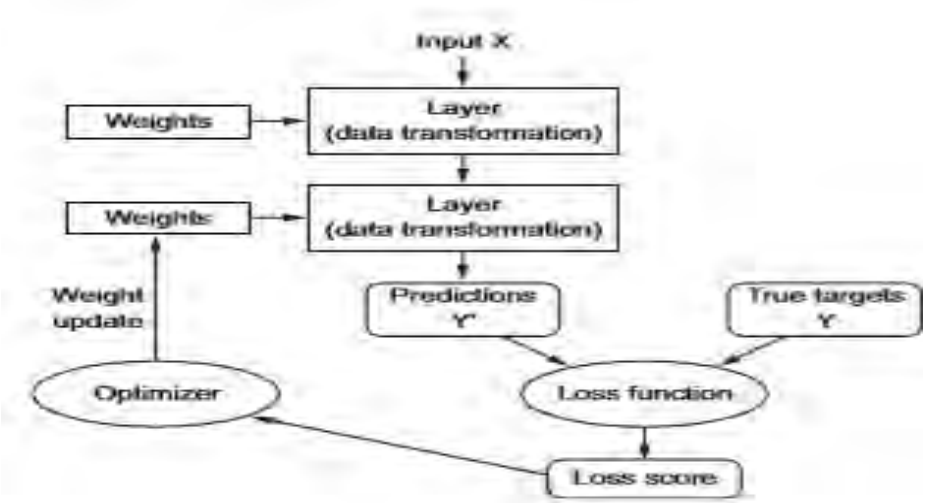
3.4.4 Optimizer

Την ακριβώς πιο πάνω εργασία, το να μειώσουμε δηλαδή όσο το δυνατόν περισσότερο τη διαφορά μεταξύ πρόβλεψης και πραγματικής τιμής έρχεται να την κάνει ο optimizer. Ουσιαστικά είναι ένας μηχανισμός μέσω του οποίου το δίκτυο ανανεώνεται με βάση τα δεδομένα που έχει δει αλλά και το loss function

Για κάθε πρόβλημα που τίθεται προς επίλυση από ένα μοντέλο μηχανικής μάθησης υπάρχουν δύο παράμετροι που το εκπροσωπούν. Πρώτον είναι τα patterns τα οποία καλείται να εντοπίσει το μοντέλο ώστε να κατηγοριοποιήσει κατάλληλα τα δεδομένα και να αυξήσει την ακρίβεια και την αποδοτικότητα του. Παράλληλα όμως υπάρχει και η παράμετρος του stochastic noise. Συγκεκριμένα, εκτός από τα patterns το μοντέλο μπορεί να μάθει και κομμάτια θορύβου τα οποία ενώ από τη μία αυξάνουν το accuracy του training

χαλάνε πολύ τη διαδικασία πρόβλεψης του μοντέλου πάνω σε δεδομένα τα οποία είναι άγνωστα προς αυτό. (overfitting)

Το overfitting, ή αλλιώς υπερμοντελοποίηση, είναι όταν το αποτέλεσμα μίας ανάλυσης είναι προσαρμοσμένο σε ένα συγκεκριμένα σύνολο δεδομένων (data set) με αποτέλεσμα το μοντέλο να μην μπορεί να προσαρμοστεί σε νέα δεδομένα ή να προβλεψει νέες παρατηρήσεις. Οι παραπάνω μηχανισμοί παρουσιάζονται στην Εικόνα 8



Εικόνα 8. Το νευρωνικό δίκτυο και οι μηχανισμοί του.

Για την καταπολέμηση αυτού του προβλήματος υπάρχουν συγκεκριμένες μέθοδοι:

Regularization

Είναι μια μέθοδος η οποία προσθέτει ένα πέναλτι σε διαφορετικές παραμέτρους του μοντέλου με σκοπό να μειώσει την ελευθερία του. Με αυτόν τον τρόπο είναι πιο δύσκολο να εντοπίσει το θόρυβο κατά την εκπαίδευση και βελτιώνει την αποδοτικότητα του σε άγνωστα δεδομένα.

-L1 regularization (lasso)

Η L1 regularization προσθέτει ένα πέναλτι ίσο με το άθροισμα των απόλυτων τιμών των

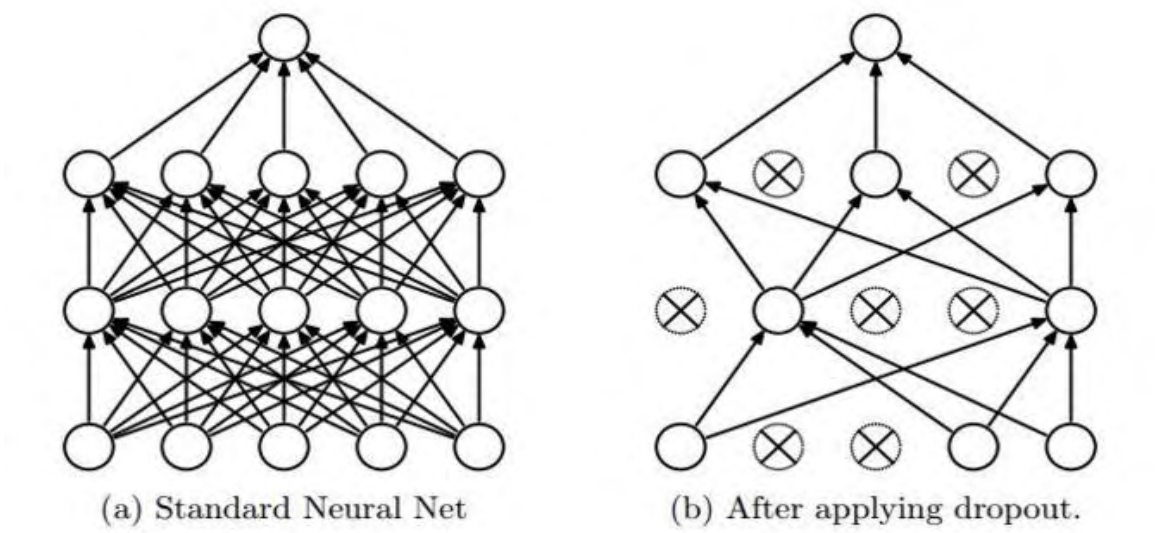
συντελεστών. Συγκεκριμένα θα τοποθετήσει κάποιες στο μηδέν.

-L2 regularization (ridge)

Η L2 regularization προσθέτει ένα πέναλτι ίσο με το άθροισμα των τετραγώνων των συντελεστών. Εξαναγκάζει τις παραμέτρους να πάρουν τιμές σχετικά μικρές.

Dropout

Μία άλλη τεχνική για την καταπολέμηση του overfitting είναι το Dropout. Συγκεκριμένα, στη μέθοδο αυτή επιλέγονται τυχαία κάποιοι νευρώνες οι οποίοι αγνοούνται κατά την εκπαίδευση (Εικόνα 9). Αυτό σημαίνει πως η συνεισφορά τους απενεργοποιείται προσωρινά αφού δεν ανανεώνονται συγκεκριμένα βάρη και ότι οι υπόλοιποι νευρώνες είναι υπεύθυνοι για την πρόβλεψη των τιμών. Η επίδραση είναι ότι το μοντέλο γίνεται λιγότερο ευαίσθητο για συγκεκριμένους νευρώνες και γίνεται ικανότερο στο generalization. Έτσι, αποφεύγεται το overfitting στα δεδομένα της εκπαίδευσης.



Εικόνα 9. Νευρωνικό Δίκτυο μετά την υλοποίηση του dropout.

3.5 Data Set

Το Data Set, δηλαδή το σύνολο των δεδομένων που χρησιμοποιούνται ως είσοδο στο νευρωνικό δίκτυο, χωρίζονται σε τρείς κατηγορίες ανάλογα με τον τρόπο διαχείρισης τους αλλά και την λειτουργία τους. Οι κατηγορίες αυτές είναι τα δεδομένα για training (training data), τα δεδομένα για testing (test data), και τα δεδομένα για validation (validation data) (Εικόνα 10).

Εικόνα 10. Διαχωρισμός του data set.

3.5.1 Training Data - Training Accuracy

Training data είναι τα δεδομένα τα οποία δέχεται ως είσοδο το εκάστοτε μοντέλο με σκοπό την εκπαίδευση του. Με τον όρο εκπαίδευση εννοούμε να μπορεί να συσχετίσει με τον καλύτερο δυνατό τρόπο τα data-labels. Training labels είναι ουσιαστικά η αντιστοιχία ενός data με το label του. Γενικότερα όσο καλύτερα είναι τα training data τόσο καλύτερο θα είναι το μοντέλο στις επιδόσεις του. Το ίδιο ανακαλύπτει συσχετίσεις, αναπτύσσει αποφάσεις και πραγματοποιεί προβλέψεις με βάση το training data που του έχει δοθεί. Φυσικά δεν αρκεί τα δεδομένα να είναι πολλά και σωστά δομημένα αλλά πρέπει να είναι και κατάλληλα για το σύστημα μας. Για παράδειγμα τα αυτόνομα οχήματα δε χρειάζονται απλά εικόνες από το δρόμο, χρειάζονται εικόνες άλλων οχημάτων, πεζών, πινακίδες αλλά και το πως θα αλληλοεπιδρούν με αυτά. Είναι πολύ σημαντικό τα training data να βρίσκονται σε συνεχή ανάπτυξη και να γίνονται όλο και καλύτερα, ώστε το μοντέλο να εκπαιδεύεται με τον ιδανικότερο τρόπο.

Με τον όρο training accuracy αναφερόμαστε στην ακρίβεια του μοντέλου κατά την εκπαίδευση του με είσοδο τα training data-labels.Συγκεκριμένα υπολογίζεται το κατά πόσο ακριβής είναι η συσχέτιση μεταξύ training data με training labels.

3.5.2 Test Data – Test Accuracy

Αφού έχει προηγηθεί η εκπαίδευση του μοντέλου, στη συνέχεια γίνεται η αξιολόγηση του. Για την αξιολόγηση χρησιμοποιούμε δεδομένα που τα ονομάζουμε test data-labels με σκοπό να δούμε την λειτουργικότητα του μοντέλου πάνω σε δεδομένα τα οποία είναι άγνωστα προς το ίδιο. Συγκεκριμένα πρέπει να είναι άγνωστα, αφού δε θα είχε νόημα να ελέγχαμε την απόδοση του πάνω σε δεδομένα τα οποία γνωρίζει ήδη.

Με τον όρο test accuracy αναφερόμαστε στην ακρίβεια του μοντέλου κατά την διαδικασία της αξιολόγησης του. Δηλαδή, μετά την εκπαίδευση του μοντέλου, μετρά τις επιδόσεις του στα testing data.

3.5.3 Validation Data - Validation Accuracy

Με σκοπό να παρακολουθούμε την ακρίβεια του μοντέλου κατά τη διαδικασία της εκπαίδευσης του χρησιμοποιούμε ένα ξεχωριστό dataset το οποίο περιέχει δεδομένα τα οποία δεν τα έχει ξαναδεί το δίκτυο. Με αυτό τον τρόπο μπορούμε να ελέγξουμε το πόσο σωστό είναι το μοντέλο μας αφού μας επιστρέφει το accuracy σε δεδομένα άγνωστα προς αυτό και διαφορετικά από τα δεδομένα που χρησιμοποιούνται για το training.

3.5.4 Πλήθος Δεδομένων

Το μέγεθος των δεδομένων που χρειαζόμαστε εξαρτάται από δυο βασικούς παράγοντες. Αρχικά από την πολυπλοκότητα του προβλήματος και στην συνέχεια από την πολυπλοκότητα του επιλεγμένου αλγορίθμου. Παρόλα αυτά, οι δύο αυτοι παράγοντες δεν σε βοηθούν να επιλέξεις με ακρίβεια το μέγεθος των δεδομένων. Σε περίπτωση μεγάλου όγκου δεδομένων ίσος χρειαστεί να κάνεις περικοπές ώστε να αναπτυχθεί κατάλληλα το μοντέλο και να αποφευχθεί το overfitting. Σε περίπτωση λίγων δεδομένων ίσως χρειάζεται να αυξηθεί ο όγκος των δεδομένων ή να εφαρμοστεί κατάλληλη μέθοδος ώστε να αξιοποιούνται τα ήδη υπάρχοντα δεδομένα. Τέλος, στην περίπτωση που δεν έχουν συλλεγεί τα δεδομένα ο προγραμματιστής οφείλει να πάει με την πειραματική μέθοδο και να προσθαφερεί δεδομένα δοκιμάζοντας παράλληλα την απόδοση του μοντέλου. Στο κεφάλαιο 7.3 γίνεται ανάλυση συσχέτισης μεταξύ του όγκου των δεδομένων και της αποδοτικότητας του νευρωνικού στα δικά μας δεδομένα.

3.5.5 Επεξεργασία των Δεδομένων

Ένα νευρωνικό δίκτυο δέχεται στην είσοδο του δεδομένα τα οποία τα χρησιμοποιεί για την εκπαίδευση του. Συνεπώς όσο "καλύτερα" είναι τα δεδομένα εισόδου στο νευρωνικό,τόσο καλύτερα αποτελέσματα θα πάρουμε στην έξοδο του δικτύου. Υπάρχουν αρκετοί τρόποι να γίνει η επεξεργασία των δεδομένων. Αναλύουμε τους τρόπους αυτούς :

- Ο πρώτος , και πιο απλός τρόπος , είναι τα δεδομένα να μπουν χωρίς επεξεργασία στο δίκτυο. Σε αυτή την περίπτωση εάν τα αρχικά δεδομένα μας έχουν μεγάλη διασπορά ή υπάρχουν ακραίες τιμές ο οποίες αποκλίνουν αρκετά από το μέσο όρο των υπόλοιπων τιμών τότε το νευρωνικό θα δυσκολευτεί να κάνει το classification σωστά.

- Για να εξαλείψουμε τον κίνδυνο των ακραίων τιμών στην είσοδο του νευρωνικού θα χρειαστεί να περάσουμε τα δεδομένα μας από μια διαδικασία κανονικοποίησης. Στην συγκεκριμένη μέθοδο επεξεργασίας των δεδομένων οι ακραίες τιμές πλησιάζουν στην μέση τιμή του συνόλου των τιμών. Αρχικά υπολογίζεται ο μέσος όρος των τιμών, αφαιρώ από κάθε στοιχείο τον μέσο όρο και στην συνέχεια διαιρώ από τα στοιχεία που έχει αφαιρεθεί ο μέσος όρος την τυπική απόκλιση.

- Παράλληλα ένας άλλος τρόπος επεξεργασίας των δεδομένων για την καλύτερη εισαγωγή τους στο νευρωνικό μας δίκτυο είναι το one-hot encoding. Πιο συγκεκριμένα αποτελεί μια μέθοδο μέσω της οποίας ένας αριθμός μετατρέπεται σε ένα διάνυσμα το όποιο αποτελείται από μηδενικά εκτός από τη θέση που αντιστοιχεί αυτός ο αριθμός η οποία παίρνει την τιμή 1. Η συγκεκριμένη μέθοδος ονομάζεται One-Hot [14].

Είναι πολύ σημαντικό να σημειωθεί ότι δεν υπάρχει "καλύτερη" μέθοδος για την επεξεργασία των δεδομένων. Η βέλτιστη μέθοδος επιλέγεται μέσα από μια διαδικασία δοκιμών και παραμετροποιήσεων. Στις υλοποιήσεις μας χρησιμοποιήσαμε και τις τρείς μεθόδους και συνοψίσαμε τα αποτελέσματα σε διαγράμματα που θα παρουσιαστούν στο κεφάλαιο 5.2.

3.5.6 Επεξεργασία Δεδομένων με Παράθυρα

Η επεξεργασία των δεδομένων η οποία παρουσιάστηκε στην προηγούμενη υποενότητα μπορεί να εφαρμοστεί υλοποιόντας μια διαφορετική προσέγγιση. Συγκεκριμένα, έχοντας τα αρχικά δεδομένα μας, δηλαδή γραμμές-εγγραφές οι οποίες απεικονίζουν την κατάσταση του οχήματος σε ορισμένες χρονικές στιγμές, αποφασίστηκε σημαντικό να ομαδοποιηθούν κάποιες εγγραφές μεταξύ τους ώστε η χρονική διάρκεια να αυξηθεί και τελικά το μοντέλο να προσαρμόζεται και να εκπαιδεύεται αποδοτικότερα στα μοτίβα που εμφανίζονται. Οι ομαδοποιήσεις έχουν γίνει σε ζευγάρια των 2 και των 5.

3.6 Classification - Classifiers

Η διαδικασία του classification είναι η προσπάθεια προσέγγισης μια συνάρτησης χαρτογράφησης (f) από μεταβλητές εισόδου (x) σε διακριτές μεταβλητές εξόδου (y).

Οι μεταβλητές εξόδου συχνά ονομάζονται labels ή categories. Η συνάρτηση χαρτογράφησης προβλέπει την τάξη(labels) ή την κατηγορία(category) για μια δεδομένη παρατήρηση

Για παράδειγμα ένα μήνυμα ηλεκτρονικού ταχυδρομείου μπορεί να ταξινομηθεί σε μια από δύο κατηγορίες: “spam” ή “not spam”.Το πρόβλημα της κατάταξης απαιτεί το κάθε email να ταξινομηθεί σε μια από τις δυο κατηγορίες.

Είναι σύνηθες για τα μοντέλα ταξινόμησης να προβλέπουν μια τιμή σαν πιθανότητα ενός δεδομένου παραδείγματος που ανήκει σε μια κλάση εξόδου. Μια πιθανότητα μπορεί να μετατραπεί σε μια κλάση επιλέγοντας την ετικέτα(label) της κλάσης που έχει την υψηλότερη πιθανότητα.

Για παράδειγμα, σε ένα συγκεκριμένο μήνυμα ηλεκτρονικού ταχυδρομείου μπορεί να αποδοθούν οι πιθανότητες 0.1 ως "spam" και 0.9 ως "not spam". Μπορούμε να μετατρέψουμε αυτές τις πιθανότητες σε μια ετικέτα κλάσης(label) επιλέγοντας την ετικέτα "not spam" καθώς έχει την υψηλότερη προβλεπόμενη πιθανότητα.

Συνεπώς ένα πρόβλημα classification απαιτεί:

1. Τα παραδείγματα να ταξινομούνται σε μία από τις δύο ή περισσότερες κατηγορίες.
2. Πραγματικές ή διακριτές μεταβλητές εισόδου.
3. Η έξοδος να είναι διακριτή μεταβλητή

Ένας αλγόριθμος που εφαρμόζει την ταξινόμηση είναι γνωστός ως classifier. Ο όρος classifier αναφέρεται επίσης μερικές φορές στη μαθηματική συνάρτηση, που εφαρμόζεται από έναν αλγόριθμο ταξινόμησης, ο οποίος χαρτογραφεί δεδομένα εισόδου σε μια κατηγορία.

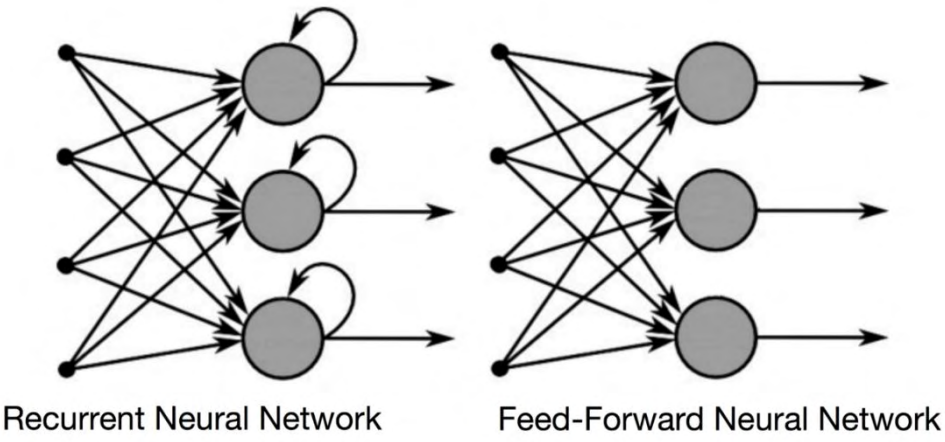
Το TensorFlow περιέχει τον Linear Classifier (ή Linear Estimator) και τον DNN Classifier. [15]

3.7 Αρχιτεκτονική Νευρωνικών Δικτύων

Βασιζόμενοι στην αρχιτεκτονική των δικτύων, μπορούν να διακριθούν σε τρείς μεγάλες κατηγορίες. Αρχικά, έχουμε την Feed-Forward αρχιτεκτονική στην οποία δεν υπάρχει σύνδεση προς τα πίσω μεταξύ των νευρώνων και το σήμα κινείται μόνο προς τα μπροστά. Κάποιοι τύποι Feed-Forward δικτύων είναι ο Fully connect neural networks, Fully Convolutional neural networks και Convolutional neural networks.

Η δεύτερη κατηγορία αποτελείται από τα Recurrent neural networks (RNN) τα οποία παρουσιάζουν feedback συνδέσεις μεταξύ των νευρώνων τους. Συγκεκριμένα, μπορούν να μάθουν προσωρινές συναρτήσεις και μοτίβα κατά την διαδικασία της μάθησης. Κάποια δίκτυα αυτής την κατηγορίας είναι το Long short term memory (LSTM), το Vanilla RNN και το Recurrent convolutional neural networks (RCNN). Οι δύο παραπάνω κατηγορίες παρουσιάζονται στην Εικόνα 11.

Συνοπτικά η τρίτη κατηγορία εμπεριέχει τα memory augmented neural networks τα οποία δεν θα χρησιμοποιηθούν στην παρούσα εργασία.[16]



Εικόνα 11. Τα δύο βασικά είδη των νευρωνικών δικτύων

Το DNN (deep neural network) αποτελεί ένα νευρωνικό δίκτυο του Tensorflow το οποίο αποτελεί ένα feedforward multilayer νευρωνικό δίκτυο το οποίο εκπαιδεύεται σε κάποια κατηγοριοποιημένα δεδομένα με σκοπό να εκτελέσει κάποιο classification σε άλλα δεδομένα άγνωστα προς αυτό. Συγκεκριμένα ειδικεύεται σε μεγάλα νευρωνικά δίκτυα με πολλά layers.

Το CNN (Convolutional neural network) αποτελεί ένα νευρωνικό δίκτυο feedforward με κύριο σκοπό να κατηγοριοποιήσει εικόνες. Συγκεκριμένα εκτελεί classification που ειδικεύεται στην ανάλυση εικόνων αναγνωρίζοντας και αναλύοντας ειδικά μοτίβα κατά τη διάρκεια της εκπαίδευσης του.

Το RNN (recurrent neural network) αποτελεί ένα νευρωνικό δίκτυο το οποίο σε αντίθεση με τα προηγούμενα δεν έχει feedforward λειτουργία αλλά δημιουργεί feedback συνδέσεις μεταξύ των νευρώνων. Συγκεκριμένα χρησιμοποιεί ένα είδος μνήμης μέσω της οποίας έχει την ικανότητα να αποθηκεύει προηγούμενες ακολουθίες που έχουν εισαχθεί σε αυτό. Ειδικεύεται σε Next word prediction, Music composition, Speech recognition και Stock market prediction

Εκτός από το keras, αποφασίστηκε σημαντικό να γίνουν προσεγγίσεις και με μοναδικό εργαλείο το TensorFlow. Οι classifier που χρησιμοποιήθηκε είναι ο DNNClassifier. Το Tensorflow ορίζει τους classifiers ως estimators δηλαδή tf.estimator.DNNClassifier tf.estimator.LinearClassifier.vO DNNClassifier αφορά τα deep neural networks δηλαδή νευρωνικά δίκτυα με πολλούς νευρώνες και πιο σύνθετη αρχιτεκτονική

3.8 Η Ανάπτυξη του Νευρωνικού Δικτύου

Η υλοποίηση του νευρωνικού δικτύου, περιλαμβάνει τέσσερα βασικά στάδια τα οποία αναλύονται εκτενέστερα στην συνέχεια:

1. Model Definition

Ξεκινώντας πρέπει να γίνει η κατασκευή του νευρωνικού δικτύου. Με τον όρο κατασκευή εννοείται επιλεχθεί η αρχιτεκτονική του δικτύου, δηλαδή η διασύνδεση μεταξύ των layers, ο τύπος των layers, καθώς και το πλήθος τους όπως και το πλήθος των απαιτούμενων εισόδων των δεδομένων αλλά και το πλήθος των εξόδων. Παράλληλα με την προηγούμενη διαδικασία, καθορίζεται και το activation function όπως και το input_shape του νευρωνικού.

1. Compile

Στην συνέχεια καθώς έχουν οριστικοποιηθεί οι παράμετροι του πρώτου σταδίου, υλοποιείται το compile του μοντέλου. Η διαδικασία αυτή απαιτεί να οριστεί ένας optimizer (κεφάλαιο 3.4.4), loss function (κεφάλαιο 3.4.3), metrics τα οποία κρίνουν την απόδοση του μοντέλου. Τα τρία αυτά στοιχεία θα πρέπει να επιλεχθούν κατάλληλα ώστε να προσαρμόζονται στα δεδομένα της εκάστοτε πειραματικής διαδικασίας.

2. Train

Στο στάδιο αυτό πραγματοποιείται η εκπαίδευση του μοντέλου στα εισερχόμενα δεδομένα. Παράλληλα ορίζονται τα δεδομένα αυτά καθώς και οι προσδοκόμενοι στόχοι. Επίσης ορίζονται τα δεδομένα και οι προσδοκόμενοι στόχοι για τη διαδικασία του validation.

3. Evaluation

Συμπερασματικά, έχοντας αναπτύξει το νευρωνικό δίκτυο και έχοντας επιλέξει κατάλληλες παραμέτρους ώστε να γίνει αποτελεσματικά η εκπαίδευση του πραγματοποιείται η αξιολόγηση του, Δηλαδή η τροφοδότηση του με νέα δεδομένα και ο έλεγχος της ακρίβειας του πάνω σε αυτά.

Κεφάλαιο 4

Η υλοποίηση του Νευρωνικού Δικτύου

Στο παρόν κεφάλαιο θα αναλυθεί η μεθοδολογία που ακολουθήθηκε για την επίτευξη του τελικού σκοπού της εργασίας, δηλαδή της ανάπτυξης νευρωνικού δικτύου με ικανότητα κατηγοριοποίησης με βάση την οδηγική συμπεριφορά.Το κεφάλαιο θα ξεκινήσει με τον τρόπο καταγραφής των δεδομένων, ανάπτυξη χαρακτηριστικών στα ήδη υπάρχοντα δεδομένα και τελικά θα καταγραφεί όλη η διαδικασία δόμησης τους δικτύου λαμβάνοντας υπόψιν τα οδηγικά δεδομένα.

4.1 Καταγραφή Δεδομένων

Όσον αφορά τη συλλογή των δεδομένων χρησιμοποιήθηκαν εργαλεία διαγνωστικού ελέγχου ή αλλιώς OBD-II. Το λογισμικό των OBD διαχειρίζεται και παρακολουθεί τις κύριες διαδικασίες του αυτοκινήτου. Πιο συγκεκριμένα, μπορεί να στείλει εντολές σε διαφορετικά συστήματα με σκοπό να διατηρήσει την υγεία του οχήματος. Επίσης μπορεί να διορθώσει πιθανά μηχανικά σφάλματα αλλά και θέματα καυσίμου.[17]

Για να κατανοηθούν οι συγκεντρωμένες πληροφορίες, είναι υποχρεωτικό να υπάρχει ένα OBD scanner. Το μηχάνημα αυτό μπορεί να αποθηκεύσει δεδομένα του αυτοκινήτου όπως: engine RPM, speed, air temperature, fuel, acceleration, braking και άλλα.

Υπάρχουν δυο είδη OBD scanner:

1. Code readers

   Είναι φθηνά μηχανήματα που μπορούν να διαβάσουν κώδικα από το όχημα. Παρ’ ότι είναι φθηνό έχει κάποιους περιορισμούς. Συγκεκριμένα, παρουσιάζει έλλειψη πληροφοριών σε συγκεκριμένους κώδικες και δεν έχει πρόσβαση σε όλους τους τύπους αρχείων.

2. Scan tools

   Είναι ακριβά μηχανήματα τα οποία έχουν πρόσβαση σε περισσότερα χαρακτηριστικά. Παρέχει real-time καταγραφή όλων των δεδομένων και μπορεί να διαβάσει οποιονδήποτε κώδικα χρειάζεται και να ενεργοποιήσει επιλογές.

Διαδικασία χρήσης

1. Αρχικά έχοντας απενεργοποιημένο το όχημα, τοποθετείται το scanner στο OBD2 data link connector(τύπου SAE J1962) το οποίο βρίσκεται στον πίνακα ελέγχου του αυτοκινήτου.
2. Ενεργοποίηση του οχήματος και boot του OBD
3. Εισαγωγή στο scanner όλων των παραμέτρων του οχήματος
4. Αποδοχή στο scanner να μπορεί να διαβάσει το διαγνωστικό σύστημα του οχήματος
5. Σκανάρισμα του συστήματος του αυτοκινήτου και συλλογή των επιθυμητών πληροφοριών.

4.2 Προετοιμασία Δεδομένων

4.2.1 Raw Δεδομένα

Στην περίπτωση μας , τα raw δεδομένα είναι τα δεδομένα που παίρνουμε απευθείας απο την συσκευή obd χωρίς καμία επεξεργασία.Τα δεδομένα αυτά είναι τα εξής :

- Speed - Ταχύτητα

  Απεικονίζει την μέση ταχύτητα σε ενα χρόνικό διάστημα 15 δευτερολέπτων σε χιλιόμετρα ανά ώρα.

- Driving Qualities

  Το Driving Qualities αποτελείτα απο 21 αριθμούς (ακέραιες τιμές) .Ξεκινώντας απο το DQ0 με επιτάχυνση -0.5g και καταλήγοντας στο DQ20 με επιτάχυνση 0.5g και με ενδιάμεσα βήμα 0.05g κάθε μεταβλητή μετράει το κατα πόσες φορές έχει εμφανιστεί η συγκεκριμένη επιτάχυνση μέσα σε ενα συγκεκριμένα χρονικό διάστημα.

- WLN_ACCL_MAX

  Η μεταβλητή αυτή ορίζει την μέγιστη επιτάχυνση σε μια συγκεκριμένη μέτρηση και υπολογίζειται ως η τιμή επί 0.12g

- WLN_BRK_MAX

  Η μεταβλητή αυτή ορίζει την μέγιστη επιβράδυνση σε μια συγκεκριμένη μέτρηση και υπολογίζεται ως η τιμή επί 0.12g

4.2.2 Ανάλυση – Εξαγωγή των Χαρακτηριστικών

Τα χαρακτηριστικά είναι δεδομένα τα οποία έχουν προέλθει απο επεξεργασία των αρχικών δεδομένων που έχουμε πάρει από το obd. Στην ουσία, έχοντας τα raw δεδομένα από την συσκευή obd, δημιουργήσαμε μαθηματικές συναρτήσεις οι οποίες δέχονται τα raw δεδομένα και τα αντιστοιχούν σε πραγματική τιμή.

Η ακριβής διαδικασία υπολογισμού των χαρακτηριστικών θα γίνει εκτενέστερα όταν παρουσιάσουμε τον κώδικα εξαγωγής τους. Αναφορικά τα χαρακτηριστικά που υπολογίζουμε είναι η ταχύτητα, η επιτάχυνση , η επιβράδυνση , ο μέσος όρος όλων των τιμών μια συγκεκριμένης χρονικής περιόδου, η διακύμανση , η μέγιστη επιτάχυνση , η μέγιστη επιβράδυνση και η ακινησία.

Η ανάγκη που μας πρότρεψε να οδηγηθούμε στην εξαγωγή δικών μας δεδομένων – χαρακτηριστικών είναι αρχικά η βελτιστοποίηση του μοντέλου μας καθώς θέλαμε να διαπιστώσουμε τις δυνατότητες του και το κατά πόσο καλύτερο μπορεί να γίνει.Αυτο προκύπτει από το γεγονός ότι το μοντέλο μπορεί να κάνει ακόμα καλύτερη κατηγοριοποίηση των δεδομένω εισόδου. Επίσης τα χαρακτηριστικά αποτελούνται από δεδομένα που ανταποκρίνονται άμεσα σε πραγματικά οδηγικά στοιχεία.

Στόχος του συγκεκριμένου κώδικα είναι η δημιουργία νέων χαρακτηριστικών από τα raw δεδομένα που δεχόμαστε από το obd. Έτσι, φτιάχνουμε οχτώ χαρακτηριστικά τα οποία χρησιμοποιούμε στη συνέχεια με σκοπό την αποδοτικότερη λειτουργία του μοντέλου μας. Πρώτα γίνεται η επεξεργασία των αρχικών δεδομένων μας κρατώντας τα στοιχεία που επιθυμούμε. Στη συνέχεια καλούνται μία μία οι συναρτήσεις για την εξαγωγή του αντίστοιχου χαρακτηριστικού. Έπειτα ενώνουμε όλα τα χαρακτηριστικά σε κοινό πίνακα και τα εξάγουμε σε ένα .csv αρχείο

```
import pandas as pd
import numpy as np
from keras import models
from keras import layers
from keras import optimizers
from keras import losses
from keras import metrics
from keras import regularizers
from keras.utils import to_categorical
from keras.layers import Flatten,Dense


def data_preparation(dataframe):
    df = dataframe[0:10000]
    driving_qua = df['driving_qualities'].str.split(",",expand=True).add_prefix('driving_')
    driving_qua=driving_qua.astype('int')
    dff=pd.concat([df[['speed','wln_accel_max','wln_brk_max','wln_crn_max']],driving_qua],1)
    return dff


def acceleration_summary(dff_Vehicle1):
    sum=np.zeros(10000).astype(int)
    k=0
    for i in range(0,10000):
        sum[i]=0
        k=0
        for j in range(14,25):
            sum[i] = sum[i] + dff_Vehicle1[i][j] * pow(10,k)
            k=k+1
    return sum

def braking_summary(dff_Vehicle1):
    sum = np.zeros(10000).astype(int)
    k=0
    for i in range(0,10000):
        sum[i]=0
        k=0
        for j in range(4,14):
            sum[i] = sum[i] + dff_Vehicle1[i][j] * pow(10,10-k)
            k=k+1
    return sum


def speed(dff_Vehicle1):
    rspeed = dff_Vehicle1[:,0]
    return rspeed

def row_mean(dff_Vehicle1):
    mean=np.zeros(10000)
    for i in range(0,10000):
        mean[i] = np.mean(dff_Vehicle1[i,:])
    return mean

def max_acc(dff_Vehicle1):
    max_ac = dff_Vehicle1[:,1]
    return max_ac

def max_bra(dff_Vehicle1):
    max_br = dff_Vehicle1[:,2]
    return max_br

def row_var(dff_Vehicle1):
    variance=np.zeros(10000)
    for i in range(0,10000):
        variance[i] = np.var(dff_Vehicle1[i,:])
    return variance
```

```
#Load and Preparation of truck data


df_Vehicle1=pd.read_csv('Vehicle1.csv')
dff_Vehicle1 = data_preparation(df_Vehicle1)

dff_Vehicle1 = dff_Vehicle1.values
print dff_Vehicle1,dff_Vehicle1.shape

#Acceleration_summary

acceleration_s = acceleration_summary(dff_Vehicle1).astype(float)
acceleration_s = acceleration_s.reshape(10000,1)
print acceleration_s

#Braking_summary

braking_s = braking_summary(dff_Vehicle1).astype(float)
braking_s = braking_s.reshape(10000,1)
print braking_s

#Speed

rspeed = speed(dff_Vehicle1).astype(float)
rspeed = rspeed.reshape(10000,1)
print rspeed

#Mean

mean = row_mean(dff_Vehicle1).astype(float)
mean = mean.reshape(10000,1)
print mean

#Max_acceleration

max_acce = max_acc(dff_Vehicle1).astype(float)
max_acce = max_acce.reshape(10000,1)
print max_acce

#Max_braking

max_br = max_bra(dff_Vehicle1).astype(float)
max_br = max_br.reshape(10000,1)
print max_br

#Variance

variance = row_var(dff_Vehicle1).astype(float)
variance = variance.reshape(10000,1)
print variance

final = np.concatenate((acceleration_s,braking_s),axis=1)
final = np.concatenate((final,rspeed),axis = 1)
final = np.concatenate((final,mean),axis = 1)
final = np.concatenate((final,max_acce),axis = 1)
final = np.concatenate((final,max_br),axis = 1)
final = np.concatenate((final,variance),axis = 1)
print final

np.savetxt("feautures_val_ix.csv", final, delimiter=",")
```

Εκτενέστερα έχουμε:

- Βιβλιοθήκες: pandas,numpy
- Συναρτήσεις:

  - def data_preparation(dataframe): Περνάμε ως όρισμα το dataframe και πραγματοποιούμε την αρχική επεξεργασία των δεδομένων μας. Συγκεκριμένα χωρίζουμε σε στήλες τις επιταχύνσεις μας οι οποίες ήταν αρχικά δωσμένες με κόμματα, έπειτα τραβάμε τις στήλες που επιθυμούμε και επιστρέφουμε τις εξής: speed,wln_accel_max,wln_brk_max,wln_crn_max,driving_qua

  - def acceleration_summary(dataframe_): Περνάμε ως όρισμα το dataframe και χρησιμοποιώντας τις επιταχύνσεις σε κάποιο χρονικό διάστημα

  - def braking summary(dataframe_): Περνάμε ως όρισμα το dataframe και χρησιμοποιώντας τις επιβραδύνσεις σε κάποια χρονικό διάστημα

  - def speed(dataframe_): Περνάμε ως όρισμα το dataframe και παίρνουμε την αντίστοιχη στήλη απο τα raw δεδομένα της ταχύτητας

  - def row_mean(dataframe_): Περνάμε ως όρισμα το dataframe και υπολογίζουμε το μέσο όρο όλων των γραμμών ξεχωριστά

  - def max_acc(dataframe_): Περνάμε ως όρισμα το dataframe και τραβάμε την στήλη με τις μέγιστες επιταχύνσεις απο τα raw δεδομένα

  - def max_bra(dataframe_): Περνάμε ως όρισμα το dataframe και παίρνουμε την στήλη με τις μέγιστες επιβραδύνσεις απο τα raw δεδομένα
  - def row_var(dataframe_): Περνάμε ως όρισμα το dataframe και βρίσκουμε τη διακύμανση όλων των γραμμών ξεχωριστά

4.3 Το Νευρωνικό Δίκτυο στις Ανάγκες των Δεδομένων

Η ανάπτυξη του κώδικα και κάτ. επέκταση του νευρωνικού δικτύου ξεκινάει από τη συλλογή και την επεξεργασία των δεδομένων ώστε να εισαχθούν στην κατάλληλη μορφή. Τι όμως σημαίνει αυτό; Αρχικά, μιας και τα δεδομένα που παίρνονται είναι από πραγματικά οχήματα σε πραγματικούς χρόνους δεν έχουν όλα το ίδιο πλήθος εγγραφών. Έτσι, έχοντας αυτοκίνητο, απορριμματοφόρο, σκούπα και φορτηγό οι γραμμές του κάθε αρχείου ήταν διαφορετικές. Αυτό αποτελεί πρόβλημα στο νευρωνικό μιας και οι εισερχόμενες εγγραφές κάθε οχήματος οφείλουν να είναι οι ίδιες, για αυτό το λόγο από κάθε όχημα επιλέχθηκαν οι ελάχιστες δυνατές γραμμές ώστε να βρίσκονται σε ισορροπία. Για το διαχωρισμό των δεδομένων μεταξύ training, validation και test δεν υπάρχει κάποιος κανόνας με συγκεκριμένα ποσοστά, αλλά απαιτούνται πολλές δοκιμές και κατανομές ώστε να βρεθεί τελικά το ιδανικό ζευγάρι δεδομένων χωρίς να παρουσιάζεται underfitting ή overfitting. Στη συνέχεια και όσον αφορά το νευρωνικό δίκτυο αλλά και την αρχιτεκτονική του, στην αρχική προσέγγιση χρησιμοποιήθηκαν οι ελάχιστες δυνατοί νευρώνες με σκοπό την παρατήρηση της απόδοσης του μοντέλου. Έπειτα, προσθέτοντας σταδιακά νευρώνες αλλά και εσωτερικά units εντοπίστηκε η χρυσή τομή, δηλαδή μια ισορροπία ώστε να είναι αρκετοί αλλά όχι πάρα πολλοί για να αποφευχθεί το overfitting. Σχετικά με την αρχιτεκτονική επιλέχθηκε ένα βασικό feed-forward νευρωνικό με τον default classifier του keras βρίσκοντας training,validation accuracy αλλά και τεστάροντας το μοντέλο σε νέα δεδομένα με το test accuracy. Το classification που έχει επιλεχθεί αποτελεί μια δυαδική περίπτωση (binary classification) δηλαδή επιλέγονται όλοι οι συνδυασμοί των τεσσάρων οχημάτων. Στη συνέχεια και αφού έχει επιλεχθεί το αρχικό μοντέλο γίνονται οι απαραίτητες τροποποιήσεις στα εισερχόμενα δεδομένα. Ξεκινώντας χρησιμοποιούνται τα δεδομένα σε raw μορφή, αφού έχουν τοποθετηθεί σε πίνακες, όπως δηλαδή έχουν εξαχθεί από τα μηχανήματα. Παρατηρώντας την απόδοση του μοντέλου γίνεται η κατάλληλη επεξεργασία ώστε το νευρωνικό δίκτυο να έχει την ικανότητα να επεξεργαστεί αποδοτικότερα τα δεδομένα. Κάποιες τεχνικές που χρησιμοποιήθηκαν ως προς την επεξεργασία των δεδομένων είναι η κανονικοποίηση (κεφάλαιο 3.5.5) και το one hot (κεφάλαιο 3.5.5). Χωρίς να μένουμε στον τύπο των δεδομένων που εξέρχονται απο το obd, θεωρήθηκε σημαντικό και αποδοτικό να εξαχθούν χαρακτηριστικά δηλαδή νέα δεδομένα τα οποία εξάγονται με την χρήση συναρτήσεων από τα αρχικά. Παράλληλα, μια άλλη τεχνική που αναπτύχθηκε για την καλύτερη αξιοποίηση των δεδομένων από το νευρωνικό δίκτυο ονομάστηκε window και αποτελεί μια μέθοδο στην οποία ομαδοποιούνται ανά δύο ή πέντε συνεχόμενες εγγραφές Αναφορικά με τον τύπο της αρχιτεκτονικής του δικτύου, προφανώς δε σταμάτησε στην απλή περίπτωση και στον stock classifier αλλά δημιουργήθηκαν παραλλαγές και επιπλέον δομές. Έτσι, με τις κατάλληλες διεργασίες εκτελέστηκαν υλοποιήσεις με δίκτυα CNN, RNN, DNN, με όλους τους τύπους δεδομένων, ελέγχοντας παράλληλα τη συμπεριφορά και την ακρίβεια του μοντέλου σε κάθε περιβάλλον.

Κεφάλαιο 5

Κώδικες και Αποτελέσματα

Οι κώδικες οι οποίοι γράφτηκαν και αναπτύχθηκαν για την αξιολόγηση της οδηγικής συμπεριφοράς, είναι ολόκληρα καταγεγραμμένοι στο Παράρτημα. Στη συγκεκριμένη ενότητα θα παρατεθούν και θα αναλυθούν τα μέρη του κώδικα τα οποία παραμετροποιήθηκαν με κατάλληλο τρόπο ώστε να προσαρμόζονται στα δεδομένα τα οποία έχουν συλλεχθεί. Στη συνέχεια θα παρουσιαστούν τα αποτελέσματα από τις εκτελέσεις των πειραμάτων. Ολοκληρώνοντας θα γίνουν συγκρίσεις και αναλύσεις πάνω στις διαφορετικές εισόδους που δέχεται το νευρωνικό δίκτυο.

5.1 Οι κώδικες που Αναπτύχθηκαν

Για την ανάλυση του κάθε κώδικα θα ακολουθηθεί η ίδια διαδικασία όπως στο κεφάλαιο 3.8. Η διαδικασία αυτή μας βοηθάει στην βαθύτερη κατανόηση του μοντέλου καθώς αναλύεται το κάθε κομμάτι – λειτουργία του μοντέλου ξεχωριστά.

Σε όλους τους κώδικες χρησιμοποιήθηκε ο Nadam ως optimizer και binary_crossentropy ως loss function. Για metrics χρησιμοποιήθηκε binary accuracy.

Οι κώδικες χωρίστηκαν σε δύο πίνακας. Ο πρώτος πίνακα περιέχει όλα τα νευρωνικά και τους τρόπους επεξεργασίας δεδομένων με είσοδο τα δεδομένα που παίρνουμε απευθείας απο το obd(Πίνακας 1). Ο δεύτερος πίνακας περιέχει αποτελέσματα δικτύων που δέχονται ως είσοδο τα χαρακτηριστικά που αναπτύχθηκαν (Πίνακας 2).

Raw Data

Πίνακας 1. Χρήση Raw Data στα μοντέλα που αναπτύχθηκαν

Αριθμός	Αρχιτεκτονική	API	Επεξεργασία	Window
1	Fully Connected Feed Forward	Keras	Χωρίς	1
2	Fully Connected Feed Forward	Keras	One Hot	1
3	Fully	Keras	Normalization	1

	Connected Feed Forward			
4	CNN Feed Forward	Keras	One Hot	1
5	RNN Feedback	Keras	Normalization	1
6	DNN Estimator	TensorFlow	Normalization	1
7	Fully Connected Feed Forward	Keras	Χωρίς	2
8	Fully Connected Feed Forward	Keras	Χωρίς	5

Features Data

Πίνακας 2. Χρήση Feature Data στα μοντέλα που αναπτύχθηκαν

Αριθμός	Αρχιτεκτονική	API	Επεξεργασία	Window
1	Fully Connected Feed Forward	Keras	Χωρίς	1
2	Fully Connected Feed Forward	Keras	Χωρίς	2
3	Fully Connected Feed Forward	Keras	Χωρίς	5

Στην συνέχεια παρουσιάζονται τα στάδια που απαιτούνται τόσο για την προετοιμασία των δεδομένων, την επεξεργασία τους, τις μεθόδους που χρησιμοποιήθηκαν όσο και για την διαδικασία που απαιτείται για την ανάπτυξη του νευρωνικού δικτύου όπως παρουσιάστηκε στο κεφάλαιο 3.8

Στάδιο Πρώτο

Σε όλους τους παραπάνω κώδικες, έχει γραφτεί ένα σύνολο εντολών το οποίο αποτελεί σταθερό κομμάτι σε όλους. Το συγκεκριμένο κομμάτι κώδικα έχει ως σκοπό αρχικά να εισάγει τα δεδομένα από τα csv σε data frames, να διαχωρίσει το επιθυμητό ποσοστό των train data, validation data, test data για την κάθε δυαδική κατηγοριοποίηση οχήματος. Στην συνέχεια, ενώνει τα αντίστοιχα data frames των δύο οχημάτων και τα μετατρέπει σε arrays με σκοπό την σωστή εισαγωγή τους στο νευρωνικό δίκτυο.

Αρχικά για το πρώτο όχημα ακολουθείται η παρακάτω διαδικασία :

```
def data_preparation(dataframe):

    df = dataframe["number_of_csv_tuples"]

    driving_qua = df['driving_qualities'].str.split(",",expand=True).add_prefix('driving_')
    driving_qua=driving_qua.astype('int')
    dff=pd.concat([df[['wln_accel_max','wln_brk_max','speed']],driving_qua],1)
    return dff

#Load and Preparation of Vehicle1 data


df_Vehicle1=pd.read_csv('Vehicle1.csv')

dff_Vehicle1 = data_preparation(df_Vehicle1)

#Train_Vehicle1
train_size = "percentage of train_data"*len(dff_Vehicle1)
train_size = int(train_size)

train_data_Vehicle1 = dff_Vehicle1[:train_size]
train_labels_Vehicle1 = np.ones((train_size),dtype=int)

#Validate_Vehicle1
val_size = "percentage of validate_data*len(dff_Vehicle1)
val_size = int(val_size)

val_data_Vehicle1 = dff_Vehicle1[train_size:train_size+val_size]
val_labels_Vehicle1 = np.ones((val_size),dtype=int)

#Test_Vehicle1
test_size = "percentage of test_data"*len(dff_Vehicle1)
test_size = int(test_size)

test_data_Vehicle1 = dff_Vehicle1[train_size+val_size:len(dff_Vehicle1)]

test_labels_Vehicle1 = np.ones((test_size),dtype=int)
```

Για το δεύτερο όχημα ακολουθείται η παρακάτω διαδικασία:

```
#Load and Preparation of Vehicle2 data


df_Vehicle2=pd.read_csv('Vehicle2.csv')

dff_Vehicle2 = data_preparation(df_Vehicle2)

#Train_Vehicle2

train_data_Vehicle2 = dff_Vehicle2[:train_size]
train_labels_Vehicle2=np.zeros((train_size),dtype=int)

#Validate_Vehicle2

val_data_Vehicle2 = dff_Vehicle2[train_size:train_size+val_size]
val_labels_Vehicle2 = np.zeros((val_size),dtype=int)

#Test_Vehicle2

test_data_Vehicle2 = dff_Vehicle2[train_size+val_size:len(dff_Vehicle2)]
test_labels_Vehicle2 = np.zeros((test_size),dtype=int)

#Tables merge train

frames_train = [train_data_Vehicle2,train_data_Vehicle1]
merge_train_data = pd.concat(frames_train)
merge_train_labels = np.concatenate((train_labels_Vehicle2,train_labels_Vehicle1),axis=0)

#Table merge validate

frames_val = [val_data_Vehicle2,val_data_Vehicle1]
merge_val_data = pd.concat(frames_val)
merge_val_labels = np.concatenate((val_labels_Vehicle2,val_labels_Vehicle1),axis=0)

#Table merge test

frames_test = [test_data_Vehicle2,test_data_Vehicle1]
merge_test_data = pd.concat(frames_test)
merge_test_labels = np.concatenate((test_labels_Vehicle2,test_labels_Vehicle1),axis=0)

#Metatropi Dataframe se Array


merge_train_data = merge_train_data.values
merge_val_data = merge_val_data.values
merge_test_data = merge_test_data.values
```

Στάδιο Δεύτερο

Σε περίπτωση, που θέλουμε να πάρουμε τα οδηγικά δεδομένα και να τα ομαδοποιήσουμε (windows) τότε έχει αναπτυχθεί ξεχωριστό κομμάτι κώδικα το οποίο παράγει τα νέα ομαδοποιημένα δεδομένα για να χρησιμοποιηθούν ως είσοδο στο νευρωνικό. Τα νέα δεδομένα έχουν μικρότερα πλήθος από τα αρχικά μας δεδομένα. Συγκεκριμένα, εάν τα δεδομένα έχουν window 2 τότε τα νέα δεδομένα θα είναι τα μισά από τα αρχικά. Παραθέτονται οι κώδικες για ομαδοποίηση ανά δύο και ανά πέντε.

Window/2

```
import pandas as pd
import numpy as np

def data_preparation(dataframe):
        df = dataframe["tuples_of_csv"]
        driving_qua = df['driving_qualities'].str.split(",",expand=True).add_prefix('driving_')
        driving_qua=driving_qua.astype('int')
        dff=pd.concat([df[['speed','wln_accel_max','wln_brk_max','wln_crn_max']],driving_qua],1)
        return dff



df_Vehicle=pd.read_csv('Vehicle.csv')

dff_Vehicle = data_preparation(df_Vehicle)
dff_Vehicle = dff_Vehicle.values

dff_Vehicle = dff_Vehicle.reshape("tuples_of_csv/2",50)

np.savetxt("window_Vehicle.csv", dff_Vehicle, delimiter=",")
```

Window/5

```
import pandas as pd
import numpy as np

def data_preparation(dataframe):
        df = dataframe["tuples_of_csv"]
        driving_qua = df['driving_qualities'].str.split(",",expand=True).add_prefix('driving_')
        driving_qua=driving_qua.astype('int')
        dff=pd.concat([df[['speed','wln_accel_max','wln_brk_max','wln_crn_max']],driving_qua],1)
        return dff



df_Vehicle=pd.read_csv('Vehicle.csv')

dff_Vehicle = data_preparation(df_Vehicle)
dff_Vehicle = dff_Vehicle.values

dff_Vehicle = dff_Vehicle.reshape("tuples_of_csv/5",125)

np.savetxt("window_Vehicle.csv", dff_Vehicle, delimiter=",")
```

Στάδιο Τρίτο

Στην συνέχεια, αφού έχει γίνει η κατάλληλη επεξεργασία των οδηγικών δεδομένων, θα πρέπει να επεξεργαστούν για να αυξηθεί η ακρίβεια του δικτύου. Σε αυτό το στάδιο έχουμε τρείς τρόπους επεξεργασίας. Ο πρώτος είναι να μπουν τα δεδομένα όπως είναι (συνεπώς δεν υπάρχει ξεχωριστό κομμάτι επεξεργασίας), ο δεύτερος με κανονικοποίηση και ο τρίτος να μετατραπούν τα δεδομένα σε 0 και 1 (One hot). Παρατίθενται οι κώδικες για την κανονικοποίηση και το One hot :

Κανονικοποίηση – Normalization

```
#Normalization


merge_train_data = merge_train_data.astype(float)

mean = merge_train_data.mean(axis=0)
merge_train_data -= mean
std= merge_train_data.std(axis=0)
for i in range(len(std)):
    if(std[i] == 0):
        std[i]=1
merge_train_data /= std

merge_val_data = merge_val_data.astype(float)

mean = merge_val_data.mean(axis=0)
merge_val_data -= mean
std= merge_val_data.std(axis=0)
for i in range(len(std)):
    if(std[i] == 0):
        std[i]=1
merge_val_data /= std

merge_test_data = merge_test_data.astype(float)

mean = merge_test_data.mean(axis=0)
merge_test_data -= mean
std= merge_test_data.std(axis=0)
for i in range(len(std)):
    if(std[i] == 0):
        std[i]=1
merge_test_data /= std
```

One Hot

```
#ONE HOT

temp = np.zeros((train_size*2,24,300))
print temp.shape
for i in range(0,train_size*2):
    temp[i]=one_hot(merge_train_data[i],300)
merge_train_data=temp
temp = np.zeros((val_size*2,24,300))
for i in range(0,val_size*2):
    temp[i]=one_hot(merge_val_data[i],300)
merge_val_data=temp
temp = np.zeros((test_size*2,24,300))
for i in range(0,test_size*2):
    temp[i]=one_hot(merge_test_data[i],300)
merge_test_data=temp

#preparation for layers
merge_train_data=merge_train_data.reshape(train_size*2,24*300)
merge_val_data=merge_val_data.reshape(val_size*2,24*300)
merge_test_data=merge_test_data.reshape(test_size*2,24*300)
```

Στάδιο Τέταρτο

Το τέταρτο και τελευταίο στάδιο, αναλύθηκε θεωρητικά στο κεφάλαιο 3.8. Στην παρούσα ενότητα θα παρουσιαστεί η υλοποίηση του μοντέλου και ο κώδικας που γράφτηκε για κάθε αρχιτεκτονική νευρωνικού δικτύου ξεχωριστά.

Fully Connected Feed Forward

```
#Model Definition

model=models.Sequential()

model.add(layers.Dense(64,kernel_regularizer=regularizers.l2(0.001),activation='relu',input_shape=(24,)))
#model.add(layers.Dropout(0.5))
model.add(layers.Dense(64,kernel_regularizer=regularizers.l2(0.001),activation='relu'))
#model.add(layers.Dropout(0.5))
model.add(layers.Dense(1,activation='sigmoid'))

#Compiling the model and configuring optimizer,loss and metrics

model.compile(optimizer=optimizers.Nadam(),loss='binary_crossentropy',metrics=[metrics.binary_accuracy])

#Training the model

history=model.fit(merge_train_data,merge_train_labels,epochs=30,batch_size=512,validation_data=(merge_val_data,merge_val_labels))

#Evaluating the model

test_loss,test_accu = model.evaluate(merge_test_data,merge_test_labels)
```

CNN

```
#Merge One Hot

merge_train_onehot=to_categorical(merge_train_data)
merge_train_onehot=merge_train_onehot.reshape((9206,24,25,1))

merge_val_onehot=to_categorical(merge_val_data)
merge_val_onehot=merge_val_onehot.reshape((2394,24,18,1))

merge_tlabels_onehot = np.asarray(merge_train_labels).astype('float32')
merge_vlabels_onehot = np.asarray(merge_val_labels).astype('float32')

#Model Definition

model=models.Sequential()

model.add(layers.Conv2D(32,(3,3),activation='relu',input_shape=(24,25,1)))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(64,(3,3),activation='relu'))
model.add(layers.MaxPooling2D((2,2)))
model.add(layers.Conv2D(64,(3,3),activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64,activation='relu'))
model.add(layers.Dense(1,activation='sigmoid'))

#Compiling the model and configuring optimizer,loss and metrics

model.compile(optimizer=optimizers.Nadam(),loss='binary_crossentropy',metrics=[metrics.binary_accuracy])

#Training the model

history=model.fit(merge_train_onehot,merge_tlabels_onehot,epochs=30,batch_size=512,validation_data=(merge_val_onehot,merge_vlabels_onehot))

#Evaluating the model

test_loss,test_accu = model.evaluate(merge_test_onehot,merge_testlabels_onehot)
```

RNN

```
#Model Definition

merge=sequence.pad_sequences(merge,maxlen=27)
model=models.Sequential()

model.add(Embedding(27,64))
#model.add(layers.Dropout(0.2))
model.add(SimpleRNN(64))
#model.add(layers.Dropout(0.2))
model.add(Dense(1,activation='sigmoid'))

#Compiling the model and configuring optimizer,loss and metrics

model.compile(optimizer=optimizers.Nadam(),loss='binary_crossentropy',metrics=[metrics.binary_accuracy])

#Training the model

history=model.fit(merge,merge_labels,epochs=30,batch_size=512,validation_data=()

history=model.fit(merge_train_data,merge_train_labels,epochs=30,batch_size=512,validation_data=(merge_val_data,merge_val_labels))
```

DNN

Η υλοποίηση του DNN επειδή είναι μακροσκελής και αποτελείται απο εκτενές κομμάτι κώδικα, η παρουσίαση του θα γίνει στο Παράρτημα.

5.2 Αποτελέσματα

Raw	Classifier	Data Preparation	Training Accuracy / Validation Accuracy
	Keras	Normalization	74.4% / 78%
	Keras	Onehot	76.4%/79%
	Keras	No Preparation	75.5% / 78%
	CNN - Keras	onehot	77.3%/78.4%
	DNN	Normalization	76.9%/78.1%
	RNN	No preparation	75.4 % / 77.53%
	Keras/2	No preparation	85.55% / 85.34%
	Keras/3	No preparation	86.54% / 85.37%
	Keras/5	No preparation	88.95% / 88.71%
Χαρακτηριστικά			
	Keras	Normalization	80% / 78%
	Keras/2	No preparation	80 % / 81%
	Keras/3	No preparation	84% / 84 %
	Keras/5	No preparation	88%/ 88%

Κεφάλαιο 6

Συμπεράσματα

Στο συγκεκριμένο κεφάλαιο θα προσπαθήσουμε να συνοψίσουμε τα πιο βασικά αποτελέσματα του μοντέλου σε πίνακες και διαγράμματα με σκοπό να οδηγηθούμε σε συμπεράσματα για την συμπεριφορά του και για την αποδοτικότητα του.Το μοντέλο που θα χρησιμοποιήσουμε για τις δοκιμές είναι το βασικό μοντέλο που αναπτύξαμε, δηλαδή χωρίς τις παραμετροποιήσεις που έγιναν στα κεφάλαια 3.5.4, 4**.** Οποιαδήποτε αλλαγή χρειαστεί να γίνει για τα αποτελέσματα θα αναγράφεται όπως και τα δεδομένα που χρησιμοποιούνται κάθε φορά.

6.1 Συμπεράσματα από τα Πειράματα

Στο κεφάλαιο 5.2 φαίνονται όλα τα αποτελέσματα από τις πειραματικές διαδικασίες που εφαρμόστηκαν. Υπενθυμίζεται πως τα πειράματα και όπως είναι φανερό τα διαφορετικά αποτελέσματα προκύπτουν καθώς γίνονται αλλαγές τόσο στα δεδομένα όσο και στη μορφή του νευρωνικού δικτύου αλλά και στις παραμέτρους που αντιπροσωπεύει. Για παράδειγμα είναι πολύ σημαντικό να αναφερθεί η ουσιώδης διαφορά στην ακρίβεια όσον αφορά το απλό μοντέλο του Keras με τα Raw δεδομένα και την χρήση χαρακτηριστικών στο ίδιο ακριβώς μοντέλο. Παρατηρείται μια αύξηση του ποσοστού της τάξεως του 5%. Η αύξηση αυτή οφείλεται στο ότι τα χαρακτηριστικά, τα οποία έχουν εξαχθεί από εμάς, δηλαδή είναι τιμές επεξεργασμένες μέσα απο μαθηματικούς τύπους έχουν τη δυνατότητα να ενσωματωθούν ευκολότερα από το μοντέλο. Αυτό σημαίνει πως το νευρωνικό δίκτυο μπορεί να εντοπίσει αποδοτικότερα τις διαφορές όσον αφορά τα δεδομένα των δύο οχημάτων αλλά και τα πιθανά μοτίβα που θα εμφανιστούν.

Ακόμα πιο σαφής είναι η ποσοστιαία διαφορά στην ακρίβεια στην περίπτωση της μορφής των windows. Κατά την εκπαίδευση παρατηρείται το ποσοστό να εκτοξεύεται στην τάξη του 85% ενώ στο αντίστοιχο χωρίς αυτά να είναι 75%. Παράλληλα και για αντίστοιχους λόγους με προηγουμένως το μοντέλο φαίνεται να προσαρμόζεται καλύτερα στις ομαδοποιημένες εγγραφές μιας και αφομοιώνει τα patterns που δημιουργούνται. Εξίσου σημαντικό είναι ότι όσο μεγαλώνει το εύρος των παραθύρων αυξάνεται και η ακρίβεια.

Όσον αφορά τη διαφοροποίηση με βάση τους classifiers παρατηρούμε πως αποδοτικότερος είναι ο CNN (convolutional neural network), ακολουθεί ο DNN, RNN, Keras. Όπως φαίνεται οι διαφορές που προκύπτουν είναι αρκετά μικρές.

Αυτό συμβαίνει επειδή τα οδηγικά μας δεδομένα αποτελούνται από πραγματικούς αριθμούς. Ειδικότερα ο CNN ειδικεύεται στην αναγνώριση εικόνων ενώ ο RNN στο time series prediction κάτι που δε σχετίζεται άμεσα με τα δεδομένα μας.

Τελειώνοντας τα συμπεράσματα είναι σημαντικό να τονιστεί πως μέσω των μετασχηματισμών που εφαρμόστηκαν αλλά και ειδικότερα των καινούριων μεθόδων που χρησιμοποιήσαμε για να εξάγουμε νέα δεδομένα υπήρξε μεγάλη βελτίωση στην ακρίβεια του μοντέλου. Ο χρόνος υλοποίησης ήταν μεγάλος αλλά τελικά αποτελεσματικός αφού οι αυξήσεις στην απόδοση από 10%-15% είναι ιδιαίτερα ενθαρρυντικές για περαιτέρω βελτιώσεις έξω από τα πλαίσια της παρούσας διπλωματικής εργασίας.

6.2 Έλεγχος για Θόρυβο στα Χαρακτηριστικά

Λόγω του μεγάλου πλήθους δεδομένων και ακριβέστερα λόγω των διαφορετικών οδηγικών παραμέτρων που δέχεται το νευρωνικό, υπάρχει η περίπτωση κάποια ή και κάποιες από τις παραμέτρους να δημιουργούν θόρυβο. Ως θόρυβος νοείται τα δεδομένα τα οποία χρησιμοποιούνται ως είσοδοι στο νευρωνικό και επηρεάζουν την ακρίβεια του δικτύου μειώνοντάς την. Ο λόγος για τον οποίο μπορεί να δημιουργηθεί θόρυβος είναι ο εξής:

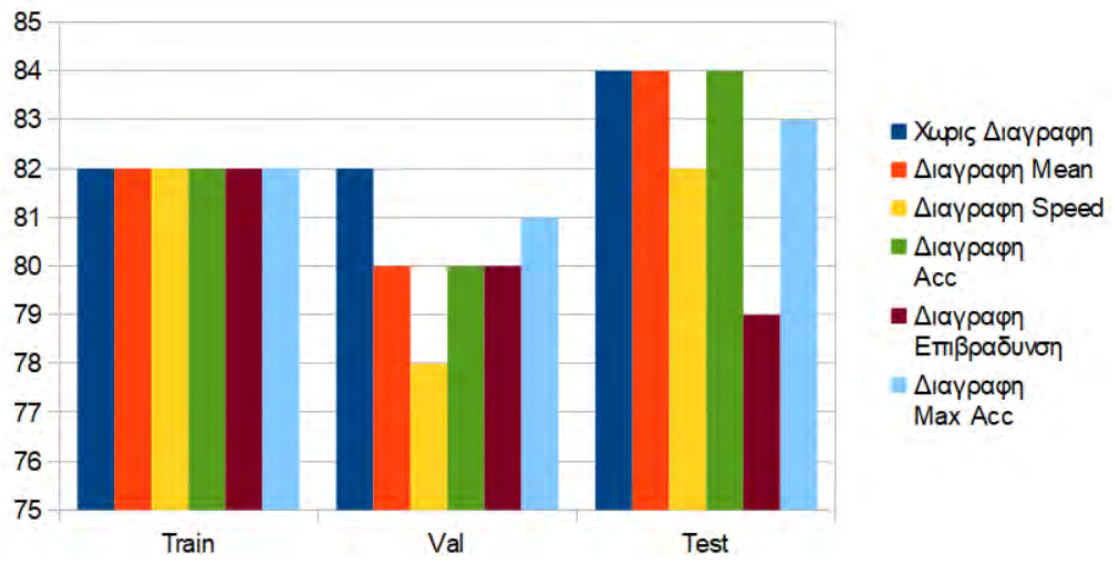
- Οι τιμές μιας συγκεκριμένης στήλης του πίνακα συμπίπτουν μεταξύ δυο διαφορετικών οχημάτων. Για παράδειγμα η επιτάχυνση ενός απορριμματοφόρου μπορεί να συμπίπτει με την επιτάχυνση επιθετικής οδήγησης ενός ΙΧ.

Η ανάλυση θορύβου αποτελεί μια σημαντική ανάλυση η οποία θα συνεισφέρει στην βελτίωση των δεδομένων που εξάγουμε.

Όλα τα test που κάνουμε είναι σε κατανομή 80% για training data set, 10% για validation set και 10% για το test set. Τα αποτελέσματα παρατίθενται στους Πίνακες 3, 4 :

Πίνακα 3. Ακρίβεια μετά απο επεξεργασία

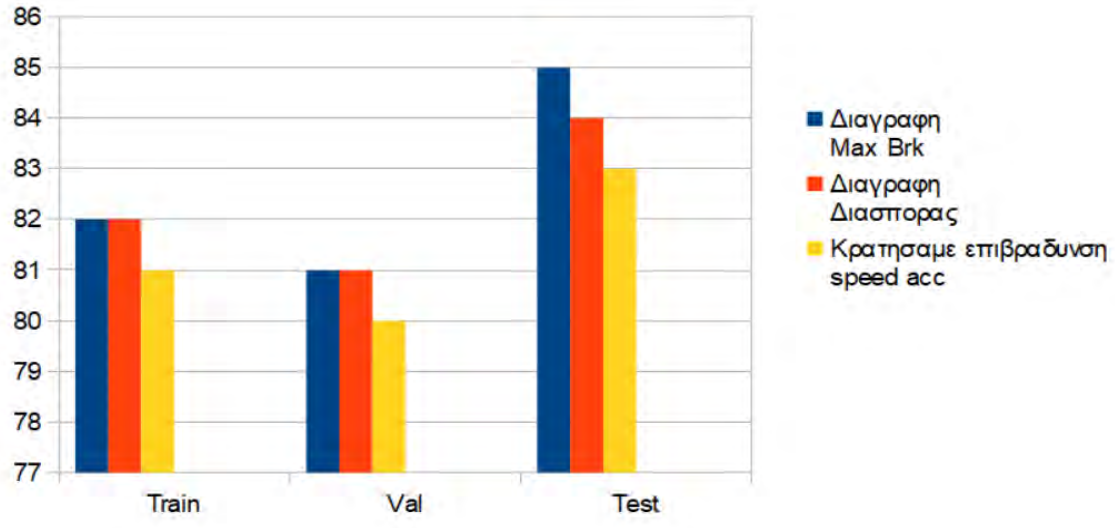
	Χωρις Διαγραφη	Διαγραφη Mean	Διαγραφη Speed	Διαγραφη Acc	Διαγραφη Επιβραδυνση	Διαγραφη Max Acc
Train	82	82	82	82	82	82
Val	82	80	78	80	80	81
Test	84	84	82	84	79	83



Πίνακας 4. Ακρίβεια μετά απο επεξεργασία

	Διαγραφη Max Brk	Διαγραφη Διασπορα	Κρατησαμε επιβραδυνση speed acc			
Train	82	82	81			

Val	81	81	80			
Test	85	84	83			



Συνεπώς δεν υπάρχει κάποια στήλη η οποία να δημιουργεί θόρυβο και σε περίπτωση που αφαιρεθεί να οδηγεί σε αύξηση της ακρίβειας.

6.3 Ανάλυση Συσχέτισης

Στο παρόν κεφάλαιο θα αναλυθεί η έννοια της σχέσης (associations) και της συσχέτισης (correlation) και θα εφαρμοστούν στατιστικές αναλύσεις στο μοντέλο μας. Σκοπός είναι να εξαχθούν αποτελέσματα σχετικά με την σχέση των δεδομένων που χρησιμοποιούνται για την κατασκευή του νευρωνικού και των αποτελεσμάτων που παράγει το νευρωνικό.

6.3.1 Η έννοια της Σχέσης και της Συσχέτισης

Στην ουσία η σχέση και η συσχέτιση μεταξύ των δεδομένων και αποτελεσμάτων δεν είναι τίποτα παραπάνω από στατιστικές έρευνες που εφαρμόζονται ούτως ώστε να διαπιστωθεί εάν πραγματικά υπάρχει σύνδεση και κατά πόσο ισχυρή είναι η σύνδεση αυτή. Σε αυτό το σημείο αξίζει να σημειωθεί το εξής : Εύκολα κάποιος θα μπορούσε να διαπιστώσει την συσχέτιση απλά παρατηρώντας τα δεδομένα και τα αποτελέσματα. Αυτό γίνεται σε περίπτωση απλών παραδειγμάτων όπως λόγου χάρη την συσχέτιση μεταξύ ύψους και

κιλών. Στην δική μας περίπτωση όμως τα δεδομένα δεν είναι τόσο απλά. Έχουμε να επεξεργαστούμε χιλιάδες γραμμές δεδομένων, τα οποία αξίζει να τονίσουμε ότι δεν έχουν περάσει από κάποια αρχική επεξεργασία , και το νευρωνικό δίκτυο σε αρκετές περιπτώσεις είναι αρκετά ευαίσθητο με αποτελέσματα να παράγει διαφορετικά αποτελέσματα με διαφορετικές εισόδους. Συνεπώς η ανάλυση που χρειάζεται να γίνει πρέπει να είναι βαθύτερη και πιο λεπτομερής.

Ξεκαθαρίζοντας λοιπόν τις έννοιες σχέσης-συσχέτισης, γίνεται εμφανές ότι χρειάζονται ορισμένα στατιστικά εργαλεία για να εξαχθούν τα απαιτούμενα συμπεράσματα. Δύο μέθοδοι για εξαγωγή τέτοιων συμπερασμάτων είναι η μέθοδος Pearson και η μέθοδος Spearman.

6.3.2 Συντελεστής Συσχέτισης Spearman

Ο συντελεστής συσχέτισης Spearman είναι ένα μη-παραμετρικό μέτρο της στατιστικής εξάρτησης μεταξύ δύο μεταβλητών. Αξιολογεί το πόσο καλά μπορεί να περιγράφει η σχέση μεταξύ των δύο μεταβλητών χρησιμοποιώντας μια μονότονη συνάρτηση. Εάν δεν υπάρχουν επαναλαμβανόμενες τιμές των δεδομένων, μια τέλεια συσχέτιση Spearman κατά +1 ή -1 συμβαίνει όταν κάθε μία από τις μεταβλητές είναι μια τέλεια μονότονη συνάρτηση της άλλης. Ο συντελεστής Spearman, όπως κάθε συντελεστής συσχέτισης, είναι κατάλληλος και για συνεχείς και για διακριτές μεταβλητές, συμπεριλαμβανομένων των τακτικών διακριτών μεταβλητών.[9]

6.3.3 Συντελεστής Συνδιακύμανσης Pearson

Ένα από τα πιο γνωστά μέτρα της εξάρτησης μεταξύ δύο ποσοτήτων είναι ο συντελεστής ελέγχου συνδιακύμανσης Perason, ή "Pearson συντελεστής συσχέτισης", που συνήθως ονομάζεται απλά "ο συντελεστής συσχέτισης". Είναι το πηλίκο της διαίρεσης της συνδιακύμανσης των δύο μεταβλητών με το γινόμενο των τυπικών αποκλίσεων.[10]

6.3.4 Εφαρμογή των Μεθόδων Συσχέτισης

Όπως είχε αναφερθεί σε προηγούμενες ενότητες, ένα data set δεδομένων χρησιμοποιείται για το train,το validation καθώς και το test του νευρωνικού μας δικτύου. Το train data set είναι το σύνολο των δεδομένων που επηρεάζουν άμεσα το test που θα γίνει για να ελεγχθεί η ακρίβεια του δικτύου. Το πρώτο ερώτημα που δημιουργείται είναι λοιπόν σε τι βαθμό επηρεάζεται το test του δικτύου σε σχέση με το data set που χρησιμοποιήθηκε για την δημιουργία του; Αυτό θα είναι συνεπώς το πρώτο ερώτημα που θα γίνει προσπάθεια για να απαντήσουμε.

Αρχικά θα εξαχθεί ένα απλό γράφημα για να διαπιστωθεί εάν όντως επηρεάζεται το test με βάση το πλήθος των training data που έχουμε ξεκινώντας από 500 γραμμές δεδομένων εκπαίδευσης και φτάνοντας μέχρι 8000. Τα δεδομένα που χρησιμοποιούνται για το validation και το test είναι χωρισμένα ισόποσα. Θα κάνουμε ένα γράφημα για raw δεδομένα και ένα για τα χαρακτηριστικά. Εάν εντοπιστεί σχέση ανάμεσα στα δυο set θα προχωρήσουμε την περαιτέρω έρευνα για την συσχέτιση

Raw

Πίνακας 5. Αποτελέσματα πειραμάτων στα Raw δεδομένα

Γραμμές Train	500	1000	2000	4000	6000	8000
Train	87	91	90	84	83	82
Val	71	75	72	81	78	82
Test	76	78	77	80	83	84

Συμπεραίνουμε ότι αυξάνοντας το πλήθος των δεδομένων εκπαίδευσης αυξάνεται και η ακρίβεια του δικτύου.

Χαρακτηριστικά

Πίνακας 6. Αποτελέσματα πειραμάτων στα Χαρακτηριστικά

Γραμμές Train	500	1000	2000	4000	6000	8000
Train	88	92	90	83	83	82
Val	69	73	71	80	78	82
Test	73	78	77	78	82	84

Αντίστοιχα με τα raw δεδομένα ,συμπεραίνουμε ότι αυξάνοντας το πλήθος των δεδομένων εκπαίδευσης αυξάνεται και η ακρίβεια του δικτύου.

Έχοντας λοιπόν μια πρώτη διαπίστωση σχετικά με την σχέση training set και test set , συνεχίζουμε βρίσκοντας την ακριβή συσχέτιση μεταξύ των δεδομένων.

Ανάλυση Συσχέτισης Train Data -Test Data

Όπως έχει προαναφερθεί, οι μέθοδοι που θα χρησιμοποιηθούν για την εύρεση της συσχέτισης θα είναι η Spearman και η Pearson. Σκοπός είναι ο υπολογισμός της ακριβούς συσχέτισης μεταξύ του πλήθους των δεδομένων που εκπαιδεύεται το νευρωνικό και της ακρίβειας του νευρωνικού όταν δοκιμάζεται σε δεδομένα άγνωστα σε αυτό (test set).

Spearman Train Data, Test Data σε Raw δεδομένα

1. Αρχικά δημιουργείται πίνακας με τις βαθμολογίες, τις τάξεις τους, τις διαφορές στις τάξεις (di) και τις διαφορές των τάξεων στο τετράγωνο (di^2).

Πίνακας 7. Εφαρμογή μεθόδου Spearman

Βαθμολογία 1	Βαθμολογία 2	Τάξη 1	Τάξη 2	di	Di^2
500	76	1	1	0	0
1000	78	2	3	1	1
2000	77	3	2	1	1
4000	80	4	4	0	0
6000	83	5	5	0	0
8000	84	6	6	0	0

2. Στην συνέχεια γίνεται πρόσθεση των di^2

$$\sum d_i^2 = 0 + 1 + 1 + 0 + 0 + 0$$
$$= 2$$

3. Υπολογίζεται ο συντελεστής του Spearman με τον εξής τύπο:

$$\rho = 1 - \frac{6 \text{ x } 2}{6[(6)2 - 1]}$$

$$\rho = 1 - \frac{12}{210}$$

p = 1 – 0.057

p = 0.943 .

Συνεπώς ο συντελεστής συσχέτισης Spearman είναι 0.943 και συμπεραίνεται ότι η συσχέτιση μεταξύ του πλήθους των δεδομένων που δέχεται το δίκτυο για εκπαίδευση σε σχέση με την ακρίβεια του δικτύου είναι πολύ ισχυρή. Το ίδιο συμπέρασμα μπορεί να από προκύψει και από την μέθοδο του Pearson. Παρακάτω αναλυέται η διαδικασία για τον υπολογισμό συνδιακύμανσης του Pearson.

Πίνακας 8. Εφαρμογή μεθόδου Spearman

Γραμμές Train X	Test Y	X*Y	X^2	Y^2
500	73	36500	250000	5329
1000	78	78000	1000000	6084
2000	77	154000	4000000	5929
4000	78	312000	16000000	6084
6000	82	492000	36000000	6724
8000	84	672000	64000000	7056

Για τον υπολογισμό της συνδιακύμανσης Pearson χρησιμοποιούμε τον εξής τύπο:

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[\, n\Sigma x^2 - (\Sigma x)^2 \,][\, n\Sigma y^2 - (\Sigma y)^2 \,]}}$$

Για την ολοκλήρωση της διαδικασίας υπολογίζουμε:

- Σx*y = 1744500
- Σx = 21500
- Σy = 472
- Σx^2 = 121250000
- Σy^2 = 37206
- n = 6 (ο αριθμός των δειγμάτων)

r = 0.94

Βιβλιογραφία

1. Λιμνιάτη, Λ. (n.d.). Αυτόνομα οχήματα Uber. Νέα, τεχνολογία, εξελίξεις. Retrieved October 3, 2018, from https://www.autonomous.gr/tag/uber/

2. Autopilot. (n.d.). Retrieved October 3, 2018, from https://www.tesla.com/autopilot

3. A new way forward for mobility – Waymo. (n.d.). Retrieved October 3, 2018, from https://www.google.com/selfdrivingcar/

4.Meiring, G. A., & Myburgh, H. C. (2015, December). Retrieved October 3, 2018, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4721742/

5. https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf

6. https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf

7. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

8. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

9. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

10. G. (n.d.). Ανάλυση: Τι είναι η Τεχνητή Νοημοσύνη και πως θα αλλάξει τη ζωή μας;. Retrieved October 3, 2018, from http://todiktio.eu/index.php/activity/papers-publications/item/810-analysi-ti-einai-i-texniti-noimosyni-kai-pos-tha-allaksei-ti-zoi-mas

11. TensorFlow. (n.d.). Retrieved October 3, 2018, from https://www.tensorflow.org/

12. Apache MXNet on AWS. (n.d.). Retrieved October 3, 2018, from https://aws.amazon.com/mxnet/

13. Why Use Keras? (n.d.). Retrieved October 3, 2018, from https://cran.r-project.org/web/packages/keras/vignettes/why_use_keras.html

14. Why One-Hot Encode Data in Machine Learning? (2018, May 19). Retrieved October 3, 2018, from https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/

15. Tf.estimator.Estimator | TensorFlow. (n.d.). Retrieved October 3, 2018, from https://www.tensorflow.org/api_docs/python/tf/estimator/Estimator

16. Retrieved October 3, 2018, from https://www.quora.com/Is-a-fully-connected-neural-network-equal-to-a-feed-forward-neural-network/answer/Chomba-Bupe#

17. Everything About OBD2 Scanners, How Does It Work & How To Use It. (n.d.). Retrieved October 3, 2018, from https://curateview.com/everything-about-obd2-scanners/

Αναφορά Εικόνων

1. Blondel, P. (2018, April 03). Which machine learning algorithm to choose for my problem ? Retrieved October 3, 2018, from https://recast.ai/blog/machine-learning-algorithms/

2. Zhao, Q. (n.d.). AI, Machine Learning, and Deep Learning Explained. Retrieved October 3, 2018, from http://blog.operasolutions.com/ai-machine-learning-and-deep-learning-defined

3. https://www.quora.com/What-is-the-difference-between-deep-learning-and-usual-machine-learning

4. (2017, December 21). Deep Learning made easy with Deep Cognition – Becoming Human: Artificial Intelligence Magazine. Retrieved October 3, 2018, from https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351

5.https://www.slideshare.net/activeeon/machinelearningfordummiesandrewssobralactiveeon-81373804

6 Why Use Keras? (n.d.). Retrieved October 3, 2018, from https://cran.r-project.org/web/packages/keras/vignettes/why_use_keras.html

7. What's the difference between a Tensorflow Keras Model and Estimator? (n.d.). Retrieved October 3, 2018, from https://stackoverflow.com/questions/51455863/whats-the-difference-between-a-tensorflow-keras-model-and-estimator

8. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

9. Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

10. Sign In. (n.d.). Retrieved October 3, 2018, from http://rpubs.com/josevilardy/crossvalidation

11. Donges, N. (2018, February 25). Recurrent Neural Networks and LSTM – Towards Data Science. Retrieved October 3, 2018, from https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5

ΠΑΡΑΡΤΗΜΑ

Όλοι οι κώδιες που αναπτύχθηκαν είναι ανεβασμένοι στο github στο παρακάτω link :

https://github.com/DValouxis/Codes_Diploma