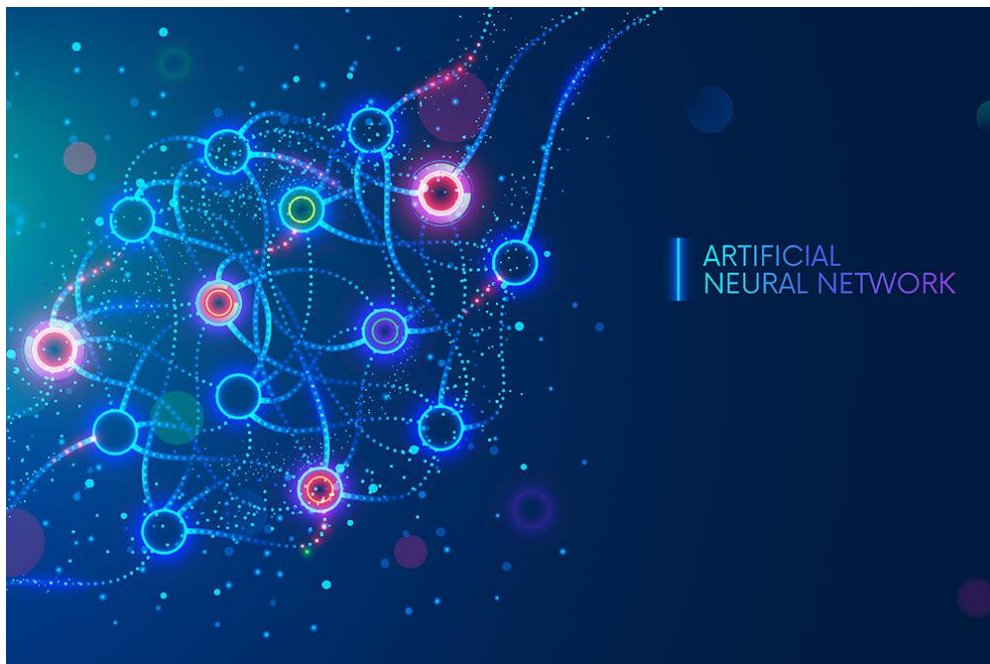


ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ  
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ  
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

# ΕΦΑΡΜΟΓΕΣ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ ΣΤΗ ΜΗΧΑΝΟΛΟΓΙΑ: ΜΙΑ ΜΕΛΕΤΗ ΠΕΡΙΠΤΩΣΗΣ ΤΕΧΝΗΤΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ



ΦΟΙΤΗΤΕΣ: ΓΙΔΑΚΟΣ ΓΕΩΡΓΙΟΣ (Α.Μ. 7124)  
ΠΑΠΑΔΟΠΟΥΛΟΣ ΠΟΛΥΔΩΡΟΣ (Α.Μ. 7199)

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΘΕΟΦΑΝΗΣ ΑΡΑΒΑΝΗΣ  
ΑΚΑΔΗΜΑΪΚΟΣ ΥΠΟΤΡΟΦΟΣ

ΠΑΤΡΑ 2021

# ΠΡΟΛΟΓΟΣ

Η παρούσα Πτυχιακή Εργασία εκπονήθηκε κατά την θερινή περίοδο του Ακαδημαϊκού Έτους 2020 - 2021, στα πλαίσια του Προπτυχιακού Προγράμματος σπουδών της Σχολής Τεχνολογικών Εφαρμογών του Τμήματος Μηχανολόγων Μηχανικών του Πανεπιστημίου Πελοποννήσου. Η εργασία πραγματοποιήθηκε υπό την επίβλεψη του κ. Θεοφάνη Αραβανή, Ακαδημαϊκό Υπότροφο του Τμήματος Μηχανολόγων Μηχανικών.

Η παρούσα Πτυχιακή Εργασία είναι μία μελέτη περίπτωσης Τεχνητής Νοημοσύνης πιο συγκεκριμένα Τεχνητών Νευρωνικών Δικτύων. Αντικείμενό της αποτελεί η αναφορά μερικών βασικών γνώσεων όσον αφορά την Τεχνητή Νοημοσύνη και κάποιων σημαντικών ιστορικές στιγμών κατά τη διάρκεια της εξέλιξής της. Θα αναφερθούν επίσης χρήσιμες εφαρμογές τους στον ευρύ τομέα της Μηχανικής. Έπειτα θα ακολουθήσει το κατασκευαστικό (πειραματικό) μέρος της εργασίας αυτής που είναι η υλοποίηση δύο Τεχνητών Νευρωνικών Δικτύων (ΤΝΔ). Ένα ΤΝΔ απλού (συμβατικού) τύπου και ένα συνελκτικό ή αλλιώς Convolutional Neural Network (CNN). Σκοπός μας είναι η εξειδίκευσή τους στην αναγνώριση προτύπων/μοτίβων και στην ταξινόμηση διαφόρων Μηχανολογικών εξαρτημάτων (όπως κοχλιών), με αφορμή την ένταξή της Τεχνητής Νοημοσύνης στην Μηχανολογία.

Στο σημείο αυτό θα θέλαμε να ευχαριστήσουμε τον επιβλέποντα καθηγητή μας ακαδημαϊκό υπότροφο κ. Θεοφάνη Αραβανή, για την αμέριστη συμπαράστασή, προσοχή του και καθοδήγησή του κατά την εκπόνηση της εργασίας αυτής, καθώς και τον πολύ καλό μας φίλο Γεώργιο Παναγόπουλο, πτυχιούχο της σχολής Εφαρμοσμένων Μαθηματικών και Φυσικών Επιστημών του Μετσόβιου Πανεπιστημίου, για τις συμβουλές που μας παρείχε σχετικά με την συγγραφή της εργασίας αυτής. Φυσικά δεν θα μπορούσαμε να μην πούμε ένα μεγάλο ευχαριστώ στους γονείς μας, διότι χάριν σε αυτούς φτάσαμε στο σημείο αυτό.

Γιδάκος Γεώργιος  
Παπαδόπουλος Πολύδωρος  
Δεκέμβριος 2021

**Υπεύθυνη Δήλωση Φοιτητών:** Οι κάτωθι υπογεγραμμένοι Φοιτητές έχουμε επίγνωση των συνεπειών του Νόμου περί λογοκλοπής και δηλώνουμε υπεύθυνα ότι είμαστε συγγραφείς αυτής της Διπλωματικής Εργασίας, αναλαμβάνοντας την ευθύνη επί ολοκλήρου του κειμένου εξ ίσου, έχουμε δε αναφέρει στην Βιβλιογραφία μας όλες τις πηγές τις οποίες χρησιμοποιήσαμε και λάβαμε ιδέες ή δεδομένα. Δηλώνουμε επίσης ότι, οποιοδήποτε στοιχείο ή κείμενο το οποίο έχουμε ενσωματώσει στην εργασία μας προερχόμενο από Βιβλία ή άλλες εργασίες ή το διαδίκτυο, γραμμένο ακριβώς ή παραφρασμένο, το έχουμε πλήρως αναγνωρίσει ως πνευματικό έργο άλλου συγγραφέα και έχουμε αναφέρει ανελλιπώς το όνομά του και την πηγή προέλευσης.

Οι Φοιτητές  
(Ονοματεπώνυμο) (Ονοματεπώνυμο)

ΠΑΠΑΔΟΠΟΥΛΟΣ ΠΟΛΥΔΩΡΟΣ ΓΙΔΑΚΟΣ ΓΕΩΡΓΙΟΣ

## ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία με τίτλο «Εφαρμογές Τεχνητής Νοημοσύνης στη Μηχανολογία: Μια Μελέτη Περίπτωσης Τεχνητών Νευρωνικών Δικτύων» έχει ως αντικείμενο, καταρχήν την παρουσίαση κάποιων βασικών γνώσεων της τεχνητής νοημοσύνης και σημαντικών ιστορικών γεγονότων. Επιπροσθέτως θα γίνει αναφορά στις διάφορες εφαρμογές της Τεχνητής Νοημοσύνης στον τομέα της Μηχανικής, ειδικότερα της Μηχανολογίας, ενώ επίσης θα υλοποιηθούν δύο Τεχνητά Νευρωνικά Δίκτυα ικανά να αναγνωρίζουν και να ταξινομούν κατάλληλα κάποια μηχανολογικά στοιχεία (π.χ. κοχλίες).

Το πρώτο κεφάλαιο αφορά την φυσική και την τεχνητή νοημοσύνη όπου δίνονται οι ορισμοί τους. Στην συνέχεια γίνεται μια προσπάθεια διαχωρισμού της τεχνητής και φυσικής νοημοσύνης, ενώ βεβαίως εμπεριέχεται και η απαραίτητη ιστορική αναδρομή ώστε να μπορεί ο αναγνώστης να αντιληφθεί την εξέλιξή της με την πάροδο του χρόνου έως και το σήμερα. Δεν παραλείπεται φυσικά η αναφορά ενός μέρους του θεωρητικού υπόβαθρου της πτυχιακής εργασίας.

Στο δεύτερο κεφάλαιο αρχικά θα αναφερθούμε στις εφαρμογές της τεχνητής νοημοσύνης στον τομέα της μηχανικής αλλά και πιο ειδικά στον τομέα της μηχανολογίας. Θα αναφέρουμε κάποιες από τις πιο σημαντικές εφαρμογές στους τομείς αυτούς έτσι ώστε να μπορέσει κανείς να αντιληφθεί το πόσο πολύ έχει διεισδύσει η τεχνητή νοημοσύνη στους αντίστοιχους επιστημονικούς τομείς, καταφέροντας έτσι, πράγματα που δεν θα μπορούσαμε αλλιώς να πετύχουμε.

Στο τρίτο κεφάλαιο θα αναφερθούμε στα τεχνητά νευρωνικά δίκτυα και σε πολύ θεμελιώδεις έννοιες που αφορούν αυτά. Θα πραγματοποιηθεί ανάλυση σχετικά με τη σημασία τους, ανάλυση της αρχιτεκτονικής τους, ο τρόπος κατασκευής τους και το πως αυτά λειτουργούν. Ακολούθως θα αναφερθούν οι τεχνικές και μέθοδοι εκπαίδευσης τους καθώς και ο τρόπος αξιολόγησής τους.

Το τέταρτο κεφάλαιο αφορά την υλοποίηση δύο Τεχνητών Νευρωνικών Δικτύων με χρήση της γλώσσας προγραμματισμού Python, τα οποία θα κατατάσσουν σε τέσσερεις κλάσεις διάφορα μηχανολογικά στοιχεία. Το πρώτο Τεχνητό Νευρωνικό Δίκτυο θα είναι ένα απλό Τεχνητό Νευρωνικό Δίκτυο ή αλλιώς Artificial Neural Network (ANN), ενώ το δεύτερο θα είναι ένα συνελκτικό νευρωνικό δίκτυο ή αλλιώς CNN, ένα δίκτυο διαφορετικού τύπου. Προφανές είναι το γεγονός ότι αποσκοπούμε στην σύγκριση των δυο αυτών δικτύων ως προς τα αποτελέσματά τους.

Το πέμπτο και τελευταίο κεφάλαιο αποτελεί μια επισκόπηση των αποτελεσμάτων της πτυχιακής εργασίας, ενώ θα γίνει αναφορά και στην μελλοντική εξέλιξή της τεχνητής νοημοσύνης, δίνοντας έμφαση στις επιτεύξεις της τεχνητής νοημοσύνης σε μια πληθώρα

επιστημών. Ολοκληρώνοντας, θα σχολιάσουμε το τι ακόμα μέλει να συμβεί και το πόσο μακριά μπορεί να φτάσει.

Η Τεχνητή Νοημοσύνη είναι μία επιστήμη με πολλά υποσχόμενα, ως εκ τούτου η ακρίβεια των αποτελεσμάτων των Τεχνητών Νευρωνικών Δικτύων που θα κατασκευάσουμε αναμένεται να είναι μεγάλη. Η εργασία αυτή καταλήγει σε μία σύγκριση των αποτελεσμάτων ενός απλού Νευρωνικού Δικτύου ( ANN ) με ένα πιο εξειδικευμένο ανταγωνιστή του, ένα συνελικτικό Νευρωνικό Δίκτυο ( CNN ). Θα δούμε ότι το CNN είναι από την μία πιο αξιόπιστο αλλά από την άλλη πιο κοστοβόρο σε φυσικούς πόρους και υπολογιστική ισχύ από ότι το ANN.

## ABSTRACT

The present thesis entitled "Application of Artificial Intelligence in Mechanical Engineering: A Case Study of Artificial Neural Networks" has as its object, first of all, the disclosure of some basic knowledge of artificial intelligence, its history, its various applications in the field of engineering, especially in Mechanical Engineering, and our experimentation with it, by constructing two artificial neural networks capable of recognizing and appropriately analyzing some mechanical elements (e.g. a bolt).

The first chapter deals with natural and artificial intelligence, where firstly their definitions are given. Then an attempt to distinguish artificial and natural intelligence is made, and of course the necessary historical background is included so that the reader can understand its evolution over time, up to the present day.

In the second chapter, we will first refer to the applications of artificial intelligence in the field of engineering and more specifically in the field of mechanical engineering. We will mention some of the most important applications in these fields so that one can understand how much AI has been implemented in the respective scientific fields, thus achieving things that we could not otherwise achieve.

In the third chapter we will discuss artificial neural networks and very fundamental concepts related to them. An analysis on their importance, analysis of their architecture, how they are built and how they work will be carried out. Subsequently, their training techniques and methods and how to evaluate them will be discussed.

The fourth chapter deals with the construction of two artificial neural networks AI using the Python programming language, which will classify various engineering components into four classes. The first neural network will be a simple artificial neural network or ANN while the second will be a convolutional neural network or CNN, a network of a different type. It is obvious that we aim to compare these two networks in terms of their results.

The fifth and final chapter is an overview of the thesis as well as the evolution of artificial intelligence from today to the future, emphasizing the achievements of artificial intelligence in a multitude of disciplines and concluding with what is still to come and how far it can go.

AI is a promising science, hence the accuracy of the results of the artificial neural networks we will construct is expected to be high. This paper concludes with a comparison of the results of a simple neural network (ANN) with a more specialized competitor, a convolutional neural network (CNN). We will see that the CNN is on the one hand more reliable but on the other hand more costly in terms of physical resources and computational power than the ANN.

# Π Ε Ρ Ι Ε Χ Ο Μ Ε Ν Α

ΠΡΟΛΟΓΟΣ... I	
ΠΕΡΙΛΗΨΗ ... II	
ABSTRACT... IV	
ΠΕΡΙΕΧΟΜΕΝΑ... V	

## ΕΙΣΑΓΩΓΗ

### ΚΕΦΑΛΑΙΟ 1<sup>ο</sup>: ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

1.1 Ορισμός της νοημοσύνης .....	2
1.2 Ορισμός της τεχνητής νοημοσύνης.....	2
1.3 Διαφορά της Τεχνητής από τη Φυσική νοημοσύνη .....	3
1.4 Δοκιμασία Turing .....	3
1.5 Ιστορικά Γεγονότα .....	5
1.6 Μηχανική Μάθηση.....	8
1.7 Βαθιά Μάθηση (Deep Learning) .....	9
1.8 Σχεδιασμός Ενεργειών και Χρονοπρογραμματισμός.....	10
1.9 Ευφυείς πράκτορες (Intelligent Agents).....	10
1.10 Ευφυείς Υπηρεσίες Διαδικτύου(semantic web).....	11
1.11 Προσαρμοζόμενα και Εξελισσόμενα Ευφυή Συστήματα.....	12

### ΚΕΦΑΛΑΙΟ 2<sup>ο</sup>: ΕΦΑΡΜΟΓΕΣ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ ΣΤΟΥΣ ΤΟΜΕΙΣ ΤΗΣ ΜΗΧΑΝΙΚΗΣ

2.1 Έλεγχος της διαδικασίας τήξης του μετάλλου.....	13
2.2 Condition Monitoring - Fault Detectection.....	14
2.3 Τεχνητή Νοημοσύνη στην Αυτοκινητοβιομηχανία.....	15
2.4 Τεχνητή Νοημοσύνη στην Αεροδιαστημική.....	17

### ΚΕΦΑΛΑΙΟ 3<sup>ο</sup>: ΕΙΣΑΓΩΓΗ ΣΤΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΤΑ ΤΕΧΝΗΤΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ

3.1 Βιολογική Έμπνευση .....	20
3.2 Η έννοια του τεχνητού νευρώνα .....	21
3.3 Συνάρτηση ενεργοποίησης.....	22
3.4 Perceptron .....	25
3.5 Τοπολογία Τεχνητών Νευρωνικών Δικτύων .....	25
3.6 Τεχνικές Εκπαίδευσης ANN.....	27
3.6.1 Επιβλεπόμενη Μάθηση (Supervised Learning) .....	28
3.6.2 Μέθοδος κ-πλησιέστερων γειτόνων (k-NN) .....	29
3.6.3 Μη Επιβλεπόμενη Μάθηση (Unsupervised Learning).....	29
3.6.4 Ενισχυτική Μάθηση (Reinforced Learning) .....	30

3.7 Μέτρα Αξιολόγησης .....	30
3.7.1 Loss .....	30
3.7.2 Accuracy .....	31
3.8 Κύριες Προκλήσεις .....	31
3.8.1 Overfitting/Underfitting .....	31
3.8.2 Data Mismatch .....	32
3.9 Μέθοδος Απώτομης Καθόδου (Gradient Decent) .....	33
3.10 Αλγόριθμος Οπισθοδιάδοσης (Backpropagation) .....	33
3.11 Συνελικτικά Νευρωνικά Δίκτυα .....	34

## **ΚΕΦΑΛΑΙΟ 4<sup>ο</sup>: ΧΡΗΣΗ ΤΕΧΝΗΤΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΓΙΑ ΑΝΑΓΝΩΡΙΣΗ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΕΞΑΡΤΗΜΑΤΩΝ**

4.1 Σημαντικές Βιβλιοθήκες .....	36
4.1.1 NumPy .....	36
4.1.2 TensorFlow .....	37
4.1.3 Keras API .....	38
4.2 Στρώματα ΤΝΔ (Layers) .....	38
4.2.1 Dense Layer (Πυκνό Στρώμα) .....	38
4.2.2 Convolutional Layers (Συνελικτικό Στρώμα) .....	39
4.2.3 Pooling Layers (Στρώματα Συγκέντρωσης) .....	41
4.3 Κατασκευή Βάσης δεδομένων (Dataset) .....	42
4.4 Υλοποίηση Απλού Τεχνητού Νευρωνικού Δικτύου (ANN) .....	44
4.4.1 Διεξαγωγή και Αποτύπωση Αποτελεσμάτων ANN .....	47
4.5 Κατασκευή Συνελικτικού Νευρωνικού Δικτύου .....	49
4.5.1 Διεξαγωγή και Αποτύπωση Αποτελεσμάτων CNN .....	51
4.6 Σύγκριση των δύο Δικτύων .....	53
4.6.1 Σύγκριση ως προς τον τρόπο λειτουργίας .....	53
4.6.2 Σύγκριση των εισόδων τους .....	54
4.6.3 Σύγκριση στην ταξινόμηση εικόνων .....	54

## **ΚΕΦΑΛΑΙΟ 5<sup>ο</sup>: ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΕΡΑΙΤΕΡΩ ΜΕΛΕΤΗ**

5.1 Συμπεράσματα και Περαιτέρω Μελέτη .....	56
---	----

<b>ΠΑΡΑΡΤΗΜΑ Ι</b> .....	57
--------------------------	----

<b>ΠΑΡΑΡΤΗΜΑ ΙΙ</b> .....	60
---------------------------	----

<b>ΒΙΒΛΙΟΓΡΑΦΙΑ</b> .....	63
---------------------------	----

## ΣΥΜΒΟΛΙΣΜΟΙ

$w_i$  = βάρη Νευρωνικού Δικτύου  
 $x_i$  = μεταβλητές εισόδου Νευρωνικού Δικτύου  
 $y$  = μεταβλητές εξόδου Νευρωνικού Δικτύου  
 $\Phi(x)$  = έξοδος συναρτήσεων ενεργοποίησης

## ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

ReLU = Rectified Linear Unit  
Conv2d = Στρώμα συνέλιξης στο Συνελικτικό Νευρωνικό Δίκτυο  
RGB= Red, Green, Blue (χρώματα που απαρτίζουν μία εικόνα)  
CPU= Επεξεργαστής Η/Υ  
GPU= Κάρτα Γραφικών Η/Υ  
API = Application Programming Interface  
ANN = Artificial Neural Network  
CNN = Convolutional Neural Network  
NumPy = Numerical Python  
nd-array = πίνακας n- διαστάσεων  
TPU's = μονάδα επεξεργασίας τανιστή  
TN = Τεχνητή Νοημοσύνη  
NN = Neural Net  
CM = Confusion Matrix  
NDT = Non- Destructive Tests  
HPC = High Performance Computing  
VRAM = Μνήμη Κάρτας Γραφικών





## ΕΙΣΑΓΩΓΗ

Σε αυτή την εποχή της νοημοσύνης, οι άνθρωποι περιβάλλονται από σύγχρονες προηγμένες τεχνολογίες. Με μια συσκευή αρκετά μικρή ώστε να χωρέσει στο χωρέσει στην παλάμη μας, οι εφαρμογές τεχνητής νοημοσύνης έχουν δώσει τη δυνατότητα πρόσβασης σε όλες τις πληροφορίες γύρω από τον κόσμο. Το λογισμικό τεχνητής νοημοσύνης κάνει την ανθρώπινη ζωή ευκολότερη με ποικίλους τρόπους. Επίσης, οι αλγόριθμοι αυτοεκμάθησης και η διαθεσιμότητα διαδικτυακών δεδομένων με υπολογισμούς χαμηλού κόστους έχουν οδηγήσει τη μηχανική μάθηση στο επόμενο επίπεδο. Η δημοφιλία της τεχνητής νοημοσύνης έχει αυξηθεί γρήγορα και έχει γίνει μέρος της ανθρώπινης καθημερινότητας. Η γρήγορη ανάπτυξη της σύγχρονης ευφυούς τεχνολογίας έχει δώσει ελπίδες για ένα καλύτερο ανθρώπινο μέλλον

Η θέληση για τη δημιουργία “έξυπνων” μηχανών είχε ξεκινήσει πολύ νωρίτερα, τις τελευταίες δεκαετίες η τεχνητή νοημοσύνη ήταν ένα όνειρο για τους ερευνητές παγκοσμίως. Οι επιτυχίες της τεχνητής νοημοσύνης έχουν καταδειχθεί, από την αυξανόμενη υπολογιστική ισχύ και την ικανότητα συγκέντρωσης και αποθήκευσης μεγάλου αριθμού δεδομένων. Η τεχνητή νοημοσύνη είναι μια διεπιστημονική επιστήμη γνωστή και ως μηχανική νοημοσύνη που μιμείται τις ανθρώπινες διανοητικές συμπεριφορές και ικανότητες.

Αντικείμενο της πτυχιακής εργασίας είναι η παρουσίαση κάποιων βασικών εννοιών της τεχνητής νοημοσύνης και ακολούθως η αναφορά σε μερικές από τις πιο ιστορικά αξίες στιγμές της. Δεν παραλείπονται οι αναφορές διαφόρων εφαρμογών της στον τομέα της Μηχανικής, ειδικότερα της Μηχανολογίας. Η εργασία καταλήγει στον πειραματισμό μας με αυτήν κατασκευάζοντας δύο τεχνητά νευρωνικά δίκτυα ικανά να αναγνωρίζουν και να ταξινομούν κατάλληλα κάποια μηχανολογικά στοιχεία. Τα νευρωνικά δίκτυα έχουν έναν ολοένα και πιο σημαντικό ρόλο στους τομείς της μηχανικής (όπως τη βιομηχανία), προσφέροντας μια λύση στις αυξανόμενες «παραγωγές κομματιών» λόγω απαιτήσεων του καταναλωτικού κοινού, καθώς και αποτελούν μια σταθερή και αποδοτική μέθοδο για διασφάλιση μέγιστης ποιότητας.

# 1. ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Για να καταλάβει κανείς καλύτερα την τεχνητή νοημοσύνη ως προς το τι είναι αλλά και πως λειτουργεί, θα πρέπει πρώτα να γνωρίσει μερικές εισαγωγικές έννοιες για αυτήν, αλλά και από που πάρθηκε η έμπνευση για την δημιουργία της. Παρακάτω στο κεφάλαιο αυτό θα αναφερθούμε στις πρώτες, εισαγωγικές έννοιες όσον αφορά την τεχνητή νοημοσύνη, ξεκινώντας από το μηδέν.

## 1.1 ΟΡΙΣΜΟΣ ΦΥΣΙΚΗΣ ΝΟΗΜΟΣΥΝΗΣ

Η νοημοσύνη είναι ένα σύνολο λειτουργιών που μπορούμε να χρησιμοποιήσουμε έτσι ώστε να αντιμετωπίσουμε νέες καταστάσεις και να λύσουμε προβλήματα αξιοποιώντας παλαιότερες εμπειρίες.

Η νοημοσύνη για να μπορεί να μελετηθεί καλύτερα χωρίστηκε σε επιμέρους κομμάτια που ονομάζονται Επίπεδα Νοημοσύνης ενώ η αντίστοιχη θεωρία που προτάθηκε από τον Χάουαρντ Γκάρντνερ (Howard Gardner) ονομάζεται Θεωρία Πολλαπλής Νοημοσύνης. Κάποια από αυτά τα επίπεδα είναι:

- Συναισθηματική Νοημοσύνη
- Λογικό-μαθηματική Νοημοσύνη
- Χωροταξική Νοημοσύνη
- Μουσική Νοημοσύνη
- Σωματοκινητική Νοημοσύνη

## 1.2 ΟΡΙΣΜΟΣ ΤΗΣ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ

Έχουν διατυπωθεί αρκετοί ορισμοί της Τεχνητής Νοημοσύνης. Οι ορισμοί αυτοί μπορούν να ταξινομηθούν σε τέσσερις κατηγορίες οι οποίες προσεγγίζουν την περιοχή από διαφορετική οπτική όσον αφορά το στόχο της Τεχνητής Νοημοσύνης.

- 1) Συστήματα σκεπτόμενα σαν τον άνθρωπο
- 2) Συστήματα με ορθολογική σκέψη
- 3) Συστήματα που δρουν όπως ο άνθρωπος
- 4) Συστήματα που ενεργούν ορθολογικά.

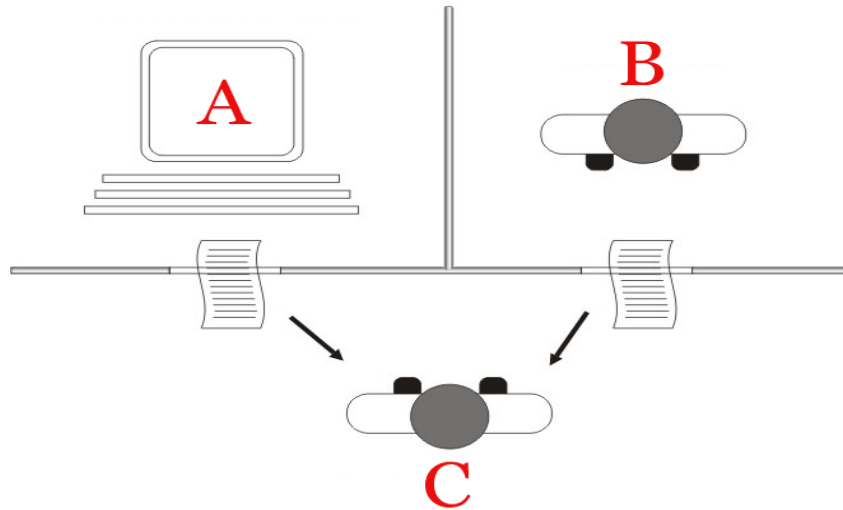
Απόρεια των παραπάνω κατηγοριών είναι ένας γενικευμένος όρος της ΤΝ. Η Τεχνητή Νοημοσύνη είναι τομέας της επιστήμης των υπολογιστών ο οποίος ασχολείται με τη αρχιτεκτονική και υλοποίηση προγραμμάτων που μπορούν να μιμηθούν τις ανθρώπινες γνωστικές ικανότητες, όπως η μάθηση, η επίλυση προβλημάτων, η κατανόηση της φυσικής γλώσσας κ.α., παρουσιάζοντας με αυτόν τον τρόπο «ανθρώπινη συμπεριφορά».

### **1.3 ΔΙΑΦΟΡΑ ΤΕΧΝΗΤΗΣ ΚΑΙ ΦΥΣΙΚΗΣ ΝΟΗΜΟΣΥΝΗΣ**

Ως άνθρωποι μπορούμε για οποιοδήποτε φαινόμενο να διακρίνουμε τόσο το πραγματικό από το ψεύτικο όσο και το φυσικό από το τεχνητό. Το Φυσικό σημαίνει ότι δημιουργείται από τη φύση ενώ τεχνητό ότι δημιουργείται από τους ανθρώπους. Έτσι και η τεχνητή νοημοσύνη μπορεί να ερμηνευτεί ως το αντίθετο της πραγματικής νοημοσύνης. Αν ένα πρόγραμμα συμπεριφέρεται έξυπνα είναι ευφυές επομένως, η τεχνητή νοημοσύνη θα είναι πραγματική νοημοσύνη η οποία θα έχει δημιουργηθεί με τεχνητό τρόπο. Καθορίζεται επιπλέον από την εξωτερική συμπεριφορά, αυτό ήταν το κίνητρο για τη δημιουργία μιας δοκιμασίας για τη νοημοσύνη που είναι γνωστή ως δοκιμασία Turing.

### **1.4 ΔΟΚΙΜΑΣΙΑ TURING**

Το τεστ Τούρινγκ, που αρχικά ονομάστηκε ως παιχνίδι μίμησης από τον Άλαν Τούρινγκ το 1950, είναι μια δοκιμή της ικανότητας μιας μηχανής να επιδεικνύει ευφυή συμπεριφορά ισοδύναμη ή αδιάκριτη από αυτή ενός ανθρώπου. Ο Turing πρότεινε ότι ένας ανθρώπινος αξιολογητής θα κρίνει τις συνομιλίες φυσικής γλώσσας μεταξύ ενός ανθρώπου και μιας μηχανής σχεδιασμένης να παράγει ανθρώπινες αποκρίσεις. Ο αξιολογητής θα γνωρίζει ότι ένας από τους δύο εταίρους στη συνομιλία είναι μια μηχανή και όλοι οι συμμετέχοντες θα είναι διαχωρισμένοι μεταξύ τους. Η συνομιλία θα περιοριζόταν σε ένα κανάλι μόνο κειμένου, όπως ένα πληκτρολόγιο και οθόνη υπολογιστή, ώστε το αποτέλεσμα να μην εξαρτάται από την ικανότητα του μηχανήματος να αποδίδει λέξεις ως ομιλία. Εάν ο αξιολογητής δεν μπορεί να ξεχωρίσει αξιόπιστα το μηχάνημα από τον άνθρωπο, λέγεται ότι το μηχάνημα έχει περάσει τη δοκιμή. Τα αποτελέσματα των δοκιμών δεν εξαρτώνται από την ικανότητα του μηχανήματος να δίνει σωστές απαντήσεις σε ερωτήσεις, μόνο από το πόσο πολύ μοιάζουν οι απαντήσεις του με αυτές που θα έδινε ένας άνθρωπος.



**Εικόνα 1.1:** Δοκιμασία Τούρινγκ

Ο Turing είχε προβλέψει πως μέχρι το 2000 θα είχε αναπτυχθεί τεχνητή νοημοσύνη που θα μπορούσε να ξεγελάσει το 30% των αξιολογητών, μέσα σε πέντε λεπτά συζήτησης. Το πρώτο πρόγραμμα που κατάφερε να περάσει το συγκεκριμένο τεστ θεωρείται πως είναι το ELIZA που έφτιαξε το 1976 ο Αμερικανός προγραμματιστής Τζόσεφ Βάιζενμπαουμ (Joseph Weizenbaum) και το οποίο κατάφερε να πείσει τη γραμματέα του πως συνομιλούσε με εκείνον. Ύστερα ακολούθησαν και άλλα προγράμματα τα οποία έδειξαν μεταξύ άλλων πως η δοκιμασία Turing παρόλο που μπορεί να αποδείξει ότι οι άνθρωποι μπορούν να ξεγελαστούν από μηχανές είναι ανίκανη να απαντήσει στα σύγχρονα ερωτήματα για την τεχνητή νοημοσύνη, για τις συνέπειές της αλλά, ούτε μπορεί να προβλέπει το πότε θα επιτευχθεί. Παρόλο που τα προγράμματα που προσπαθούν μέχρι σήμερα να περάσουν το συγκεκριμένο τεστ είναι αξιοθαύμαστα δεν επιδεικνύουν την ικανότητα της αλληλεπίδρασης με τον εξωτερικό κόσμο ή αλλιώς το περιβάλλον πράγμα που είναι ανθρώπινο χαρακτηριστικό, ενώ ένας προσεκτικός παρατηρητής με τις κατάλληλες ερωτήσεις θα καταφέρει να διαπιστώσει πως πρόκειται για μηχανές. Ανοικτό πάντως παραμένει το ερώτημα για το εάν και πότε θα καταφέρει η ανθρωπότητα να παραγάγει τεχνητή νοημοσύνη. Μία από τις σημαντικότερες προβλέψεις είναι αυτή του στελέχους της Google, Ρέι Κούρτςβαϊλ (Ray Kurzweil), η οποία αναφέρει ότι η ανακάλυψη της τεχνητής νοημοσύνης ίσως πραγματοποιηθεί έως το έτος 2029, βασίζεται στο νόμο του Μουρ για την πρόοδο των ηλεκτρονικών διατάξεων. Αντίθετα ο Φρεντ Μπρουκς (Fred Brooks), από τους πρωτεργάτες της IBM, αναλύει το ζήτημα από τη σκοπιά του λογισμικού, υποστηρίζοντας πως δεν είμαστε αρκετά κοντά στο να μπορούμε να προσομοιώσουμε προγραμματιστικά τις πολλές (10<sup>14</sup> σε αριθμό) διασυνδέσεις των νευρώνων του ανθρώπινου εγκεφάλου, καταλήγοντας στο ότι θα χρειαστούν ακόμη περίπου πέντε αιώνες.

## 1.5 ΙΣΤΟΡΙΚΑ ΓΕΓΟΝΟΤΑ

Οι απαρχές της τεχνητής νοημοσύνης ανάγονται στους «συλλογισμούς» του Αριστοτέλη (384-322 π.Χ.), οι οποίοι παρείχαν πρότυπα εκφράσεων που έδιναν πάντα σωστά συμπεράσματα από σωστές υποθέσεις (Αριστοτέλεια συλλογιστική). Έπειτα, οι πιο σημαντικές στιγμές είναι:

- 1854: Ο George Boole έθεσε τις βάσεις της προτασιακής λογικής.
- 1879: Ο Gottlieb Frege πρότεινε ένα σύστημα αυτοματοποιημένης συλλογιστικής και έθεσε τις βάσεις του κατηγορηματικού λογισμού (predicate calculus).
- 1943: Ο McCulloch και ο Pitts πρότειναν ένα μοντέλο τεχνητών νευρώνων που είχε τη δυνατότητα να μαθαίνει και να υπολογίζει κάθε υπολογίσιμη συνάρτηση.
- 1950: Ο Alan Turing (1913-1954), ο οποίος θεωρείται ο πατέρας της ΤΝ, εμπνεύστηκε ένα τεστ (γνωστό ως Turing test) για την αναγνώριση ευφυών μηχανών.
- 1951: Ο Minsky και ο Edmonds υλοποίησαν το πρώτο νευρωνικό δίκτυο, το SNARC με 40 νευρώνες το οποίο χρησιμοποιούσε 3.000 λυχνίες.
- 1956: Διοργάνωση συνεδρίου (workshop) καθοριστικού στη γέννηση της τεχνητής νοημοσύνης. Οργανώθηκε από τους McCarthy, Minsky, Shannon και Rochester και αφορούσε τη θεωρία αυτομάτων, νευρωνικά δίκτυα και μελέτη της ευφυΐας.
- 1958: Ο McCarthy όρισε τη συναρτησιακή γλώσσα LISP. Πρότεινε ένα υποθετικό σύστημα (τον advice taker) που χρησιμοποιούσε γνώση (όπως το LT) αλλά αφορούσε γενικά καθημερινά προβλήματα.
- 1958: Ο Friedberg πρότεινε μία τεχνική, τη μηχανική εξέλιξη (machine evolution) ή όπως ονομάζεται τώρα τους γενετικούς αλγόριθμους (genetic algorithms).
- Δεκαετία του '60: Στο Stanford υλοποιήθηκε το πρώτο robot, το Shakey robot, το πρώτο κινητό ρομπότ γενικής χρήσης που ήταν σε θέση να αιτιολογήσει τις ίδιες του ενέργειες. Ενώ άλλα ρομπότ θα έπρεπε να λάβουν οδηγίες για κάθε μεμονωμένο βήμα ολοκλήρωσης μιας μεγαλύτερης εργασίας, το Shakey μπορούσε να αναλύσει εντολές και να τις χωρίσει σε βασικά κομμάτια από μόνο του.
- 1968: Το πρόγραμμα ANALOGY του Tom Evans έλυσε προβλήματα γεωμετρικής αναλογίας που χρησιμοποιούνταν σε τεστ ευφυΐας.
- 1962: Βελτιώσεις της μεθόδου μάθησης των νευρωνικών δικτύων του Hebb από τον Rosenblatt με τα perceptrons.

- 1965: Το πρόγραμμα ELIZA του Weizenbaum μπορούσε να κάνει συζήτηση για οποιοδήποτε θέμα χρησιμοποιώντας και παραφράζοντας τις προτάσεις που έδινε σαν ερώτηση ο χρήστης.

```

Welcome to
          EEEEE LL   IIII ZZZZZ AAAAA
          EE  LL   II   ZZ  AA  AA
          EEEEE LL   II   ZZZ  AAAAAA
          EE  LL   II   ZZ  AA  AA
          EEEEE LLLLL IIII ZZZZZ AA  AA

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you ?
YOU:   Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU:   They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU:   Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU:   He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU:   It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:

```

**Εικόνα 1.2:** Συζήτηση με το ELIZA

- Δεκαετία του '70: Εποχή της κριτικής ότι τα συστήματα ήταν κατάλληλα μόνο για παιχνίδια (toy problems). Το χαρακτηριστικό των συστημάτων της εποχής ήταν ότι περιείχαν ελάχιστη ή καθόλου γνώση για το πεδίο του προβλήματος (weak methods). Το εύρος εφαρμογών των νευρωνικών δικτύων ήταν μικρό. Αναπτύχθηκαν συστήματα που περιείχαν την απαιτούμενη γνώση ώστε να συμπεριφέρονται όπως οι άνθρωποι ειδικοί σε διάφορα θέματα. Ονομάστηκαν έμπειρα συστήματα (Expert Systems) ή συστήματα γνώσης (Knowledge Systems): DENDRAL (Stanford 1969). Εύρεση της μοριακής δομής οργανικών ενώσεων με δεδομένα από φασματογράφο μάζας. MYCIN (Stanford). Διάγνωση μολύνσεων του αίματος. 13 PROSPECTOR (1979). Συμβουλές για τοποθεσίες γεώτρησης για το στοιχείο μολυβδαίνιο. R1 (McDermott, 1982 για την εταιρεία Digital Equipments). Διαμόρφωση (σύνταξη) των παραγγελιών με βάση τις ανάγκες των πελατών. SHRDLU (Winograd) και LUNAR (William Woods, 1973). Κατανόηση φυσικής γλώσσας
- Αρχές δεκαετίας του '70: Προτάθηκε η γλώσσα προγραμματισμού Prolog.
- 1975: Προτάθηκαν από τον Minsky τα πλαίσια (frames).
- 1981: Ιάπωνες επιστήμονες ανακοίνωσαν ένα πρόγραμμα της 5ης γενιάς, ένα δεκαετές πρόγραμμα για την κατασκευή υπολογιστών σε γλώσσα προγραμματισμού την Prolog. Στόχος ήταν να κατασκευαστούν ευφυή συστήματα, τα οποία εκτός των άλλων, θα ήταν σε θέση να επικοινωνούν πλήρως με τον άνθρωπο σε φυσική γλώσσα.

- Μέσα της δεκαετίας του '80: Επανεμφανίστηκαν τα νευρωνικά δίκτυα.
- Δεκαετία του '80: Εμφανίστηκε πάλι ο αλγόριθμος μάθησης με οπισθοδρόμηση (Back-propagation) και εφαρμόστηκε σε πολλά προβλήματα με μεγάλη επιτυχία.
- 1997: Ο Garry Kasparov, ηττάται από το Deep Blue της IBM. Η πρώτη μηχανή που κατάφερε να νικήσει τον παγκόσμιο πρωταθλητή του σκάκι.

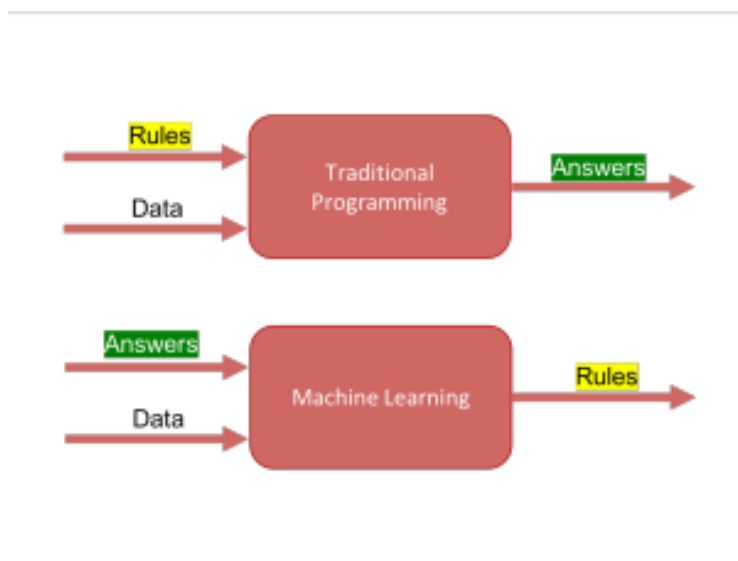
Στη σύγχρονη εποχή η Τεχνητή Νοημοσύνη έχει εξελιχθεί ώστε να καλύπτει όχι μόνο συστήματα που βασίζονται σε κανόνες και έμπειρα συστήματα αλλά και συστήματα που βασίζονται στην εξελικτική διαδικασία ή πράκτορες (agents). Τομείς έρευνας περιλαμβάνουν την ανάπτυξη της γνώσης για συλλογιστικά μοντέλα όπως οντολογίες και εφαρμογές εξόρυξης δεδομένων για την αυτόματη απόκτηση γνώσης. Σχετικοί τομείς έρευνας και ανάπτυξης περιλαμβάνουν το ηλεκτρονικό εμπόριο, την ηλεκτρονική οικονομία και τις διαπραγματεύσεις βάσει πρακτόρων. Αυτήν τη στιγμή υπάρχουν:

- 1) Συστήματα αναγνώρισης φωνής (π.χ. Pegasus), τα οποία κλείνουν αεροπορικές θέσεις τηλεφωνικά, βρίσκοντας τις βέλτιστες πτήσεις με βάση το κόστος ή το χρόνο) ή δίνουν διάφορες πληροφορίες γενικού ενδιαφέροντος.
- 2) Έμπειρα συστήματα πραγματικού χρόνου (π.χ. MARVEL) που επεξεργάζονται τα δεδομένα που μεταδίδονται από διαστημόπλοια.
- 3) Ρομποτικά συστήματα που οδηγούν αυτοκίνητα σε αυτοκινητόδρομο χρησιμοποιώντας video κάμερες και sonar.
- 4) Συστήματα που διεξάγουν ιατρικές διαγνώσεις.
- 5) Συστήματα που ελέγχουν και ρυθμίζουν την κυκλοφορία αυτοκινήτων.
- 6) Προγράμματα πράκτορες (agents) και οι αρχιτεκτονικές συστημάτων που βασίζονται σε πράκτορες (σύστημα SOAR) και πολλά άλλα
- 7) Συστήματα που ενσωματώνονται σε διάφορες γραμμές παραγωγής βιομηχανιών αποσκοπώντας στον έλεγχο ποιότητας του προϊόντος, αντικαθιστώντας τον άνθρωπο σε πολλές θέσεις εργασίας.



## 1.6 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ (MACHINE LEARNING)

Μηχανική μάθηση ονομάζουμε τη δημιουργία μοντέλων ή προτύπων από ένα σύνολο δεδομένων που προέρχονται από ένα υπολογιστικό σύστημα. Η μηχανική μάθηση ασχολείται με το σχεδιασμό και τη κατασκευή συστημάτων και αλγορίθμων ικανά να μαθαίνουν από την υπάρχουσα εμπειρία. Μάθηση σημαίνει βελτίωση σε κάποια εργασία, δεδομένου κάποιου μέτρου απόδοσης. Η έννοια της εμπειρίας αναφέρεται στην πληροφορία του παρελθόντος, δηλαδή τα δεδομένα που είτε έχουν συλλεχθεί, είτε παράγονται τεχνητά (μέσω μεθόδων προσομοίωσης) και είναι διαθέσιμα στο σύστημα. Οι τεχνικές αυτές ξεφεύγουν από το αλγοριθμικό πλαίσιο που γνωρίζουμε μέσω του κλασσικού προγραμματισμού και είναι ικανές να κάνουν ακριβείς προβλέψεις, μαθαίνοντας από τις συσχετίσεις μεταξύ των δεδομένων. Τα πιο βασικά είδη μηχανικής μάθησης είναι δύο: το πρώτο είναι η μάθηση με επίβλεψη όπου το σύστημα καλείται να μάθει από ένα σύνολο δεδομένων και το δεύτερο είναι η μάθηση χωρίς επίβλεψη όπου το σύστημα πρέπει να μάθει από τη δημιουργία προτύπων (pattern recognition).



Εικόνα 1.3: Κλασσικός προγραμματισμός vs Μηχανική Μάθηση

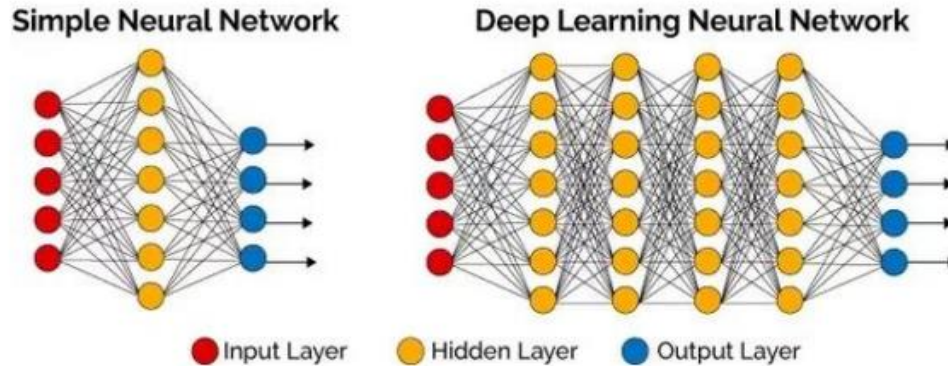
Ως επιστημονικό εγχείρημα, η μηχανική μάθηση αναπτύχθηκε από την αναζήτηση για την τεχνητή νοημοσύνη. Ήδη από την πρώιμη περίοδο της έρευνας στον τομέα της τεχνητής νοημοσύνης σε ακαδημαϊκό επίπεδο, το ζήτημα της κατασκευής μηχανών που θα μάθαιναν από δεδομένα απασχόλησε τους ερευνητές. Προσπάθησαν να προσεγγίσουν το πρόβλημα με διάφορες μεθόδους, καθώς και με τα λεγόμενα νευρωνικά δίκτυα, μοντέλα, που όπως διαπιστώθηκε αργότερα ήταν επανεφευρέσεις των γενικευμένων γραμμικών μοντέλων της στατιστικής. Τα τελευταία χρόνια, η μηχανική μάθηση έχει συμβάλει στο μέγιστο βαθμό, στην αξιοποίηση του τεράστιου όγκου δεδομένων για την εξαγωγή χρήσιμων πληροφοριών. Εφαρμόζεται σε μια σειρά από υπολογιστικές εργασίες, όπου τόσο ο σχεδιασμός όσο και ο ρητός προγραμματισμός των αλγορίθμων είναι ανέφικτος, ή η υλοποίηση κι επίλυση του προβλήματος έχει μεγάλο

βαθμό δυσκολίας. Το φίλτρο ανεπιθύμητων ηλεκτρονικών μηνυμάτων είναι ένα πολύ καλό παράδειγμα. Αν προσπαθήσουμε να υλοποιήσουμε ένα τέτοιο πρόγραμμα μέσω του κλασσικού προγραμματισμού, το αποτέλεσμα θα είναι μια μακροσκελής λίστα κανόνων και περιορισμών τις οποίες πρέπει να εντοπίσουμε και να εισάγουμε. Σε αντίθεση, ένα φίλτρο βασισμένο σε τεχνικές μηχανικής μάθησης, είναι σε θέση να μάθει αυτόματα μοτίβα και συσχετίσεις λέξεων ή φράσεων σε μηνύματα τα οποία θεωρούνται ανεπιθύμητα. Ως αποτέλεσμα έχουμε απλούστερες και πιο ακριβείς λύσεις. Μερικά χαρακτηριστικά παραδείγματα εφαρμογών αφορούν :

- Ανίχνευση όγκου σε αξονικές εγκεφάλων. Όπου αναλύεται και ταξινομείται κάθε εικονοστοιχείο (pixel) της αξονικής, με σκοπό να προσδιοριστεί το σχήμα και η ακριβής θέση του όγκου. Η τεχνική αυτή είναι γνωστή ως σημασιολογική τμηματοποίηση (Semantic segmentation), ανήκει στην ευρύτερη περιοχή της υπολογιστικής όρασης (Computer Vision).
- Πρόβλεψη των εσόδων μιας επιχείρησης τον επόμενο χρόνο. Αυτό αποτελεί ένα πρόβλημα παλινδρόμησης, το οποίο επιδέχεται διάφορα μοντέλα και τεχνικές ως μεθόδους επίλυσης, όπως είναι τα μοντέλα γραμμικής ή πολυωνυμικής παλινδρόμησης και τα τεχνητά νευρωνικά δίκτυα.
- Τμηματοποίηση πελατών βάσει των αγορών τους, ώστε να σχεδιαστεί μία διαφορετική στρατηγική πωλήσεων για κάθε τμήμα. Όπου χρησιμοποιούνται τεχνικές συσταδοποίησης (clustering).

## 1.7 ΒΑΘΙΑ ΜΑΘΗΣΗ (DEEP LEARNING)

Η “βαθιά μάθηση”(deep learning) είναι ένα υποσύνολο της μηχανικής μάθησης. Δηλαδή πρόκειται για ένα πεδίο που εξετάζει τους αλγορίθμους υπολογιστών που μαθαίνουν και βελτιώνονται μόνοι τους. Με άλλα λόγια είναι απλά ένας άλλος τρόπος για να περιγράψουμε τα μεγάλα νευρωνικά δίκτυα, μια τεχνολογία που συναντάμε κάθε μέρα όταν κάνουμε περιήγηση στο διαδίκτυο ή ακόμα και όταν χρησιμοποιούμε το κινητό μας τηλέφωνο. Η κατάρτιση ενός μοντέλου βαθιάς μάθησης απαιτεί πολλά δεδομένα. Μάλιστα όσο περισσότερα είναι τα δεδομένα με τα οποία τροφοδοτείται τόσο πιο ακριβές θα είναι και το μοντέλο deep learning . Οι επιστήμονες έχουν καταφέρει να προσεγγίσουν όλο και περισσότερο την κατασκευή μοντέλων βαθιάς μάθησης που έχουν μεγαλύτερη ακρίβεια και που μπορούν να μάθουν χωρίς επίβλεψη. Έτσι το deep learning θα γίνει ταχύτερο και θα απαιτεί λιγότερη δουλειά. Αυτό φυσικά σημαίνει μεγαλύτερα και καλύτερα πράγματα για το μέλλον αυτών των μοντέλων.



**Εικόνα 1.4:** Απεικόνιση ενός απλού NN και ενός DNN

Τα τελευταία χρόνια η “βαθιά μάθηση” προτιμάτε περισσότερο από την κλασική μηχανική μάθηση για τα προβλήματα τεχνητής νοημοσύνης. Ο λόγος για αυτό είναι ότι το deep learning έχει επανειλημμένα αποδείξει ότι έχει καλύτερη απόδοση σε μια ευρεία ποικιλία λειτουργιών όπως στη φυσική γλώσσας και τα παιχνίδια. Παρόλο που το deep learning έχει τόσο υψηλές επιδόσεις εξακολουθούν να υπάρχουν μερικά πλεονεκτήματα στη χρήση της κλασικής μηχανικής μάθησης και σε ορισμένες περιπτώσεις είναι προτιμότερη για χρήση.

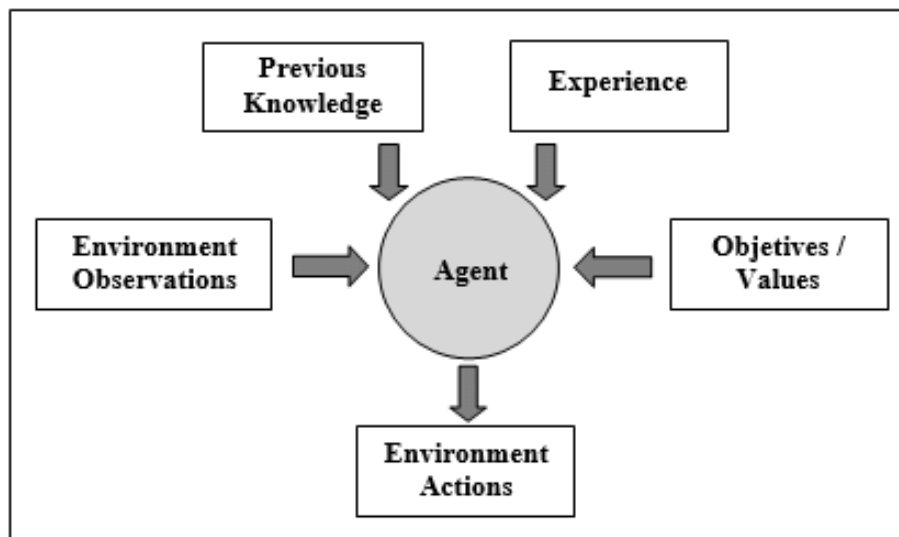
## 1.8 ΣΧΕΔΙΑΣΜΟΣ ΕΝΕΡΓΕΙΩΝ ΚΑΙ ΧΡΟΝΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Ο χρονοπρογραμματισμός είναι ένα πάγιο πρόβλημα που εμφανίζεται σχεδόν σε κάθε βιομηχανική εφαρμογή. Βασικά Χρονοπρογραμματισμός είναι η διαδικασία κατά την οποία ορίζουμε μέσα σε ένα πεπερασμένο χρονικό διάστημα πόρους σε εργασίες με μοναδικά δεδομένα μας, ένα κριτήριο και κάποιους περιορισμούς. Ο ρόλος του χρονοπρογραμματισμού στις βιομηχανικές εφαρμογές είναι πολύ σημαντικός. Προκειμένου να λύσουμε ένα πρόβλημα τέτοιου είδους κάνουμε χρήση κατάλληλων αλγορίθμων Μηχανικής Μάθησης.

## 1.9 ΕΥΦΥΕΙΣ ΠΡΑΚΤΟΡΕΣ (INTELLIGENT AGENTS)

Ένα έξυπνο σύστημα επεξεργάζεται πληροφορίες προκειμένου να κάνει κάτι σκόπιμο. Κάποια γνωστά έξυπνα συστήματα είναι οι υπολογιστές και τα ρομπότ. Ένα ακόμα είδος έξυπνου-ευφυούς συστήματος που έχει αρκετό ενδιαφέρον για τους επιστήμονες σε όλο τον κόσμο είναι αυτό του ευφυούς πράκτορα. Ένα σύστημα τέτοιου είδους ενεργεί με βάση τις πληροφορίες που αντιλαμβάνεται. Οι ευφυείς πράκτορες

αποτελούν τεχνητούς πράκτορες όπου κατασκευάζονται από τον άνθρωπο. Έχουν την ικανότητα να ενεργούν με βάση την εμπειρία τους αλλά και τις πληροφορίες τις οποίες αντιλαμβάνονται. Επίσης αναζητούν το καλύτερο σχέδιο δράσης για την κατάσταση την οποία καλούνται να επιλύσουν ώστε να πετύχουν το καλύτερο δυνατό αποτέλεσμα, αυτό σημαίνει ότι διαλέγουν τον πιο σύντομο δρόμο με την μικρότερη κατανάλωση πόρων. Αξίζει να σημειώσουμε πως οι ευφυείς πράκτορες έχουν εφαρμογή σε τομείς όπως για παράδειγμα τη δικτύωση, την ιατρική και το ηλεκτρονικό εμπόριο κλπ. Κάποια παραδείγματα από έξυπνους πράκτορες είναι η Siri και η Alexa, καθώς χρησιμοποιούν αισθητήρες όπως μικρόφωνα και άλλες εισόδους για να κατανοήσουν ένα αίτημα και αντλούν πληροφορίες από τη συλλογική τους εμπειρία και γνώση μέσω υπερυπολογιστών και από τράπεζες δεδομένων (servers) σε όλο τον κόσμο ώστε να πάρουν μια απόφαση. Ο όγκος των πληροφοριών που οι ευφυείς πράκτορες εκτίθενται μεγαλώνουν και εξελίσσονται στιγμή προς στιγμή, με την πάροδο του χρόνου γνωρίζουν περισσότερα για τους χρήστες τους και συλλέγουν νέα δεδομένα όπως παρουσιάζονται στον κόσμο τη στιγμή εκείνη.



Εικόνα 1.5: Λειτουργία Ευφυούς Πράκτορα

## 1.10 ΕΥΦΥΕΙΣ ΥΠΗΡΕΣΙΕΣ ΔΙΑΔΙΚΤΥΟΥ (SEMANTIC WEB)

Με απλούστερα λόγια αναφερόμαστε στην επέκταση του παγκόσμιου δικτύου (world wide web) όπου επιτρέπει στους χρήστες να μοιράζονται περιεχόμενο που είναι πέρα από τα όρια των εφαρμογών και των ιστοτόπων. Οι υπολογιστές έχουν πρόσβαση σε μία παγκόσμια κοινή βάση δεδομένων και σε ένα πλήθος από συμπερασματικούς κανόνες με σκοπό την αυτόματη παραγωγή συμπερασμάτων. Το σημασιολογικό δίκτυο βασίζεται στο συνδυασμό των οντολογιών, των ρητών μεταδεδομένων meta data και τη λογική συλλογιστική. Συγκεκριμένα η λογική συλλογιστική καθιστά δυνατή την εξαγωγή

συμπερασμάτων από το συνδυασμό μεταδεδομένων με οντολογίες. Οι οντολογίες περιγράφουν βασικές έννοιες ενός τομέα και τις σχέσεις τους όπως για παράδειγμα η οντολογία του σχολείου μπορεί να περιέχει έννοιες όπως οι τάξεις και τα μαθήματα. Τέλος τα ρητά μεταδεδομένα επιτρέπουν στις ιστοσελίδες να εξάγουν περαιτέρω πληροφορίες. Για παράδειγμα από την ιστοσελίδα ενός επιστήμονα μπορούν να προσδιοριστούν το όνομα και οι μελέτες του.

## **1.11 ΠΡΟΣΑΡΜΟΖΩΜΕΝΑ – ΕΞΕΛΙΣΣΟΜΕΝΑ ΕΥΦΥΗ ΣΥΣΤΗΜΑΤΑ**

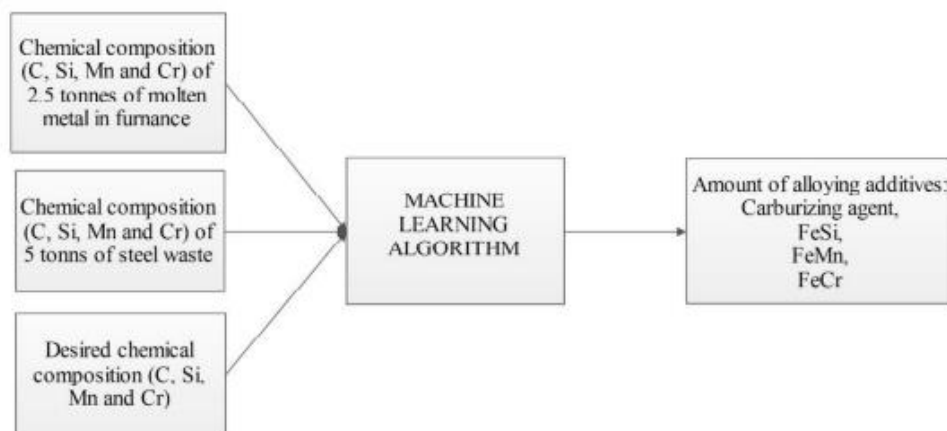
Είναι τεχνολογικά εξελεγμένα μηχανήματα τα οποία αντιλαμβάνονται τον κόσμο γύρω τους και φυσικά ανταποκρίνονται. Τα ευφυή συστήματα μπορούν να λάβουν πολλές μορφές. Παρόλο όμως που λέμε ευφυή συστήματα δεν έχει επιτευχθεί ακόμη πλήρης νοημοσύνη. Η τεχνολογία μπορεί να δώσει νέες και σημαντικές δυνατότητες σε πλήθος συστημάτων σε όλους τους τομείς δυνατότητες οι οποίες θα έχουν επίδραση στη βελτίωση των προσφερόμενων υπηρεσιών, την αύξηση της παραγωγικότητας και την αναβάθμιση της ποιότητας ζωής. Ειδικά οι βιομηχανίες δίνουν μεγάλη έμφαση στην ανάπτυξη καινοτόμων έξυπνων αλγορίθμων και με ιδιαίτερη έμφαση στα ευφυή ρομποτικά συστήματα. Έτσι παρουσιάζεται η ανάγκη για επένδυση σε ευφυή συστήματα προσομοίωσης, ευφυή συστήματα πληροφοριών, ευφυή συστήματα υποστήριξης και ευφυή robot.

## 2. ΕΦΑΡΜΟΓΕΣ ΤΕΧΝΗΤΗΣ ΝΟΗΜΟΣΥΝΗΣ ΣΤΟΥΣ ΤΟΜΕΙΣ ΤΗΣ ΜΗΧΑΝΙΚΗΣ

Η τεχνητή νοημοσύνη στη σύγχρονη εποχή μπορεί να εντοπιστεί πραγματικά σε αμέτρητους τομείς και ιδίως στον τομέα της μηχανικής (engineering), με εφαρμογές που ποικίλουν σε είδος. Παρακάτω θα αναφέρουμε μερικές ενδεικτικές από αυτές. Ονομαστικά αυτές θα είναι εφαρμογές τεχνητής νοημοσύνης και μηχανικής μάθησης στην μεταλλουργία, στην αεροδιαστημική, στη βιομηχανία και στην αυτοκινητοβιομηχανία.

### 2.1 ΕΛΕΓΧΟΣ ΔΙΑΔΙΚΑΣΙΑΣ ΤΗΞΗΣ ΜΕΤΑΛΛΟΥ

Ο έλεγχος με στόχο την επίτευξη της επιθυμητής χημικής σύνθεσης, του λευκού χυτοσίδηρου, βασίζεται στην εφαρμογή του αλγορίθμου μηχανικής μάθησης. Ο αλγόριθμος μηχανικής μάθησης πραγματοποιεί πρόβλεψη της ποσότητας των πρόσθετων κραμάτων που είναι αναγκαία για την επίτευξη της επιθυμητής χημικής σύνθεσης του λευκού χυτοσίδηρου. Σύμφωνα με την ιδέα ελέγχου που παρουσιάζεται στο παρακάτω σχήμα, τα δεδομένα που συλλέγονται στο χυτήριο χωρίζονται σε εισόδους (inputs) και εξόδους (outputs) για την ανάπτυξη του αλγορίθμου μηχανικής μάθησης.



**Εικόνα 2.1 :** Παραγωγή χυτοσίδηρου με επισυναπτόμενο νευρωνικό δίκτυο.

Το σύνολο των δεδομένων εισόδου για την εκπαίδευση του αλγορίθμου μηχανικής μάθησης αποτελείται από από: μια χημική σύνθεση 2,5 τόνων λιωμένου μετάλλου στον κλίβανο, μια χημική σύνθεση 5 τόνων αποβλήτων χάλυβα και μια τελική χημική σύνθεση του λιωμένου μετάλλου. Το σύνολο των δεδομένων εξόδου για τον αλγόριθμο εκπαίδευσης του νευρωνικού δικτύου μηχανικής μάθησης είναι η ποσότητα των κραμάτων που προστίθενται κατά τη διάρκεια της διαδικασίας κραματοποίησης.

Η τήξη μετάλλων είναι μια πολύ σύνθετη διεργασία λόγω της πολύπλοκης αλληλεπίδρασης μεταξύ διάφορων παραμέτρων, όπως η θερμικές απώλειες και η δυναμική των μη γραμμικών χημικών συστημάτων. Η πολυπλοκότητα και η μη

γραμμικότητα αυτής της παραγωγικής διαδικασίας αποτέλεσαν μια ερευνητική πρόκληση και με την εφαρμογή αλγορίθμων μηχανικής μάθησης επιτεύχθηκε η βελτίωσή της.

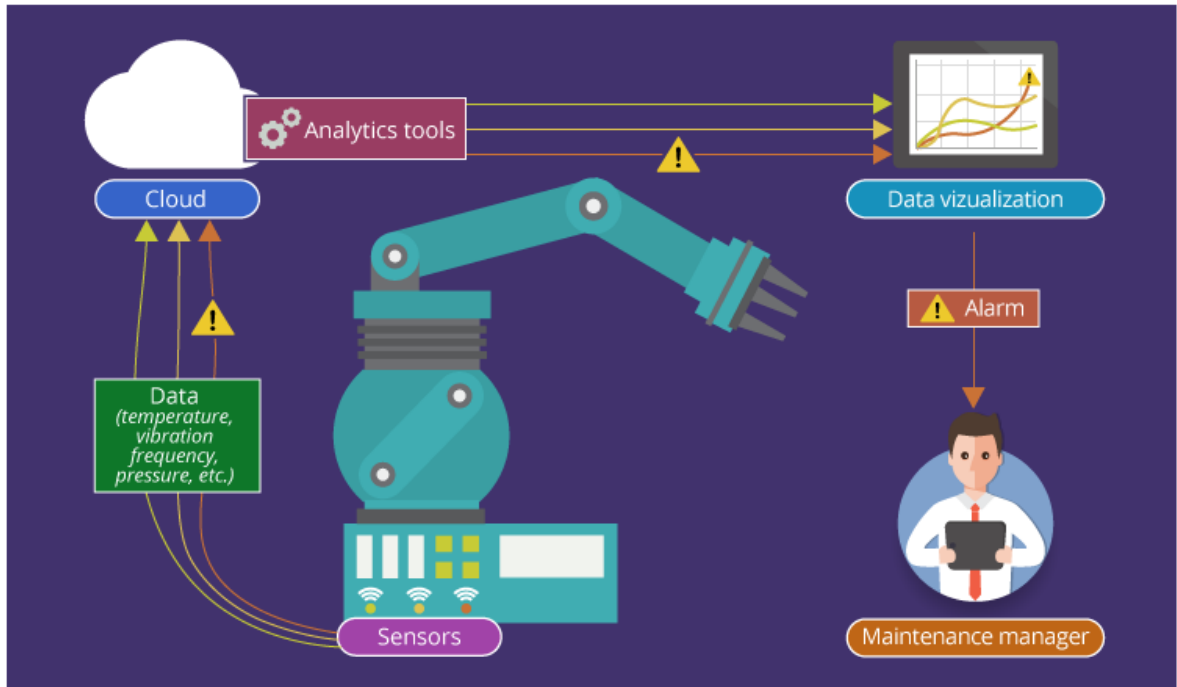
## 2.2 CONDITION MONITORING - FAULT DETECTION

Το Condition Monitoring (κοινώς, CM) είναι η διαδικασία παρακολούθησης μιας παραμέτρου της κατάστασης σε μηχανήματα (κραδασμοί, θερμοκρασία κ.λπ.), προκειμένου να εντοπιστεί κάποια σημαντική αλλαγή αποτελώντας ένδειξη μιας αναπτυσσόμενης βλάβης. Αποτελεί σημαντικό στοιχείο της προληπτικής συντήρησης. Η χρήση της παρακολούθησης της κατάστασης επιτρέπει τον προγραμματισμό της συντήρησης ή τη λήψη άλλων μέτρων για την πρόληψη επακόλουθων ζημιών και την αποφυγή των συνεπειών τους. Η παρακολούθηση της κατάστασης έχει ένα μοναδικό πλεονέκτημα, καθώς οι συνθήκες που θα μείωναν την κανονική διάρκεια ζωής μπορούν να αντιμετωπιστούν πριν εξελιχθούν σε σημαντική βλάβη. Οι τεχνικές παρακολούθησης της κατάστασης χρησιμοποιούνται συνήθως σε περιστρεφόμενο εξοπλισμό, βοηθητικά συστήματα και άλλα μηχανήματα (συμπιεστές, αντλίες, ηλεκτροκινητήρες, κινητήρες εσωτερικής καύσης, πρέσες, κ.λ.π.), ενώ η περιοδική επιθεώρηση με τη χρήση τεχνικών μη καταστρεπτικών δοκιμών (NDT) και η αξιολόγηση καταλληλότητας για χρήση χρησιμοποιούνται για στατικό εξοπλισμό εγκαταστάσεων, όπως λέβητες ατμού, σωληνώσεις και εναλλάκτες θερμότητας.

Έως και σήμερα οι πιο συνηθισμένοι τρόποι παρακολούθησης μιας παραμέτρου κατάστασης είναι οι εξής:

- Ανάλυση Κραδασμών (vibration analysis ή αλλιώς AE analysis)
- Θερμογραφία Υπερύθρων (infrared thermography)
- Ακουστική Δοκιμή (acoustic emission)
- Διαχείριση Λίπανσης (Lubrication Management)

Οι τέσσερις παραπάνω τρόποι παρακολούθησης γίνονται με την χρήση μηχανικής μάθησης και τεχνητών νευρωνικών δικτύων, όπου με την σημερινή υπολογιστική ισχύ που διαθέτουμε, τα τεράστια πόσα δεδομένων που έχουμε μπορούμε να τα εκμεταλευτούμε πλήρως. Ο τρόπος με τον οποίο αυτό γίνεται έχει ως εξής: Αρχικά αισθητήρες εξωτερικοί εντοπισμού κραδασμών, ηλεκτρικών σημάτων καθώς και κάμερες υπέρυθρων τοποθετούνται κατάλληλα, ώστε να μπορούν να πάρουν μετρήσεις ανενόχλητα, στην συνέχεια αυτές οι μετρήσεις αποθηκεύονται σε ένα λεγόμενο αποθηκευτικό νέφος ή αλλιώς cloud storage. Έπειτα τεχνητά νευρωνικά δίκτυα πολλαπλών στρωμάτων, ειδικευμένα στην αναγνώριση προτύπων, που λειτουργούν απο έναν κεντρικό υπολογιστή ελέγχου, έχουν την δυνατότητα σε σχετικά μικρά χρονικά διαστήματα να αναλύσουν όλο αυτό τον όγκο δεδομένων και να εξαγάγουν χρήσιμα αποτελέσματα που αφορούν την φθορα του όποιου μηχανήματος ή μέρους αυτού.



Εικόνα 2.2: Απεικόνιση παρακολούθησης παραμέτρων κατάστασης

## 2.3 ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ ΣΤΗΝ ΑΥΤΟΚΙΝΗΤΟΒΙΟΜΗΧΑΝΙΑ

Η αυτοκινητοβιομηχανία έχει μακρόχρονο ιστορικό αξιοποίησης των τελευταίων τεχνολογιών για την κυκλοφορία αποδοτικών, καινοτόμων και ασφαλών οχημάτων στην αγορά, ενώ παράλληλα εργάζεται συνεχώς για τη μείωση του κόστους κατασκευής. Σήμερα, οι τεχνολογίες αυτές περιλαμβάνουν την τεχνητή νοημοσύνη και τους υπολογιστές υψηλών επιδόσεων HPC, που αποτελούν δύο κλειδιά για την επιτυχία της αυτοκινητοβιομηχανίας.

Ενώ για πολλά χρόνια ο κλάδος επένδυε σε μεγάλο βαθμό σε συστήματα HPC για την τροφοδοσία εφαρμογών μοντελοποίησης, σχεδιασμού και προσομοίωσης, σήμερα η εστίαση αυτή διευρύνεται και περιλαμβάνει σημαντικές επενδύσεις στην τεχνητή νοημοσύνη για την οδήγηση αυτόνομων και ημιαυτόνομων οχημάτων. Οι κατασκευαστές αυτοκινήτων σε όλο τον κόσμο αναγνωρίζουν ότι αυτά τα πιο έξυπνα οχήματα είναι σαφώς το μέλλον και γνωρίζουν ότι δεν μπορούν να φτάσουν εκεί χωρίς τεχνητή νοημοσύνη.

Η τεχνητή νοημοσύνη χρησιμοποιείται επίσης για ένα ευρύ φάσμα προγνωστικών δυνατοτήτων που εξατομικεύουν την εμπειρία οδήγησης και επιτρέπουν την προληπτική συντήρηση. Αυτή η εξατομίκευση ενεργοποιείται από τα συνδεδεμένα οχήματα που στέλνουν δεδομένα από τους αισθητήρες επί του οχήματος σε κατασκευαστές, διαφημιστές και ασφαλιστικές εταιρείες.

Αυτές οι προηγμένες δυνατότητες, σε συνδυασμό με τις αυξανόμενες προσδοκίες των καταναλωτών, έχουν ωθήσει την αυτοκινητοβιομηχανία σε μια περίοδο ψηφιακού μετασχηματισμού. Οι σημερινοί κατασκευαστές γνωρίζουν ότι πρέπει να αξιοποιήσουν



τις νέες τεχνολογίες - όπως η τεχνητή νοημοσύνη, η μηχανική μάθηση και η βαθιά μάθηση, για να μειώσουν το κόστος και να προσφέρουν στους οδηγούς περισσότερα από αυτά που θέλουν.

Σε ένδειξη της αυξανόμενης σημασίας των αναδυόμενων τεχνολογιών για τον κλάδο, η αγορά της τεχνητής νοημοσύνης στην αυτοκινητοβιομηχανία αναμένεται να αυξηθεί με σύνθετο ετήσιο ρυθμό ανάπτυξης (CAGR) σχεδόν 40% έως το 2025.

Οι περιπτώσεις χρήσης της τεχνητής νοημοσύνης καλύπτουν πλέον το σύνολο της οδηγικής εμπειρίας. Ακολουθούν μερικοί από τους ποικίλους τρόπους με τους οποίους οι κατασκευαστές αυτοκινήτων χρησιμοποιούν την τεχνητή νοημοσύνη και τη δύναμη του HPC για να προσφέρουν μια ασφαλέστερη, πιο αποτελεσματική εμπειρία οδήγησης, ενώ παράλληλα να εξορθολογήσουν τις διαδικασίες τους για να περιορίσουν το κόστος

- Υποβοήθηση οδηγού (Driver assist): Με προηγμένες λειτουργίες υποβοήθησης οδηγού, πολλές από τις οποίες είναι διαθέσιμες στα σημερινά αυτοκίνητα και φορτηγά, τα συστήματα AI προειδοποιούν τους οδηγούς για επικίνδυνες οδικές συνθήκες, παρακολουθούν τα τυφλά σημεία στην οπτική γωνία του οδηγού, βοηθούν στο τιμόνι και αναλαμβάνουν αυτοματοποιημένες ενέργειες για να βοηθήσουν τα οχήματα να αποφύγουν ατυχήματα και επικίνδυνες καταστάσεις.
- Αυτόνομα οχήματα: Στην αυτοκινητοβιομηχανία, τα αυτόνομα οχήματα είναι το νέο ιερό δισκοπότηρο. Οι κατασκευαστές και οι τεχνολογικοί συνεργάτες τους εργάζονται υπερωριακά για την ανάπτυξη συστημάτων με βάση την Τεχνητή Νοημοσύνη, ώστε να καταστεί δυνατή η αυτοοδήγηση αυτοκινήτων και φορτηγών. Τα συστήματα αυτά ενσωματώνουν ένα ευρύ φάσμα τεχνολογιών με δυνατότητα τεχνητής νοημοσύνης, όπως νευρωνικά δίκτυα βαθιάς μάθησης, επεξεργασία φυσικής γλώσσας και λειτουργίες ελέγχου χειρονομιών, για να παρέχουν τον εγκέφαλο για οχήματα που μπορούν να οδηγούν με ασφάλεια μόνα τους, με ή χωρίς ανθρώπινο οδηγό στο όχημα.
- Συνδεδεμένα οχήματα (Connected vehicles): Η τεχνητή νοημοσύνη αποτελεί βασική τεχνολογία για τα συνδεδεμένα οχήματα. Για παράδειγμα, η TN μπορεί να παρακολουθεί και να προβλέπει βλάβες εξαρτημάτων, ώστε οι κατασκευαστές και οι ιδιοκτήτες οχημάτων να μπορούν να δρουν προληπτικά για την αποφυγή προβλημάτων. Μπορεί επίσης να παρέχει στους οδηγούς πληροφορίες με βάση την τοποθεσία και εξατομικευμένη διαφήμιση για να τους βοηθήσει να βρουν τα πράγματα που χρειάζονται. Παρομοίως, τα συστήματα με TN μπορούν να στέλνουν δεδομένα οδήγησης και ατυχημάτων σε ασφαλιστικές εταιρείες, οι οποίες θα μπορούσαν να προσφέρουν κίνητρα για ασφαλείς συνήθειες οδήγησης.
- Παραγωγή (Manufacturing): Η τεχνητή νοημοσύνη επιτρέπει εφαρμογές που καλύπτουν το χώρο κατασκευής αυτοκινήτων. Οι αυτοκινητοβιομηχανίες μπορούν να χρησιμοποιήσουν συστήματα με τεχνητή νοημοσύνη για τη δημιουργία χρονοδιαγραμμάτων και τη διαχείριση ροών εργασίας, να επιτρέψουν στα ρομπότ να εργάζονται με ασφάλεια μαζί με τους ανθρώπους στα εργοστάσια και τις γραμμές συναρμολόγησης και να εντοπίζουν ελαττώματα σε εξαρτήματα που μπαίνουν σε αυτοκίνητα και φορτηγά. Αυτές οι δυνατότητες μπορούν να βοηθήσουν τους

κατασκευαστές να μειώσουν το κόστος και τον χρόνο διακοπής της λειτουργίας των γραμμών παραγωγής, παρέχοντας παράλληλα καλύτερα τελικά προϊόντα στους καταναλωτές.

- Ποιοτικός έλεγχος (Quality control): Μελέτη του McKinsey Global Institute υπογραμμίζει ορισμένα από τα μοναδικά πλεονεκτήματα της χρήσης τεχνητής νοημοσύνης για την επιθεώρηση εξαρτημάτων και προϊόντων της αυτοκινητοβιομηχανίας. Σε ένα από αυτά τα πλεονεκτήματα, τα συστήματα TN βελτιώνονται με την πάροδο του χρόνου στον εντοπισμό ελαττωμάτων. "Το σύστημα AI μαθαίνει συνεχώς να βελτιώνει την ανάλυσή του με βάση την ανατροφοδότηση", σημειώνει η McKinsey. "Χρησιμοποιώντας αυτές τις μεθόδους, το υλικό με τεχνητή νοημοσύνη μπορεί να επιθεωρεί οπτικά και να παρέχει ανώτερο QC σε διάφορα προϊόντα, όπως μηχανικά κατεργασμένα εξαρτήματα, βαμμένα αμαξώματα, μεταλλικές επιφάνειες με υφή και άλλα. "
- Εφοδιαστική αλυσίδα (supply chain): Στη σημερινή παγκόσμια οικονομία, οι κατασκευαστές αυτοκινήτων διαθέτουν εξαιρετικά πολύπλοκες αλυσίδες εφοδιασμού που εκτείνονται σε πολλές γεωγραφικές περιοχές. Οποιαδήποτε δυσλειτουργία ή βλάβη στην αλυσίδα εφοδιασμού μπορεί να είναι εξαιρετικά δαπανηρή. Με την Τεχνητή Νοημοσύνη, οι κατασκευαστές μπορούν να αποκτήσουν μεγαλύτερο έλεγχο των αλυσίδων εφοδιασμού τους, συμπεριλαμβανομένων των διαδικασιών σχεδιασμού, λογιστικής, παρακολούθησης και διαχείρισης των αποθεμάτων. Για παράδειγμα, τα συστήματα με βάση την TN μπορούν να προβλέψουν πολύπλοκες αλληλεπιδράσεις μεταξύ των μονάδων παραγωγής και να αυτοματοποιήσουν τις ανάγκες για ανταλλακτικά, ανθρώπινο δυναμικό, εργαλεία και επισκευές.

## 2.4 ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ ΣΤΗΝ ΑΕΡΟΔΙΑΣΤΗΜΙΚΗ

Ο τομέας της αεροδιαστημικής αντιμετωπίζει σημαντικές προκλήσεις, όπως το εργατικό κόστος, τα ανθρώπινα λάθη και οι ανησυχίες για την υγεία και την ασφάλεια. Μαζί με αυτές τις προκλήσεις, οι διαδικασίες κατασκευής και ανάπτυξης μπορεί να είναι όλο και πιο χρονοβόρες λόγω των βιομηχανικών επιθεωρήσεων που είναι απαραίτητες για να αξιολογηθεί αν ένα εξάρτημα ανταποκρίνεται στις απαιτούμενες προδιαγραφές. Ως εκ τούτου, η αεροδιαστημική βιομηχανία αναζητά συνεχώς αποτελεσματικούς τρόπους για την επιτάχυνση των διαδικασιών ανάπτυξης, προκειμένου να ανταποκριθεί στην αυξανόμενη ζήτηση καθώς και να παραδώσει εξαρτήματα υψηλής ποιότητας.

Η τεχνητή νοημοσύνη έχει επιδείξει ανυπολόγιστες δυνατότητες σε πολλές βιομηχανίες, και η αεροδιαστημική δεν αποτελεί εξαίρεση. Η εφαρμογή της τεχνητής νοημοσύνης στην ανάπτυξη της αεροδιαστημικής μπορεί να επιτρέψει στις επιχειρήσεις να βελτιώσουν την παραγωγή διαφόρων εξαρτημάτων και να μειώσουν τα ζητήματα ασφάλειας. Επίσης, τα συστήματα TN μπορούν να αξιολογούν την ανατροφοδότηση από πολλαπλά μέσα και να επεξεργάζονται μεγάλο όγκο δεδομένων σε μικρότερο χρονικό διάστημα σε σύγκριση με τις χειροκίνητες επιθεωρήσεις. Με αυτόν τον τρόπο, οι αεροδιαστημικές επιχειρήσεις

μπορούν να διεξάγουν αποτελεσματικές και ταχύτερες επιθεωρήσεις πολλαπλών περιουσιακών στοιχείων. Η χρήση της τεχνητής νοημοσύνης στην αεροδιαστημική θα οδηγήσει επίσης στην ανάπτυξη πολλαπλών εφαρμογών που μπορούν να εξοικονομήσουν καύσιμα, να βελτιώσουν τη λειτουργική αποδοτικότητα και να ελέγξουν την εναέρια κυκλοφορία. Η αξιοποίηση της τεχνητής νοημοσύνης στην αεροδιαστημική βοηθάει την αεροδιαστημική με τους ακόλουθους τρόπους:

- Σχεδιασμός προϊόντος:

Στην αεροδιαστημική βιομηχανία, τα ελαφριά και ανθεκτικά εξαρτήματα είναι πάντα προτιμότερα για ένα αεροσκάφος. Για την ανάπτυξη τέτοιων εξαρτημάτων, οι κατασκευαστές μπορούν να χρησιμοποιήσουν τον παραγωγικό σχεδιασμό μαζί με αλγορίθμους τεχνητής νοημοσύνης. Ο γεννητικός σχεδιασμός είναι μια επαναληπτική διαδικασία, όπου οι μηχανικοί ή οι σχεδιαστές χρησιμοποιούν τους στόχους σχεδιασμού ως δεδομένα εισόδου μαζί με περιορισμούς και παραμέτρους όπως τα υλικά, οι διαθέσιμοι πόροι και ο διατιθέμενος προϋπολογισμός για την ανάπτυξη ενός βέλτιστου σχεδιασμού προϊόντος. Σε συνδυασμό με την τεχνητή νοημοσύνη, το λογισμικό γεννητικού σχεδιασμού μπορεί να επιτρέψει στους σχεδιαστές προϊόντων να διερευνήσουν πολυάριθμες επιλογές σχεδιασμού σε σύντομο χρονικό διάστημα. Χρησιμοποιώντας αυτή την τεχνολογία, οι σχεδιαστές μπορούν να αναπτύξουν νέα προϊόντα που είναι ελαφριά και βιώσιμα. Ο γεννητικός σχεδιασμός με δυνατότητα TN σε συνδυασμό με την τρισδιάστατη εκτύπωση μπορεί να χρησιμοποιηθεί για την παραγωγή διαφόρων εξαρτημάτων αεροσκαφών, όπως τουρμπίνες και πτέρυγες. Με αυτόν τον τρόπο, η εφαρμογή της TN στις εταιρείες αεροδιαστημικής μπορεί να εξορθολογήσει τις διαδικασίες σχεδιασμού και κατασκευής.

- Λειτουργική αποδοτικότητα και συντήρηση:

Τα αεροπλάνα διαθέτουν πολλαπλούς αισθητήρες που βοηθούν τους πιλότους να μετρήσουν την ταχύτητα, την πίεση του αέρα και το ύψος. Αυτοί οι αισθητήρες μπορούν να χρησιμοποιηθούν για τη συλλογή κρίσιμων δεδομένων όπως η θερμοκρασία, η υγρασία και η πίεση σε διάφορα μέρη ενός αεροσκάφους. Τα μοντέλα τεχνητής νοημοσύνης μπορούν να εκπαιδευτούν για την ανάλυση των συλλεγόμενων δεδομένων ώστε να εντοπίζουν μη φυσιολογική συμπεριφορά στα εξαρτήματα του αεροσκάφους. Για παράδειγμα, οι αισθητήρες που είναι εγκατεστημένοι στις τουρμπίνες μπορούν να συλλέγουν δεδομένα όπως η ταχύτητα περιστροφής, η πίεση του αέρα και η θερμοκρασία του εξαρτήματος. Τα δεδομένα που συλλέγονται μπορούν να χρησιμοποιηθούν για την εκπαίδευση μοντέλων τεχνητής νοημοσύνης σχετικά με την κανονική συμπεριφορά της τουρμπίνας. Αναλύοντας αυτά τα δεδομένα, τα μοντέλα TN μπορούν να ανιχνεύουν τότε οι τουρμπίνες ξεφεύγουν από την κανονική τους συμπεριφορά και να ειδοποιούν το ενδιαφερόμενο προσωπικό για πιθανά ελαττώματα. Ως εκ τούτου, οι αεροπορικές εταιρείες μπορούν να εντοπίζουν εκ των προτέρων τα ελαττωματικά εξαρτήματα των αεροσκαφών και να τα επισκευάζουν. Με αυτόν τον τρόπο, η αξιοποίηση της TN στις εταιρείες αεροδιαστημικής μπορεί να βοηθήσει τους επιχειρηματίες να βελτιώσουν την

επιχειρησιακή τους αποδοτικότητα, αποφεύγοντας τις βλάβες εξαρτημάτων που μπορεί να οδηγήσουν σε διακοπές λειτουργίας.

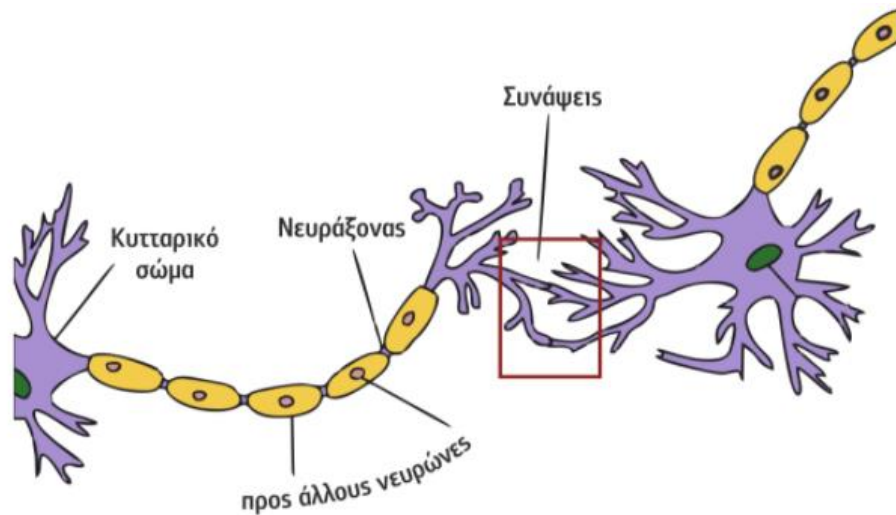
- Εντοπισμός απειλών:

Η ΤΝ μπορεί να χρησιμοποιηθεί για τον εντοπισμό και την κατηγοριοποίηση απειλών και κινδύνων με τη βοήθεια της μηχανικής όρασης, της μηχανικής μάθησης και της επεξεργασίας σημάτων. Για τον σκοπό αυτό, τα μοντέλα ΤΝ μπορούν να εκπαιδευτούν χρησιμοποιώντας εικόνες και βίντεο που λαμβάνονται από διάφορα εναέρια οχήματα και δορυφόρους. Οι συσσωρευμένες εικόνες και βίντεο μπορούν να επισημανθούν ως κανονικές ή ύποπτες. Χρησιμοποιώντας αυτά τα δεδομένα, οι αλγόριθμοι ΤΝ μπορούν να εντοπίσουν απειλές σε διάφορα σενάρια. Τέτοιες εφαρμογές που βασίζονται στην ΤΝ μπορούν να χρησιμοποιηθούν για εμπορικούς, πολιτικούς, καθώς και στρατιωτικούς σκοπούς. Ως εκ τούτου, αξιοποιώντας την ΤΝ στην αεροδιαστημική, οι επικεφαλής των επιχειρήσεων μπορούν να βοηθήσουν τους πιλότους να λαμβάνουν τεκμηριωμένες αποφάσεις με τη βοήθεια της επίγνωσης του χώρου και της κατάστασης.

### 3. ΕΙΣΑΓΩΓΗ ΣΤΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΤΑ ΤΕΧΝΗΤΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ

#### 3.1 ΒΙΟΛΟΓΙΚΗ ΕΜΠΝΕΥΣΗ

Η έμπνευση για κάθε μορφής νευρωνικό δίκτυο ξεκινά από την βιολογία. Οι ζωντανοί οργανισμοί ανεξίτητος, έχουν ένα νευρικό σύστημα, το οποίο είναι υπεύθυνο για μια σειρά διεργασιών, όπως είναι η επαφή με το εξωτερικό περιβάλλον, η μάθηση, η μνήμη, κλπ. Το νευρικό σύστημα των οργανισμών αποτελείται από πολλά Νευρωνικά δίκτυα τα οποία είναι εξειδικευμένα στις διεργασίες αυτές. Κάθε νευρωνικό δίκτυο αποτελείται από ένα μεγάλο αριθμό μονάδων, που λέγονται νευρώνες (neurons).



Εικόνα 3.1: Απεικόνιση Φυσικού νευρώνα

Ο νευρώνας είναι η πιο μικρή ανεξάρτητη μονάδα του δικτύου, όπως π.χ. το άτομο είναι η πιο μικρή μονάδα της ύλης. Οι νευρώνες συνεχώς και ασταμάτητα επεξεργάζονται πληροφορίες, παίρνοντας και στέλνοντας ερεθίσματα σε άλλους νευρώνες. Η μάθηση προκύπτει με την αλλαγή της αποτελεσματικότητας των συνάψεων, έτσι ώστε η επιρροή ενός νευρώνα σε έναν άλλο να αλλάζει.

Βλέπουμε λοιπόν ότι οι πρώτες γνώσεις μας για τα Νευρωνικά δίκτυα προέρχονται από την βιολογία και την ιατρική. Σήμερα διεξάγεται ένας μεγάλος αριθμός ερευνών σ' αυτούς τους δύο κλάδους επιστημών για την καλύτερη κατανόηση των νευρωνικών δικτύων του εγκεφάλου. Είναι προφανές ότι αυτό θα μας βοηθήσει να διαλευκάνουμε το πώς ακριβώς λειτουργεί ο εγκέφαλος, και τις τόσο περίπλοκες διεργασίες του, όπως πώς σκεπτόμαστε, πώς θυμόμαστε, κλπ. Έτσι λοιπόν, τα Νευρωνικά δίκτυα των ζωντανών οργανισμών τα ονομάζουμε βιολογικά Νευρωνικά δίκτυα.

Οι διεργασίες που επιτελούνται από τα βιολογικά Νευρωνικά δίκτυα στους ζωντανούς οργανισμούς είναι αρκετά περίπλοκες, αλλά και πολύ αναγκαίες στην

καθημερινή ζωή του ανθρώπου. Μερικές από αυτές είναι εργασίες ρουτίνας, και τις οποίες ο ανθρώπινος εγκέφαλος εκτελεί με ελάχιστη ή μηδαμινή προσπάθεια, όπως για παράδειγμα την αναγνώριση μιας εικόνας. Το ερώτημα που προκύπτει λοιπόν είναι: Μπορούν οι ηλεκτρονικοί υπολογιστές να κάνουν αυτά που κάνει το ανθρώπινο μυαλό; Μερικά από τα πιο απλά πράγματα, όπως η αναγνώριση φωνής ή εικόνας που το μυαλό κάνει πολύ εύκολα, οι υπολογιστές δεν μπορούν εύκολα να τα κάνουν με επιτυχία. Και βέβαια αυτό δεν οφείλεται στην έλλειψη ταχύτητας, καθ' όσον οι υπολογιστές είναι χιλιάδες φορές γρηγορότεροι από τον εγκέφαλο. Ο λόγος είναι ότι η δομή των υπολογιστών είναι πάρα πολύ διαφορετική από την δομή του εγκεφάλου. Με σχετικές έρευνες θα μπορούσαμε να κατασκευάσουμε έναν υπολογιστή που η δομή του να μοιάζει με αυτήν του εγκεφάλου, και έτσι να μπορέσουμε να πετύχουμε αυτό που θέλουμε.

Αυτό έχει οδηγήσει στο να γίνουν κάποιες πρώτες σκέψεις μήπως είναι δυνατόν να δημιουργηθούν κάποια αρχικά μοντέλα του νευρωνικού συστήματος του ανθρώπου, σε μια προσπάθεια να μιμηθούν τα χαρακτηριστικά του εγκεφάλου και να εκτελέσουν διάφορες εργασίες, όπως γίνεται και στα βιολογικά Νευρωνικά δίκτυα. Αυτά ονομάζονται τεχνητά Νευρωνικά δίκτυα (ΤΝΔ).

Πρόκειται για ένα αφηρημένο αλγοριθμικό κατασκεύασμα το οποίο ανήκει στον τομέα της υπολογιστικής νοημοσύνης και στόχος του είναι η επίλυση κάποιου υπολογιστικού προβλήματος. Τα δίκτυα αυτά εκπαιδεύονται με την εξάσκηση και την εμπειρία, όπως ακριβώς και οι άνθρωποι, αλλά διαφέρουν στο ότι δεν ακολουθούν κάποιους προκαθορισμένους κανόνες, που είναι χαρακτηριστικό των υπολογιστών. Η χρησιμότητα των ΤΝΔ είναι εμφανής καθώς πλέον μπορούν και πραγματοποιούνται διεργασίες που μέχρι σήμερα μόνο ο άνθρωπος μπορούσε να κάνει. Τα τελευταία τριάντα χρόνια λοιπόν, τα τεχνητά Νευρωνικά δίκτυα έχουν δει μεγάλη άνθηση και πρόοδο.

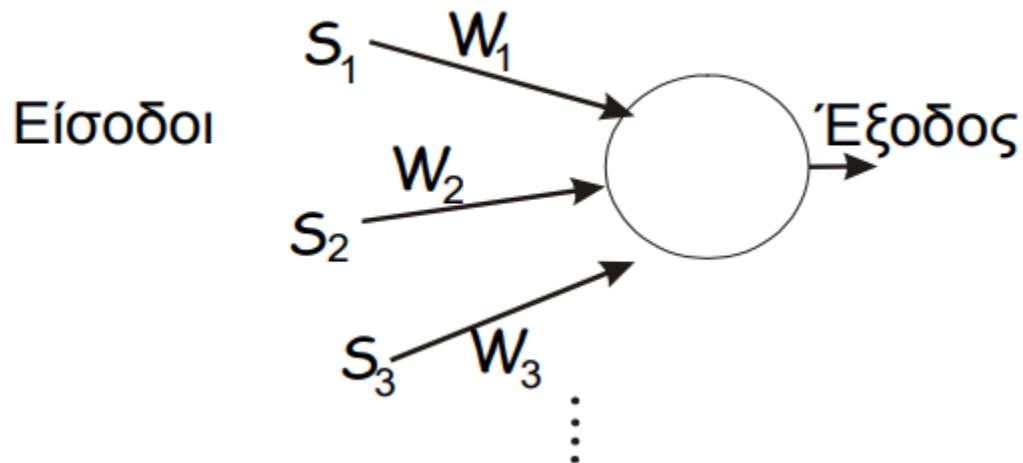
### 3.2 Η ΕΝΝΟΙΑ ΤΟΥ ΤΕΧΝΗΤΟΥ ΝΕΥΡΩΝΑ

Ένας τεχνητός νευρώνας είναι ένα σημείο σύνδεσης σε ένα τεχνητό νευρωνικό δίκτυο. Τα τεχνητά νευρωνικά δίκτυα, όπως και το βιολογικό νευρωνικό δίκτυο του ανθρώπινου σώματος, έχουν πολυεπίπεδη αρχιτεκτονική και κάθε κόμβος του δικτύου (σημείο σύνδεσης) έχει τη δυνατότητα να επεξεργάζεται δεδομένα την είσοδο του και να προωθεί δεδομένα στην έξοδο και σε άλλους κόμβους του δικτύου.

Για να γίνει πιο κατανοητή η δομή ενός τεχνητού νευρωνικού δικτύου, πρέπει αρχικά να μελετηθεί σε ατομικό επίπεδο. Το βασικό συστατικό από το οποίο δημιουργούνται τα μοντέλα αυτά, είναι ο τεχνητός νευρώνας. Μπορούμε να τον φανταστούμε ως μια συνάρτηση η οποία λαμβάνει ένα διάνυσμα  $x_i = (x_1, x_2, x_3, \dots, x_n)$  διάστασης  $n$  και το αντιστοιχεί σε ένα βαθμωτό στοιχείο, μέσω της εξής διαδικασίας :

1. Υπολογισμός του σταθμισμένου αθροίσματος  $w'x = \sum_{i=1}^n w_i x_i$  όπου  $w$ , τα προσαρμοζόμενα βάρη.
2. Στο άθροισμα αυτό προστίθεται και η τιμή της μεροληψίας του νευρώνα (bias).

3. Εφαρμογή μιας μη-γραμμικής συνάρτησης (συνάρτηση ενεργοποίησης), η οποία θα μας δώσει την έξοδο του νευρώνα.



Εικόνα 3.2: Μοντέλο τεχνητού νευρώνα [κύκλος] όπου,  $S_{1 \rightarrow 3}$  : είσοδοι και  $W_{1 \rightarrow 3}$  : βάρη

### 3.3 ΣΥΝΑΡΤΗΣΕΙΣ ΕΝΕΡΓΟΠΟΙΗΣΗΣ

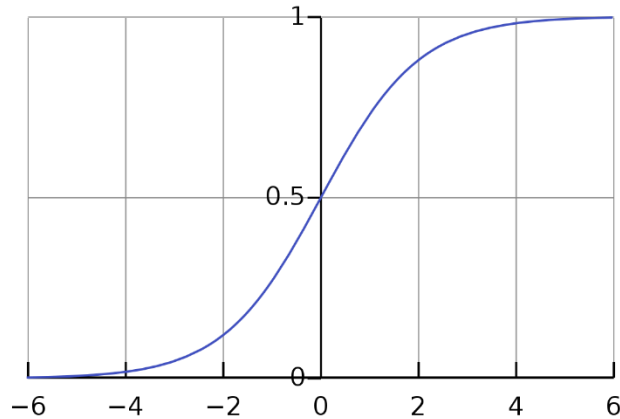
Οι συναρτήσεις ενεργοποίησης χρησιμοποιούνται για τον προσδιορισμό της εξόδου(output) του νευρωνικού δικτύου, όπως για παράδειγμα ναι ή όχι. Με άλλα λόγια χαρτογραφεί τις τιμές που προκύπτουν μεταξύ 0 έως 1 ή -1 έως 1 κ.λπ. ανάλογα με τη συνάρτηση.

Σε περίπτωση που δεν έχει ήδη γίνει σαφές, η ύπαρξη της συνάρτησης ενεργοποίησης είναι αναγκαία συνθήκη για τη μη-γραμμική απόκριση του συστήματος. Χωρίς την ύπαρξη της, η έξοδος του δικτύου θα ήταν απλώς ένας (σύνθετος) γραμμικός συνδυασμός. Για το λόγο αυτό αξίζει να αναφέρουμε μερικές από τις σημαντικότερες συναρτήσεις, καθώς και τη χρησιμότητα τους ανάλογα με τις μελέτες περίπτωσης.

- **Σιγμοειδής συνάρτηση (sigmoid)**

Η μαθηματική της έκφραση είναι,

$$\Phi(x) = \frac{1}{1 + e^{-x}}$$



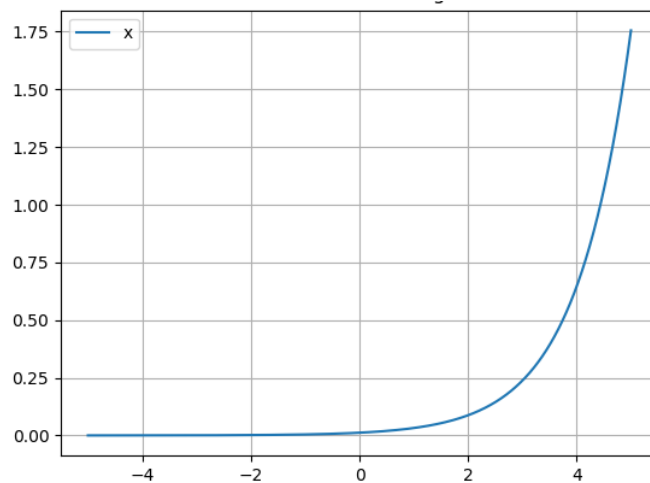
**Εικόνα 3.3:** Γράφημα σιγμοειδούς συνάρτησης.

και αποδεικνύεται βέλτιστη επιλογή σε προβλήματα δυαδικής ταξινόμησης, δηλαδή της κατηγοριοποίησης στοιχείων σε 2 διαφορετικές ομάδες, με βάση κάποια κοινά τους χαρακτηριστικά. Μια διαισθητική ερμηνεία της εξόδου της συνάρτησης, είναι ότι μας δίνει την πιθανότητα του στοιχείου να ανήκει στην αντίστοιχη κλάση (0 ή 1).

- **Softmax function :**

Η μαθηματική της έκφραση είναι,

$$\Phi : \mathbb{R}^n \rightarrow \mathbb{R}^n, \Phi(x) = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$$



**Εικόνα 3.4:** Γράφημα SoftMax

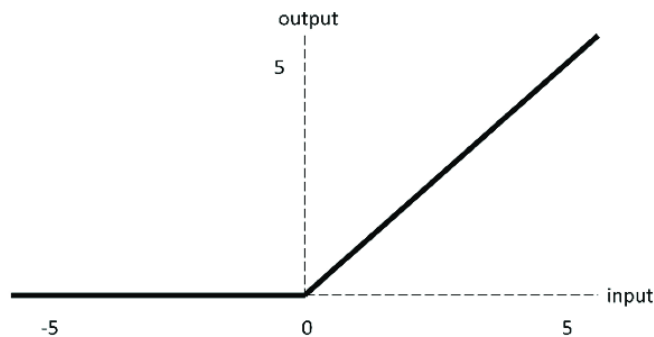
αποτελεί γενίκευση της λογιστικής συνάρτησης και αποδεικνύεται ιδιαίτερα χρήσιμη σε προβλήματα ταξινόμησης πολλών κλάσεων.



- **Rectified Linear Unit (ReLU):**

Η μαθηματική της έκφραση είναι,

$$\Phi(x) = x^+ = \max(x, 0)$$



Εικόνα 3.5: Γραφική απεικόνιση ReLU

μοιάζει κατάλληλη για προβλήματα παλινδρόμησης, στα οποία μάλιστα απαιτείται η τιμή της μεταβλητής απόκρισης να λαμβάνει μη-αρνητικές τιμές (όπως η τιμή μιας μετοχής).

### 3.4 PERCEPTRON

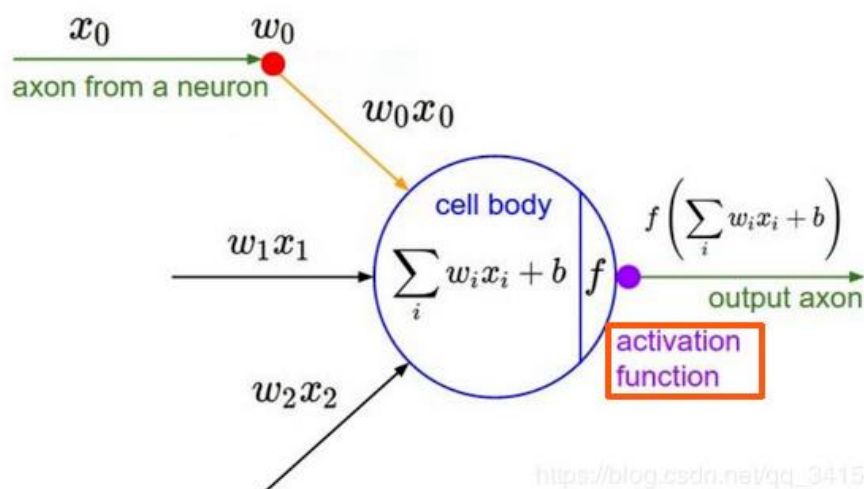
Το Perceptron στον τομέα της Μηχανικής Μάθησης, αποτελεί το πρώτο και το πιο απλοϊκό νευρωνικό δίκτυο. Είναι ένας αλγόριθμος μηχανικής μάθησης που μιμείται τον τρόπο λειτουργίας ενός νευρώνα του εγκεφάλου. Ονομάζεται επίσης νευρωνικό δίκτυο ενός στρώματος που απαρτίζεται από πολλούς νευρώνες εισόδου και μόνο έναν νευρώνα εξόδου. Η έξοδος αυτού του νευρωνικού δικτύου αποφασίζεται με βάση το αποτέλεσμα μιας μόνο γραμμικής συνάρτησης ενεργοποίησης. Στο perceptron, η πληροφορία διαδίδεται προς τα εμπρός ( forward propagation) και η έξοδος του συστήματος μπορεί να είναι μόνο δυαδική ( 0 ή 1 ).

Πολλοί Perceptrons μαζί αποτελούν ένα βαθύ νευρωνικό δίκτυο ( Deep Neural Network ) καθώς ένα ή περισσότερα perceptron είναι τοποθετημένα σε δύο ή περισσότερα στρώματα. Η είσοδος σε διαφορετικά perceptrons σε ένα συγκεκριμένο στρώμα θα τροφοδοτεί το επόμενο στρώμα συνδυάζοντάς τα με διαφορετικά βάρη (weights).

Η σύλληψη της ιδέας του perceptron και στην συνέχεια η υλοποίησή της άνοιξε νέα μονοπάτια στον τομέα των νευρωνικών δικτύων και ολοένα και μεγαλύτερος αριθμός επιστημόνων ξεκίνησαν την διατριβή τους φτάνοντας την τεχνητή νοημοσύνη σε νέα ύψη.

### 3.5 ΤΟΠΟΛΟΓΙΑ ΤΝΔ

Η δομική μονάδα του Τεχνητού Νευρωνικού Δικτύου είναι ο Τεχνητός Νευρώνας (ΤΝ), όπου εκεί γίνεται η επεξεργασία της πληροφορίας. Όλοι οι νευρώνες δέχονται και επεξεργάζονται τις πληροφορίες και δίνουν κάποιες τιμές εξόδου. Ο τεχνητός νευρώνας αποτελείται από έναν αθροιστή, η έξοδος του οποίου είναι το σταθμισμένο άθροισμα των εισόδων του. Το σταθμισμένο άθροισμα ( $\Sigma$ ) οδηγείται σε ένα στοιχείο ενεργοποίησης  $f(\cdot)$ , το οποίο είναι συνήθως, αλλά όχι πάντα, μη-γραμμικό.



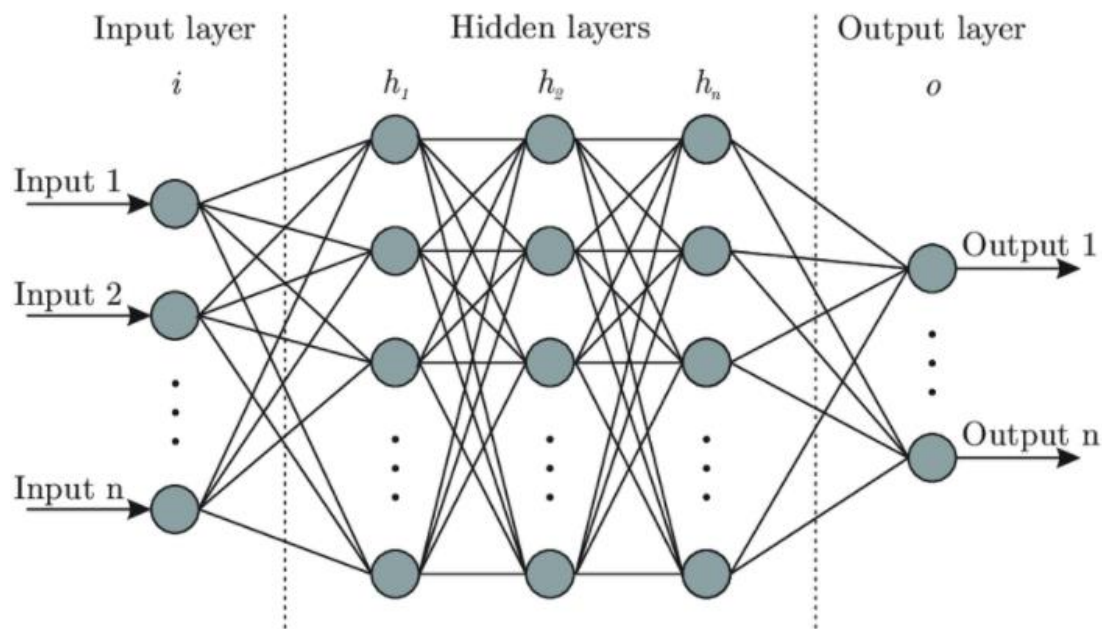
Εικόνα 3.6: Γραφική απεικόνιση ενός μη γραμμικού τυπικού τεχνητού νευρώνα

Το τεχνητό νευρωνικό δίκτυο είναι συνήθως κατασκευασμένο σε επίπεδα (layers). Ένα νευρωνικό δίκτυο μπορεί να περιέχει τα ακόλουθα 3 επίπεδα:

- Στρώμα Εισόδου (Input Layer): το στρώμα εισόδου έχει σκοπό το να λάβει στην είσοδο του τα αρχικά αριθμητικά δεδομένα, ή αλλιώς τις τιμές των επεξηγηματικών χαρακτηριστικών για κάθε κλάση. Συνήθως ο αριθμός των κόμβων εισόδου σε ένα επίπεδο εισόδου είναι ίσος με τον αριθμό των επεξηγηματικών μεταβλητών. Οι κόμβοι του επιπέδου εισαγωγής είναι παθητικοί, που σημαίνει ότι δεν αλλάζουν τα δεδομένα. Λαμβάνουν μόνο μία τιμή στην είσοδο τους και επαναλαμβάνουν την τιμή στις εξόδους τους.
- Κρυφό στρώμα (Hidden Layer): τα κρυφά στρώματα πραγματοποιούν δεδομένους μετασχηματισμούς στις τιμές εισόδου μέσα στο δίκτυο. Σε αυτό, τα εισερχόμενα τόξα που προέρχονται από άλλους κρυφούς κόμβους ή από κόμβους εισόδου συνδέονται σε κάθε νέο κόμβο. Στο κρυφό στρώμα, η πραγματική επεξεργασία γίνεται μέσω ενός συστήματος σταθμισμένων «συνδέσεων». Μπορεί να υπάρχουν ένα ή περισσότερα

κρυφά στρώματα. Οι τιμές που εισέρχονται σε έναν κρυφό κόμβο πολλαπλασιάζονται με τα βάρη, ένα σύνολο προκαθορισμένων αριθμών που είναι αποθηκευμένοι στο πρόγραμμα. Οι σταθμισμένες εισοδοί στη συνέχεια προστίθενται για να προκύψει ένας ενιαίος αριθμός.

- Στρώμα εξόδου (Output Layer): τα κρυφά στρώματα συνδέονται στη συνέχεια με ένα «στρώμα εξόδου». Το στρώμα εξόδου λαμβάνει συνδέσεις από τα κρυφά στρώματα ή από το στρώμα εισόδου. Επιστρέφει μια τιμή εξόδου που αντιστοιχεί στην πρόβλεψη της μεταβλητής απόκρισης. Στα προβλήματα ταξινόμησης, υπάρχει συνήθως μόνο ένας κόμβος εξόδου. Οι κόμβοι του στρώματος εξόδου συνδυάζουν και αλλάζουν τα δεδομένα για να παράγουν τις τιμές εξόδου.



Εικόνα 3.7: Αρχιτεκτονική τεχνητού νευρωνικού δικτύου (ΤΝΔ)

Η ικανότητα του νευρωνικού δικτύου να παρέχει χρήσιμη επεξεργασία δεδομένων έγκειται στη σωστή αναπροσαρμογή των βαρών. Κάτι που διαφέρει από τη συμβατική επεξεργασία πληροφοριών.

Η δομή ενός νευρωνικού δικτύου αναφέρεται επίσης ως «αρχιτεκτονική» ή, τοπολογία. Η επιλογή της δομής καθορίζει τα αποτελέσματα που πρόκειται να προκύψουν και αποτελεί το πιο κρίσιμο μέρος της υλοποίησης ενός νευρωνικού δικτύου. Μπορεί να υπάρχουν περισσότεροι από έναν κόμβοι εξόδου. Σε αυτή την περίπτωση, το μοντέλο που θα προκύψει θα είναι μια γραμμική ή λογιστική παλινδρόμηση. Αυτό εξαρτάται από το αν η συνάρτηση μεταφοράς είναι γραμμική ή λογιστική. Τα βάρη του δικτύου είναι οι συντελεστές παλινδρόμησης.

Με την προσθήκη ενός ή περισσότερων κρυφών στρωμάτων μεταξύ των στρωμάτων εισόδου και εξόδου και των μονάδων σε αυτό το στρώμα αυξάνεται η προβλεπτική ικανότητα ενός νευρωνικού δικτύου. Αλλά ο αριθμός των κρυφών στρωμάτων πρέπει να είναι όσο το δυνατόν μικρότερος. Αυτό διασφαλίζει ότι το

νευρωνικό δίκτυο δεν αποθηκεύει όλες τις πληροφορίες από το σύνολο μάθησης αλλά μπορεί να τις γενικεύει για να αποφεύγεται η υπερπροσαρμογή (overfitting).

### 3.6 ΤΕΧΝΙΚΕΣ ΕΚΠΑΙΔΕΥΣΗΣ ANN

Ένα ΤΝΔ για να φτάσει σε σημείο να είναι ικανό να εξάγει λογικά και αρεστά αποτελέσματα πρέπει να περάσει από το στάδιο της εκπαίδευσης. Όταν το σύνολο των δεδομένων μας είναι υπέρογκο συνιστάται η εκπαίδευση να γίνεται σε στάδια, τα στάδια ονομάζονται εποχές (epochs). Σε κάθε εποχή περιλαμβάνεται ένα σταθερό ποσό των δεδομένων εκπαίδευσης. Έτσι το ΤΝΔ καταφέρνει να εκπαιδευτεί καλύτερα και να γίνει πιο αξιόπιστο.

#### 3.6.1 ΕΠΙΒΛΕΠΩΜΕΝΗ ΜΑΘΗΣΗ (SUPERVISED LEARNING)

Η πιο συνήθης μορφή εκμάθησης που συναντάται, είναι η επιβλεπόμενη μάθηση. Ο χαρακτηρισμός 'επιβλεπόμενη' έχει δοθεί, διότι η διαδικασία εκμάθησης ενός αλγορίθμου από το σύνολο δεδομένων, μπορεί να παρομοιαστεί με τον τρόπο διδασκαλίας ενός δασκάλου σε ένα μαθητή, ο οποίος επιβλέπει τη διαδικασία και γνωρίζει (εκ των προτέρων) τις σωστές απαντήσεις. Ο βασικός στόχος ενός μαθητευόμενου είναι να γενικεύει την εμπειρία του. Σε αυτό το πλαίσιο γενίκευση είναι η ικανότητα μιας μηχανής μάθησης να αποδίδει με ακρίβεια σε καινούριες, πρωτόγνωρες εργασίες, αφού πρώτα έχει εκπαιδευτεί σε ένα σύνολο δεδομένων εκπαίδευσης. Γενικά τα προς εκπαίδευση παραδείγματα προέρχονται από κάποια άγνωστη κατανομή πιθανότητας, η οποία θεωρείται αντιπροσωπευτική του χώρου των καταστάσεων, και η μηχανή πρέπει να κατασκευάσει ένα γενικό μοντέλο που θα επιτρέπει την παραγωγή προβλέψεων σε καινούριες καταστάσεις με επαρκή ακρίβεια. Ο αλγόριθμος κάνει επαναληπτικά προβλέψεις στα δεδομένα εκπαίδευσης και διορθώνεται από τον εκπαιδευτικό. Η εκμάθηση σταματά όταν επιτευχθεί ένα αποδεκτό επίπεδο απόδοσης, το οποίο υπολογίζεται από τον αλγόριθμο, μέσω κάποιας συνάρτησης σφάλματος (loss function).

Στη περίπτωση της επιβλεπόμενης μάθησης, τα δεδομένα με τα οποία θα γίνει η εκπαίδευση αποτελούνται από μεταβλητές εισόδου ( $X$ ), καθώς και τις αντίστοιχες μεταβλητές εξόδου ( $y$ ). Στόχος των αλγορίθμων είναι η βέλτιστη προσέγγιση της συνάρτησης που αντιστοιχεί από καινούρια παραδείγματα τις μεταβλητές εισόδου σε ένα αποτέλεσμα:

$$y = f(x) \quad (1)$$

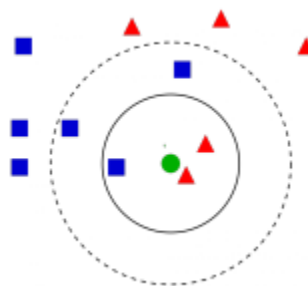
Η ανίχνευση ανεπιθύμητης αλληλογραφίας αποτελεί πρόβλημα επιβλεπόμενης μάθησης αφού κάθε είσοδος (μήνυμα) του συνόλου εκπαίδευσης έχει μια σωστή έξοδο ή αλλιώς ετικέτα (label). Τα προβλήματα που επιλύονται με μεθόδους επιβλεπόμενης μάθησης μπορούν να χωριστούν σε :

Προβλήματα Ταξινόμησης: Πρόκειται για προβλήματα αναγνώρισης προτύπων (pattern recognition), όπου ο αλγόριθμος αξιοποιώντας τα παραδείγματα εκπαίδευσης, προσπαθεί να εντοπίσει ομοιότητες σε αυτά, παράγοντας μια συνάρτηση, η οποία καλείται ταξινομητής, ο οποίος στοχεύει στη κατάταξη ενός νέου προτύπου (pattern) σε μία από τις γνωστές κλάσεις. Ανάλογα με το πλήθος των κλάσεων γίνεται η διάκριση σε δυαδική (binary), ή πολλαπλών κλάσεων (multiclass) ταξινόμηση.

Προβλήματα Παλινδρόμησης: Στα προβλήματα παλινδρόμησης δημιουργούνται μοντέλα πρόβλεψης αριθμητικών τιμών, δηλαδή η μεταβλητή εξόδου είναι συνεχής. Η πρόβλεψη της τιμής μιας μετοχής αποτελεί ένα τέτοιο πρόβλημα.

### 3.6.2 ΜΕΘΟΔΟΣ Κ-ΠΛΗΣΙΕΣΤΕΡΩΝ ΓΕΙΤΩΝΩΝ (k-NN)

Η μέθοδος αυτή είναι εξαιρετικά απλή, αλλά αρκετά διαδεδομένη και χρησιμοποιείται για το πρόβλημα της ταξινόμησης. Στη συγκεκριμένη μέθοδο, ο αλγόριθμος δεν 'εκπαιδεύεται', αλλά αποθηκεύει το σύνολο δεδομένων μαζί με τις ετικέτες (κλάσεις) της κάθε παρατήρησης. Η κατάταξη ενός νέου στιγμιότυπου γίνεται ανάλογα με το πόσο αυτό 'μοιάζει' με τα ήδη ταξινομημένα στιγμιότυπα. Το όνομά τους οφείλεται στο ότι η κατάταξη μιας νέας (άγνωστης) παρατήρησης σε κάποια κλάση, μελετώνται οι k-πλησιέστερες παρατηρήσεις, δηλαδή ταξινομείται στη κατηγορία που απέχει λιγότερο.



**Εικόνα 3.8 :** Επιλογή γειτόνων για την ταξινόμηση της άγνωστης πράσινης παρατήρησης.

### 3.6.3 ΜΗ ΕΠΙΒΛΕΠΩΜΕΝΗ ΜΑΘΗΣΗ (UNSUPERVISED LEARNING)

Στη προηγούμενη υποενότητα, το ενδιαφέρον επικεντρώθηκε στην πρόβλεψη των τιμών μιας ή περισσότερων μεταβλητών εξόδου για ένα δοσμένο σύνολο επεξηγηματικών μεταβλητών. Εάν υποθέσουμε ότι  $(x, Y)$  είναι τυχαίες μεταβλητές που αντιπροσωπεύονται από κάποια από κοινού κατανομή  $\mu_{XY}$ , τότε η επιβλεπόμενη μάθηση μπορεί να χαρακτηριστεί ως ένα πρόβλημα εκτίμησης της δεσμευμένης κατανομής  $\mu_Y | x$ . Η μη-επιβλεπόμενη μάθηση (ή 'μάθηση χωρίς δάσκαλο') από την άλλη, εξετάζει την περίπτωση όπου η κλάση στην οποία ανήκει το πρότυπο εκπαίδευσης δεν είναι γνωστή. Έτσι, κύρια μέριμνα μας τώρα είναι η 'αποκάλυψη' του τρόπου με τον οποίο τα πρότυπα οργανώνονται σε λογικές ομάδες (clusters ή groups), πράγμα το οποίο θα μας επιτρέψει να ανακαλύψουμε χρήσιμες ομοιότητες και διαφορές μεταξύ των προτύπων και να εξάγουμε χρήσιμα συμπεράσματα σχετικά με αυτά. Πιο συγκεκριμένα, έστω ένα σύνολο  $N$  παρατηρήσεων  $(x_1, x_2, \dots, x_n)$  ενός τυχαίου διανύσματος  $X$ , με κατανομή  $\mu_X$ . Ο στόχος είναι να συναγάγουμε άμεσα τις ιδιότητες αυτής της  $a$ -priori κατανομής χωρίς τη βοήθεια επόπτη ή δασκάλου που παρέχει σωστές απαντήσεις ή βαθμό σφάλματος για κάθε παρατήρηση. Αυτή είναι και η κύρια διαφορά των δύο τύπων μάθησης, από μαθηματικής απόψεως. Στη περίπτωση της μη-επιβλεπόμενης μάθησης, η διάσταση του  $X$  είναι μερικές φορές πολύ υψηλότερη από ό,τι στην επιβλεπόμενη, και τα προβλήματα που προκύπτουν είναι συχνά πιο περίπλοκα από απλές εκτιμήσεις παραμέτρων.

### 3.6.4 ΕΝΙΣΧΥΤΙΚΗ ΜΑΘΗΣΗ (REINFORCED LEARNING)

Η τρίτη βασική κατηγορία, είναι αυτή της ενισχυτικής μάθησης και πρόκειται για μια τεχνική μάθησης που επιτυγχάνεται μέσω της αλληλεπίδρασης με σκοπό την μεγιστοποίηση μιας συνάρτησης ανταμοιβής. Στην κατηγορία αυτή μελετάται πως τα συστήματα (γνωστά ως "ευφυείς πράκτορες") εκτελούν ενέργειες μέσα σε ένα περιβάλλον, μαθαίνοντας τη βέλτιστη στρατηγική που μεγιστοποιεί αυτή τη συνάρτηση.

Η ενισχυτική μάθηση συνδέεται και μελετάται από πολλούς κλάδους των μαθηματικών όπως, η θεωρία παιγνίων, επιχειρησιακή έρευνα, θεωρία ελέγχου και στατιστική. Ένα τέτοιο σύστημα μάθησης μοντελοποιείται ως εξής:

1. Ένα σύνολο καταστάσεων του περιβάλλοντος και πρακτόρων.
2. Ένα σύνολο ενεργειών του πράκτορα
3. Την πιθανότητα μετάβασης από τη κατάσταση σε μία άλλη, μέσω κάποιας ενέργειας.
4. Την ανταμοιβή που προστίθεται στη συνάρτηση ανταμοιβής, μέσω της παραπάνω ενέργειας.

## 3.7 ΜΕΤΡΑ ΑΞΙΟΛΟΓΗΣΗΣ

### 3.7.1 LOSS

Η απώλεια ή Loss, υπολογίζεται στην εκπαίδευση και την αξιολόγηση του μοντέλου και η ερμηνεία της βασίζεται στο πόσο καλά τα πάει το μοντέλο σε αυτές τις δύο περιπτώσεις. Είναι ουσιαστικά το άθροισμα των σφαλμάτων που γίνονται σε κάθε επανάληψη εκπαίδευσης (training) ή αξιολόγησης (validation). Η τιμή απώλειας υποδηλώνει πόσο άσχημα ή καλά συμπεριφέρεται ένα μοντέλο μετά από κάθε επανάληψη βελτιστοποίησης.

### 3.7.2 ACCURACY

Η ακρίβεια ενός μοντέλου προσδιορίζεται συνήθως αφού μαθευτούν (training accuracy) και διορθωθούν οι παράμετροι του μοντέλου. Στη συνέχεια, τα δείγματα από το σετ αξιολόγησης τροφοδοτούνται στο μοντέλο και καταγράφεται ο αριθμός των λαθών που κάνει το μοντέλο, μετά από σύγκριση με τους πραγματικούς στόχους. Στη συνέχεια υπολογίζεται το ποσοστό λανθασμένων προβλέψεων (validation accuracy). Ερμηνεύει λοιπόν το πόσο καλή είναι η πρόβλεπτική ικανότητα του μοντέλου σε σύγκριση με τα πραγματικά δεδομένα.

## 3.8 ΚΥΡΙΕΣ ΠΡΟΚΛΗΣΕΙΣ

### 3.8.1 OVDERFITTING / UNDERFITTING

Εστιάζοντας τη προσοχή στη κλάση της επιβλεπόμενης μάθησης, σκοπός κάθε αλγορίθμου είναι να εκτιμήσει όσο το δυνατόν καλύτερα την αντιστοίχιση (Συνάρτηση) των μεταβλητών εισόδου ( $X$ ), στις μεταβλητές εξόδου ( $y$ ). Ας υποθέσουμε ότι μετά το πέρας της εκπαίδευσης σε κάποια διαθέσιμα δεδομένα από ένα μοντέλο ΤΝΔ καταλήξαμε σε μια εκτίμηση ( $A$ ). Το αναμενόμενο σφάλμα πρόβλεψης (γνωστό και ως σφάλμα γενίκευσης) μπορεί να αποσυντεθεί σε 3 διαφορετικά μέρη ως εξής:

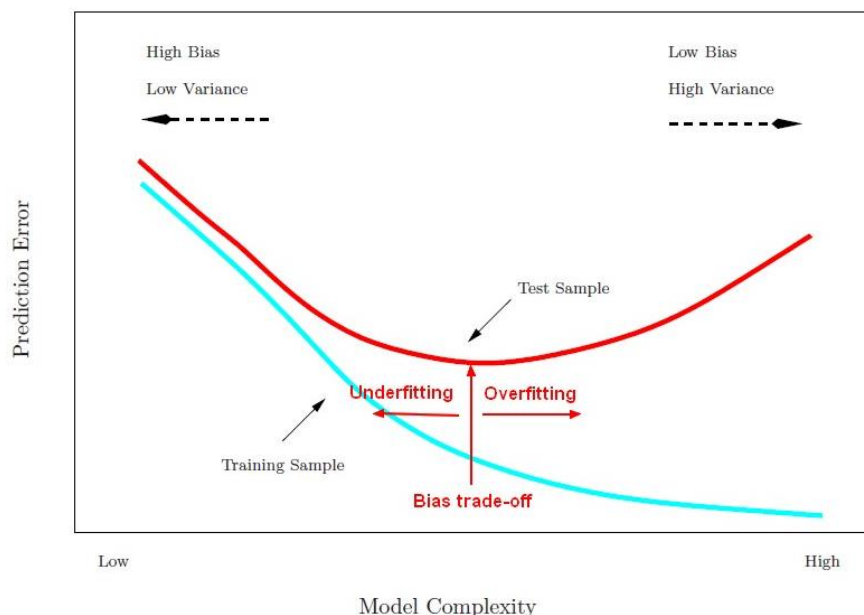
Σφάλμα λόγω μεροληψίας: Η μεροληψία (bias) ως μέτρο μας δείχνει το σφάλμα στην εκτίμηση, λόγω περιορισμών και υποθέσεων που υπάρχουν στο αλγόριθμο εκμάθησης. Για παράδειγμα εάν σε ένα σχετικά σύνθετο πρόβλημα, θέσουμε

υπεραπλουστευμένες παραδοχές, τότε εισάγουμε υψηλή μεροληψία, με αποτέλεσμα το μοντέλο να αγνοεί σημαντικές συσχετίσεις μεταξύ μεταβλητών εισόδου-εξόδου. Το φαινόμενο αυτό είναι γνωστό με τον όρο υποπροσαρμογή (underfitting).

Σφάλμα λόγω διακύμανσης: Αντίθετα, η διακύμανση ως μέτρο, μας υποδεικνύει τη μεταβολή του μοντέλου (των παραμέτρων), εάν εκπαιδευτεί (προσαρμοστεί) σε διαφορετικό σύνολο δεδομένων. Μοντέλα με υψηλή διακύμανση, επηρεάζονται σημαντικά από τα διάφορα χαρακτηριστικά και τις ιδιομορφίες που μπορεί να υπάρχουν στα δεδομένα. Ένα μοντέλο που εξηγεί (υπερβολικά) καλά τα δεδομένα στα οποία προσαρμόστηκε αλλά δεν εξηγεί καλά καινούργια δεδομένα πάσχει από το φαινόμενο της υπερπροσαρμογής (overfitting).

Τυχαίο σφάλμα: Το τυχαίο σφάλμα λέγεται αλλιώς και αμείωτο διότι προκύπτει από άγνωστη πηγή, δεδομένα τα οποία δεν έχουν ληφθεί υπόψιν κατά τη μοντελοποίηση του προβλήματος κ.λπ., επομένως είναι αναπόφευκτο. Μας ενδιαφέρουν πιο πολύ τα πρώτα δύο.

Η σχέση μεροληψίας-διακύμανσης (bias-variance tradeoff) αποτελεί κεντρικό θέμα στην επιβλεπόμενη μάθηση. Ιδανικά, στόχος σε ένα πρόβλημα τέτοιου χαρακτήρα είναι η επιλογή ενός μοντέλου, το οποίο είναι ικανό να ανιχνεύσει τις συσχετίσεις στο σύνολο δεδομένων εκπαίδευσης, αλλά και να είναι ικανό να γενικεύει σε καινούριες παρατηρήσεις.



**Εικόνα 3.9:** Σφάλμα γενίκευσης συναρτήσεως της πολυπλοκότητας του μοντέλου ΤΝΔ

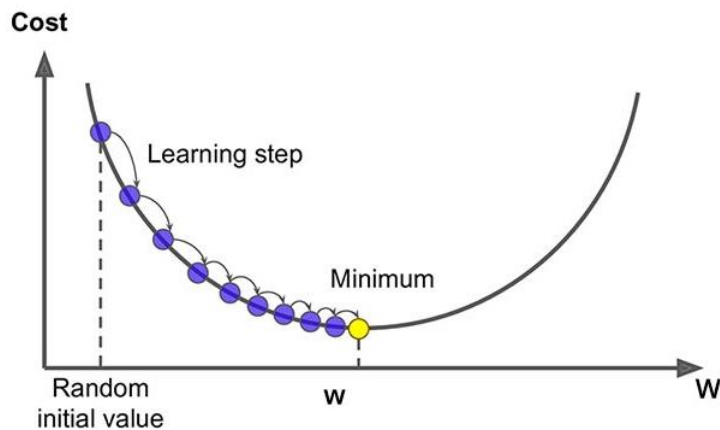


### 3.8.2 DATA MISSMATCH

Στη σημερινή εποχή της πληροφορίας που όλοι σχεδόν έχουν πρόσβαση στο διαδίκτυο, η αναζήτηση δεδομένων για τη προσαρμογή μοντέλων Μηχανικής Μάθησης, είναι μια εύκολη υπόθεση, στις περισσότερες περιπτώσεις. Παρόλα αυτά, κανείς δε μας εγγυάται ότι τα δεδομένα αυτά είναι αντιπροσωπευτικά του πληθυσμού, στον οποίο θα κληθεί το μοντέλο να δώσει προβλέψεις. Ας υποθέσουμε για παράδειγμα ότι καλούμαστε να δημιουργήσουμε μια εφαρμογή κινητού, η οποία διακρίνει και κατηγοριοποιεί διάφορα είδη λουλουδιών, τα οποία φωτογραφίζονται με τη κάμερα του κινητού. Είναι πολύ εύκολο να κατεβάσουμε από το διαδίκτυο χιλιάδες εικόνες από ποικίλα είδη λουλουδιών, οι οποίες όμως κατά πάσα πιθανότητα θα διαφέρουν σε μεγάλο βαθμό από αυτές που αποτυπώνει το κινητό. Το αποτέλεσμα είναι, οι προβλέψεις της εφαρμογής σε καινούρια δεδομένα να θα είναι ανακριβείς. Χωρίς τη γνώση αυτή για τα δεδομένα, δε μπορούμε να κρίνουμε και να αποδώσουμε την ευθύνη στο ίδιο το μοντέλο, δηλαδή στο φαινόμενο της υπερπροσαρμογής, ή στη ποιότητα και την αναντιστοιχία των δεδομένων (Data Mismatch). Πριν ασχοληθούμε λοιπόν με την επιλογή μοντέλου και τη προσαρμογή, επιβάλλεται η διερεύνηση των δεδομένων.

### 3.9 ΜΕΘΟΔΟΣ ΑΠΟΤΟΜΗΣ ΚΑΘΟΔΟΥ (GRADIENT DESCENT)

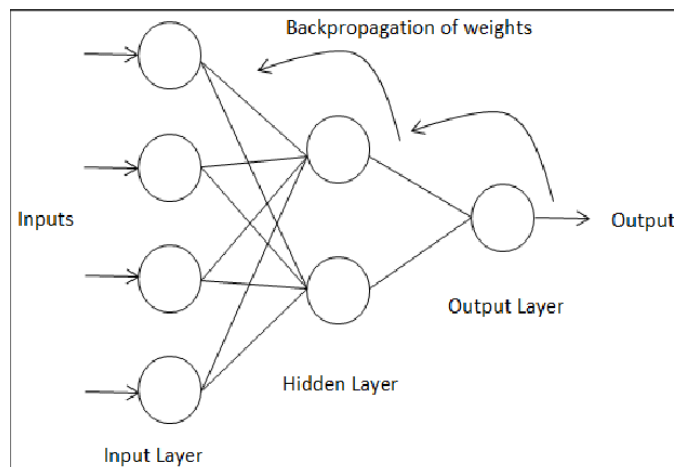
Η μέθοδος κλίσης είναι ένας αλγόριθμος βελτιστοποίησης που τροποποιεί τα εσωτερικά βάρη του νευρωνικού δικτύου για να ελαχιστοποιήσει την τιμή της συνάρτησης απώλειας. Στο σημείο της εκπαίδευσης, μετά από κάθε επανάληψη, ο αλγόριθμος βαθμωτής καθόδου προσπαθεί να μειώσει την τιμή της συνάρτησης απωλειών με τη διόρθωση των βαρών, μέχρι το σημείο όπου περαιτέρω διορθώσεις παράγουν ελάχιστη ή καμία αλλαγή στην τιμή της συνάρτησης απώλειας, που ονομάζεται επίσης σύγκλιση.



**Εικόνα 3.10:** Γραφική απεικόνιση της απότομης καθόδου

### 3.10 ΑΛΓΟΡΙΘΜΟΣ ΟΠΙΣΘΟΔΙΑΔΟΣΗΣ (BACKPROPAGATION)

Ο αλγόριθμος Οπισθοδιάδοσης ή αλλιώς οπισθοδρομική διάδοση των σφαλμάτων, είναι μια ευρέως χρησιμοποιούμενη μέθοδος για τον υπολογισμό παραγώγων μέσα σε βαθιά νευρωνικά δίκτυα τροφοδότησης. Η οπισθοδιάδοση αποτελεί σημαντικό μέρος πολλών αλγορίθμων επιβλεπόμενης μάθησης για την εκπαίδευση νευρωνικών δικτύων πρόσθιας τροφοδότησης.



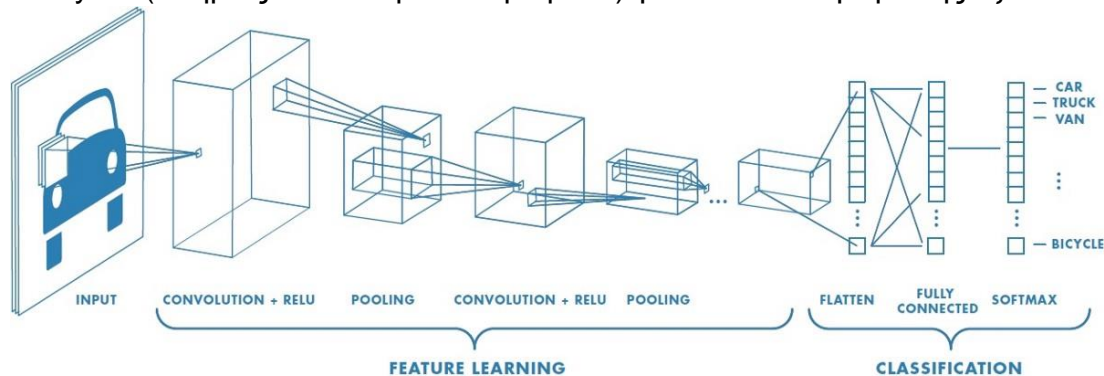
Εικόνα 3.11: Αλγόριθμος οπισθοδιάδοσης διαισθητικά

Όταν εκπαιδεύεται ένα νευρωνικό δίκτυο, υπολογίζεται μια συνάρτηση απώλειας (loss function), η οποία αντιπροσωπεύει το πόσο μακριά βρίσκονται οι προβλέψεις του δικτύου από την πραγματικότητα. Η οπισθοδιάδοση μας επιτρέπει να υπολογίσουμε την κλίση της συνάρτησης απώλειας σε συνάρτηση με κάθε ένα από τα βάρη του δικτύου. Αυτό επιτρέπει σε κάθε βάρος να αναπροσαρμόζεται ξεχωριστά, ώστε να μειώνεται σταδιακά η συνάρτηση απωλειών σε πολλές επαναλήψεις εκπαίδευσης.

### 3.11 ΣΥΝΕΛΙΚΤΙΚΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ (CNN)

Το Συνελικτικό Νευρωνικό Δίκτυο (CNN) είναι ένα πολυεπίπεδο νευρωνικό δίκτυο με ειδική αρχιτεκτονική για την ανίχνευση σύνθετων χαρακτηριστικών στα δεδομένα. Τα CNN έχουν χρησιμοποιηθεί για την αναγνώριση εικόνων, για την οπτική υποστήριξη ρομπότ και για αυτοκινούμενα οχήματα. Μόλις κατασκευαστεί ένα CNN, μπορεί να χρησιμοποιηθεί για την ταξινόμηση του περιεχομένου διαφορετικών εικόνων. Το μόνο που πρέπει να κάνουμε είναι να τροφοδοτήσουμε αυτές τις εικόνες στο μοντέλο. Ακριβώς όπως και τα ANN, τα CNN είναι εμπνευσμένα από τη λειτουργία του ανθρώπινου εγκεφάλου. Τα CNN είναι σε θέση να ταξινομήσουν εικόνες ανιχνεύοντας χαρακτηριστικά,

παρόμοια με τον τρόπο με τον οποίο ο ανθρώπινος εγκέφαλος ανιχνεύει χαρακτηριστικά για την αναγνώριση αντικειμένων. Όλα τα μοντέλα CNN μοιράζονται την ίδια αρχιτεκτονική (Εικόνα 3.7). Το μοντέλο λαμβάνει μια εικόνα στην είσοδό του, εφαρμόζει μια σειρά από φίλτρα στο συνελικτικό στρώμα (convolutional layer) τα δεδομένα με τη σειρά τους περνάνε σε κάποια στρώματα συγκέντρωσης (pooling layers) ακολουθούμενα από dense layers (πλήρως συνδεδεμένα στρώματα) για τον υπολογισμό της εξόδου.



Εικόνα 3.12: Αρχιτεκτονική CNN μοντέλου

Η συνέλιξη αξιοποιεί τρεις σημαντικές ιδέες που παρακίνησαν τους ερευνητές της όρασης υπολογιστών (computer vision): αραιή αλληλεπίδραση (sparse interaction), διαμοιρασμός παραμέτρων (parameter sharing) και ισοδύναμη αναπαράσταση (equivariant representation). Ας περιγράψουμε καθεμία από αυτές.

Τα στρώματα νευρωνικών δικτύων χρησιμοποιούν πολλαπλασιασμό πινάκων με έναν πίνακα παραμέτρων που περιγράφει την αλληλεπίδραση μεταξύ της μονάδας εισόδου και της μονάδας εξόδου. Αυτό σημαίνει ότι κάθε μονάδα εξόδου αλληλοεπιδρά με κάθε μονάδα εισόδου. Ωστόσο, τα νευρωνικά δίκτυα συνέλιξης έχουν αραιή αλληλεπίδραση (sparse interaction). Αυτό επιτυγχάνεται κάνοντας τον πίνακα μικρότερο από την είσοδο π.χ. μια εικόνα μπορεί να έχει εκατομμύρια ή χιλιάδες εικονοστοιχεία, αλλά κατά την επεξεργασία της με τη χρήση φίλτρων (kernel) μπορούμε να ανιχνεύσουμε σημαντικές πληροφορίες που είναι από δεκάδες ή εκατοντάδες εικονοστοιχεία. Αυτό σημαίνει ότι πρέπει να αποθηκεύσουμε λιγότερες παραμέτρους που όχι μόνο μειώνουν τις απαιτήσεις μνήμης του μοντέλου αλλά και βελτιώνουν τη στατιστική αποδοτικότητα του μοντέλου.

Εάν ο υπολογισμός ενός χαρακτηριστικού σε ένα χωρικό σημείο  $(x_1, y_1)$  είναι χρήσιμος, τότε θα πρέπει να είναι χρήσιμος και σε κάποιο άλλο χωρικό σημείο π.χ.  $(x_2, y_2)$ . Αυτό σημαίνει ότι για μία μόνο δισδιάστατη τομή, δηλαδή για τη δημιουργία ενός χάρτη ενεργοποίησης (feature map), οι νευρώνες περιορίζονται να χρησιμοποιούν το ίδιο σύνολο βαρών. Σε ένα παραδοσιακό νευρωνικό δίκτυο, κάθε στοιχείο του πίνακα βαρών χρησιμοποιείται μία φορά και στη συνέχεια δεν επανεξετάζεται ποτέ, ενώ το δίκτυο συνέλιξης έχει κοινές παραμέτρους, δηλαδή, για την παραγωγή εξόδου, τα βάρη που εφαρμόζονται σε μία είσοδο είναι τα ίδια με τα βάρη που εφαρμόζονται αλλού.

Λόγω του διαμοιρασμού των παραμέτρων, τα στρώματα του νευρωνικού δικτύου συνέλιξης θα έχουν την ιδιότητα της ισοδύναμης αναπαράστασης. Αν αλλάξουμε την είσοδο με έναν τρόπο, η έξοδος θα αλλάξει επίσης με τον ίδιο τρόπο.

## 4. ΧΡΗΣΗ ΤΕΧΝΗΤΩΝ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ ΓΙΑ ΑΝΑΓΝΩΡΙΣΗ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΕΞΑΡΤΗΜΑΤΩΝ

Τα νευρωνικά δίκτυα έχουν χρησιμοποιηθεί κατά καιρούς σε διάφορες εφαρμογές. Σημαντικές εφαρμογές τους παρατηρούνται στην πρόβλεψη χρηματιστηριακών μετοχών, την υπολογιστική όραση, την πρόγνωση σεισμικών φαινομένων. Στην παρούσα εργασία κατασκευάσαμε δύο τεχνητά νευρωνικά δίκτυα βαθιάς μάθησης για την αναγνώριση και ταξινόμηση διάφορων μηχανολογικών εξαρτημάτων που θα δέχονται ως είσοδο με τη μορφή εικόνων. Με χρήση κατάλληλης τεχνική εκπαίδευσης θα τις ταξινομήσει σε τέσσερις κλάσεις. Η βάση δεδομένων με την οποία τροφοδοτήσαμε τα NN αποτελείται από 7616 εικόνες (1904 για κάθε κλάση), διαστάσεων 224x224 pixels, για την διεξαγωγή των πειραμάτων χρησιμοποιήθηκε η γλώσσα προγραμματισμού Python 3 στη διαδικτυακή εφαρμογή ελεύθερου λογισμικού ανοιχτού κώδικα Jupiter Lab 3.0.5. Τέλος η επεξεργασία των δεδομένων έγινε σε κάρτα γραφικών επειδή είναι ταχύτερη από έναν επεξεργαστή σε μεγάλα σύνολα δεδομένων. Ο κύριος λόγος είναι η αποκλειστική μνήμη που έχει μια κάρτα γραφικών (VRAM) σε σχέση με έναν επεξεργαστή που χρησιμοποιεί τη μνήμη του συστήματος.

### 4.1 ΣΗΜΑΝΤΙΚΕΣ ΒΙΒΛΙΟΘΗΚΕΣ

Πριν ασχοληθούμε με την δημιουργία του μοντέλου των νευρωνικών δικτύων και την εκπαίδευσή τους είναι σημαντικό να κατανοήσουμε κάποιες από τις πιο χρήσιμες βιβλιοθήκες που χρησιμοποιήσαμε, καθώς αποτελούν κλειδιά για την κατασκευή μοντέλων ΤΝΔ.

#### 4.1.1 NUMPY

Η NumPy σημαίνει Numerical Python και είναι μία από τις πιο χρήσιμες επιστημονικές βιβλιοθήκες στον προγραμματισμό Python. Παρέχει υποστήριξη για μεγάλα πολυδιάστατα αντικείμενα συστοιχιών και διάφορα εργαλεία για την εργασία με αυτά. Διάφορες άλλες βιβλιοθήκες, όπως οι Pandas, Matplotlib και Scikit-learn, είναι

χτισμένες πάνω σε αυτή την καταπληκτική βιβλιοθήκη. Οι πίνακες είναι μια συλλογή στοιχείων/τιμών, που μπορεί να έχουν μία ή περισσότερες διαστάσεις. Ένας πίνακας με μία διάσταση ονομάζεται Vector (διάνυσμα), ενώ με δύο διαστάσεις ονομάζεται Matrix (πίνακας). Οι πίνακες NumPy ονομάζονται ndarray ή πίνακες N-διαστάσεων και αποθηκεύουν στοιχεία του ίδιου τύπου και μεγέθους. Είναι γνωστή για τις υψηλές επιδόσεις της και παρέχει αποτελεσματική αποθήκευση και λειτουργίες δεδομένων καθώς οι πίνακες αυξάνονται σε μέγεθος.

#### 4.1.2 TENSORFLOW

Το TensorFlow είναι μια βιβλιοθήκη λογισμικού ανοιχτού κώδικα που χρησιμοποιείται για αριθμητικούς υπολογισμούς με την χρήση Γράφους ροής δεδομένων (data flow graphs). Αναπτύχθηκε από την ομάδα Google Brain μέσα στα πλαίσια της έρευνας της Google, Machine Intelligence για την μηχανική μάθηση και την ανάλυση για τα «βαθιά νευρωνικά δίκτυα» (deep neural networks). Παρά την αρχική αιτία δημιουργίας του, το σύστημα πλέον είναι αρκετά γενικευμένο για να εφαρμοστεί σε μια πληθώρα άλλων χρήσεων.

Η μηχανική μάθηση γίνεται γρήγορα πολύπλοκη, και τα μοντέλα βαθιάς μάθησης αρκετά μεγάλα. Για πολλούς Γράφους μοντέλων χρειαζόμαστε κατανεμημένη εκπαίδευση για να μπορούμε να τους διαχειριστούμε σε λογικά και αποτελεσματικά χρονικά πλαίσια. Επίσης, υπάρχει η ανάγκη τα μοντέλα να εκπαιδεύονται με τέτοιο τρόπο που να μπορούν να χρησιμοποιηθούν σε πολλές διαφορετικές πλατφόρμες.

Με το TensorFlow γράφουμε κώδικα για να σχηματίσουμε έναν υπολογιστικό Γράφο και στην συνέχεια να τον εκτελέσουμε. Ο Γράφος είναι μια δομή δεδομένων η οποία περιγράφει πλήρως τους υπολογισμούς που θέλουμε να εκτελέσουμε, και έχει μια πληθώρα πλεονεκτημάτων:

- Είναι φορητός, εφόσον μπορεί είτε να εκτελεστεί άμεσα είτε να αποθηκευτεί για μεταγενέστερη χρήση, ενώ μπορεί να λειτουργήσει σε πολλές πλατφόρμες: επεξεργαστές, κάρτες γραφικών, TPUs (μονάδα επεξεργασίας τανιστή), κινητές και ενσωματωμένες συσκευές. Επίσης, μπορεί να χρησιμοποιηθεί στη γραμμή παραγωγής, χωρίς καμία απαίτηση χρήσης του κώδικα που έφτιαξε τον Γράφο, απαιτείται απλώς το κομμάτι εκτέλεσης για να χρησιμοποιηθεί.
- Είναι μεταβλητός και μπορεί να βελτιστοποιηθεί, αφού μπορεί να μεταβληθεί κατάλληλα για να «τρέξει» σε μια διαφορετική πλατφόρμα. Επίσης μπορούν να γίνουν αναβαθμίσεις μνήμης και υπολογιστικής δύναμης για να χρησιμοποιηθεί ο ίδιος Γράφος σε διαφορετικές πλατφόρμες και συνδυασμούς αυτών, με αποτέλεσμα να γίνεται πιο προσιτή η χρήση του.
- Τα υψηλού επιπέδου APIs του TensorFlow σε συνδυασμό με τους υπολογιστικούς Γράφους, δημιουργούν ένα πλούσιο και ευέλικτο περιβάλλον ανάπτυξης.

### 4.1.3 KERAS API

Το Keras είναι μια βιβλιοθήκη υψηλού επιπέδου που βασίζεται στο Theano ή το TensorFlow. Παρέχει ένα API για τη δημιουργία Νευρωνικών Δικτύων. Οι προγραμματιστές μπορούν να χρησιμοποιήσουν το Keras για να κατασκευάσουν γρήγορα νευρωνικά δίκτυα χωρίς να ανησυχούν για τις μαθηματικές πτυχές της άλγεβρας τανιστών, τις αριθμητικές τεχνικές και τις μεθόδους βελτιστοποίησης.

Η βασική ιδέα πίσω από την ανάπτυξη του Keras είναι η διευκόλυνση των πειραματισμών μέσω της γρήγορης δημιουργίας πρωτοτύπων. Η δυνατότητα να περάσουμε από μια ιδέα στο αποτέλεσμα με τη μικρότερη δυνατή καθυστέρηση. Αυτό προσφέρει ένα τεράστιο πλεονέκτημα τόσο στους επιστήμονες όσο και στους αρχάριους προγραμματιστές, επειδή μπορούν να ασχοληθούν άμεσα με τη βαθιά μάθηση χωρίς να λερώνουν τα χέρια τους με υπολογισμούς χαμηλού επιπέδου. Βασικά γνωρίσματα του Keras είναι τα εξής:

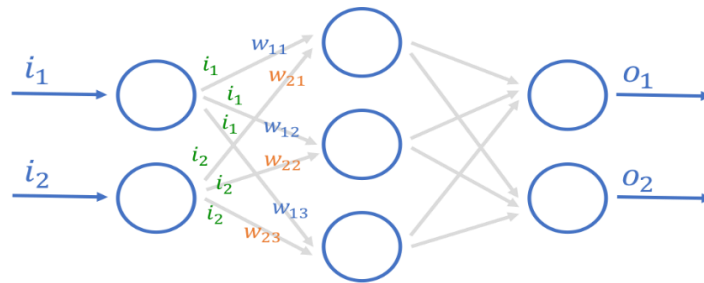
- Τρέχει ομαλά τόσο σε CPU όσο και σε GPU.
- Το Keras υποστηρίζει σχεδόν όλα τα μοντέλα ενός νευρωνικού δικτύου - πλήρως συνδεδεμένα, συνελκτικά, συγκεντρωτικά, αναδρομικά, ενσωματωμένα, κ.λπ. Επιπλέον, αυτά τα μοντέλα μπορούν να συνδυαστούν για τη δημιουργία πιο σύνθετων μοντέλων.
- Το Keras, όντας αρθρωτό στη φύση του, είναι απίστευτα εκφραστικό, ευέλικτο και κατάλληλο για καινοτόμες έρευνες.
- Το Keras είναι ένα πλαίσιο πλήρως βασισμένο στην Python, το οποίο καθιστά εύκολη την αποσφαλμάτωση και την εξερεύνηση του.

### 4.2 ΣΤΡΩΜΑΤΑ ΤΝΔ (LAYERS)

Κυρίαρχο χαρακτηριστικό σε κάθε βαθύ τεχνητό νευρωνικό δίκτυο είναι τα στρώματα ή αλλιώς layers. Για να γίνει πιο εύκολη η κατανόηση των ΤΝΔ είναι εύλογο να αναλύσουμε τα στρώματα που θα χρησιμοποιηθούν παρακάτω.

### 4.2.1 DENSE LAYER (πυκνό στρώμα)

Τα dense layer ή αλλιώς πλήρως συνδεδεμένα στρώματα είναι εκεί όπου πραγματικά γίνεται η ταξινόμηση. Σε οποιοδήποτε νευρωνικό δίκτυο, ένα πυκνό στρώμα είναι ένα στρώμα που συνδέεται στενά με το προηγούμενο, πράγμα που σημαίνει ότι σε ένα μοντέλο λαμβάνει έξοδο από κάθε νευρώνα του προηγούμενου στρώματος. Οι νευρώνες του dense layer εκτελούν πολλαπλασιασμό πίνακα διανύσματος. Ο πολλαπλασιασμός διανύσματος πίνακα είναι μια διαδικασία όπου το διάνυσμα γραμμής της εξόδου από τα προηγούμενα στρώματα ισούται με το διάνυσμα στήλης του πυκνού στρώματος. Ο γενικός κανόνας του πολλαπλασιασμού πίνακα διανύσματος είναι ότι το διάνυσμα γραμμής πρέπει να έχει τόσες στήλες όσες και το διάνυσμα στήλης.

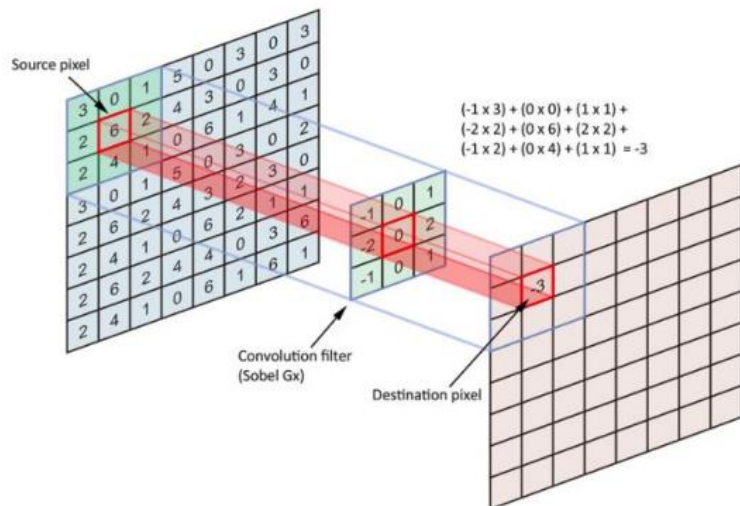


$$\begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \\ w_{13} & w_{23} \end{bmatrix} \cdot \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} (w_{11} \times i_1) + (w_{21} \times i_2) \\ (w_{12} \times i_1) + (w_{22} \times i_2) \\ (w_{13} \times i_1) + (w_{23} \times i_2) \end{bmatrix}$$

Εικόνα 4.1: Παράδειγμα πολλαπλασιασμού πινάκων

### 4.2.2 CONVOLUTIONAL LAYER (συνελικτικό στρώμα)

Ο όρος "συνέλιξη" αναφέρεται στον μαθηματικό συνδυασμό δύο συναρτήσεων για να σχηματίσουν μια τρίτη συνάρτηση. Όταν συμβαίνει αυτό, δύο σύνολα πληροφοριών συγχωνεύονται. Στο πλαίσιο των CNN, ένα convolutional layer (kernel ή φίλτρο) εφαρμόζεται στα δεδομένα εισόδου για να παραχθεί στη συνέχεια ένα feature map (χάρτης χαρακτηριστικών).

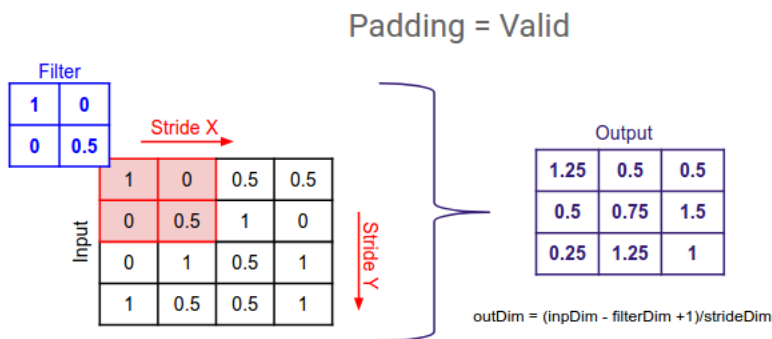


**Εικόνα 4.2:** Το συνελκτικό φίλτρο εφαρμόζεται σε στρώμα εισόδου και φαίνονται τα αποτελέσματα στο στρώμα εξόδου

Γίνεται πολλαπλασιασμός πινάκων μεταξύ ενός φίλτρου μεγέθους [3x3] (kernel) και μιας περιοχής μεγέθους [3x3] του πίνακα της εικόνας εισόδου. Τα στοιχεία του προκύπτοντος πίνακα αθροίζονται και το άθροισμα είναι η τιμή εξόδου στο feature map (χάρτη χαρακτηριστικών). Στη συνέχεια, το φίλτρο ολισθαίνει πάνω στον πίνακα εισόδου, επαναλαμβάνοντας τον πολλαπλασιασμό πινάκων με κάθε εναπομείναντα συνδυασμό μεγέθους [3x3] και ολοκληρώνεται έτσι ο χάρτης χαρακτηριστικών. Πολλαπλά φίλτρα χρησιμοποιούνται για μία είσοδο και οι προκύπτοντες χάρτες χαρακτηριστικών ενώνονται μαζί για την τελική έξοδο ενός συνελκτικού στρώματος.

Υπάρχουν δύο άλλες σημαντικές έννοιες στα convolutional layers το padding (γέμισμα) και τα strides (βήμα). Τα strides είναι ο αριθμός των εικονοστοιχείων που το φίλτρο συνέλιξης ολισθαίνει στην εικόνα εισόδου. Για παράδειγμα αν το (stride = 1) τότε θα έχουμε βηματισμό 1 pixel κάθε φορά που θα γίνεται ο πολλαπλασιασμός των πινάκων. Το padding είναι αυτό που χρησιμοποιείται όταν το φίλτρο δεν χωράει στον πίνακα εισόδου. Υπάρχουν δύο τύποι padding:

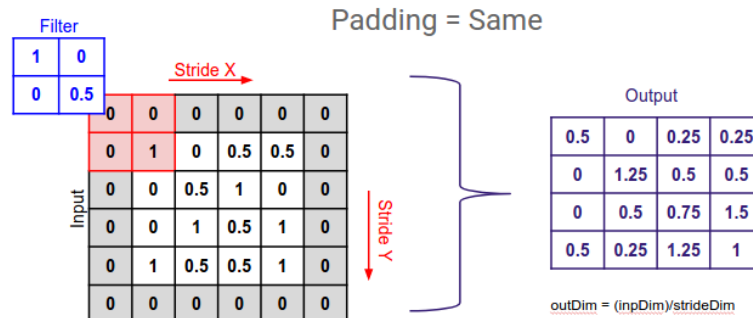
- το valid padding, όταν τα pixels του πίνακα εισόδου απορρίπτονται



**Εικόνα 4.3:** Valid padding όπου στοιχεία που περισσεύουν απορρίπτονται



•το zero ή same padding, όταν προστίθενται μηδενικά στα όρια του πίνακα έτσι ώστε το φίλτρο να ταιριάζει στον πίνακα εισόδου.

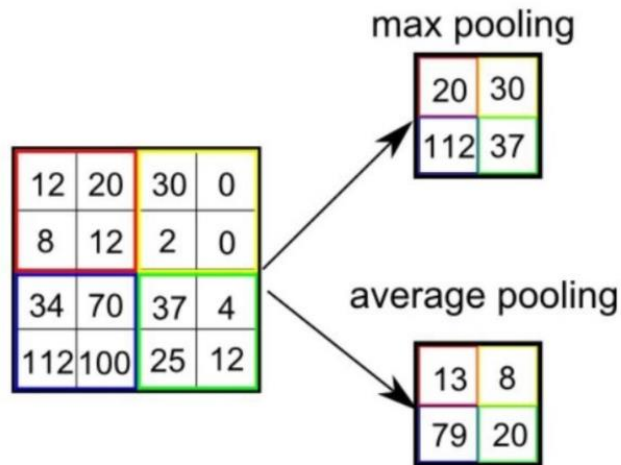


**Εικόνα 4.4:** Zero padding όπου συμπληρώνονται μηδενικά στα περιθώρια του πίνακα

### 4.2.3 POOLING LAYERS (στρώματα συγκέντρωσης)

Τα στρώματα συγκέντρωσης είναι υπεύθυνα για τη μείωση του μεγέθους των feature maps, συγκεκριμένα του ύψους και του πλάτους, διατηρώντας το βάθος. Η διαδικασία αυτή είναι χρήσιμη διότι μειώνει την απαιτούμενη υπολογιστική ισχύ για την επεξεργασία των δεδομένων, ενώ παράλληλα εξάγονται τα κυρίαρχα χαρακτηριστικά στα feature maps. Υπάρχουν δύο τύποι στρωμάτων συγκέντρωσης:

- Το **max pooling** το οποίο εξάγει τη μέγιστη τιμή των στοιχείων του χάρτη συγκέντρωσης στο τμήμα της εικόνας που καλύπτεται από το φίλτρο.
- Το **average pooling**, το οποίο αντίστοιχα επιστρέφει τη μέση τιμή των στοιχείων.



Εικόνα 4.5: Αναπαράσταση στρωμάτων συγκέντρωσης

### 4.3 ΚΑΤΑΣΚΕΥΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ (DATASET)

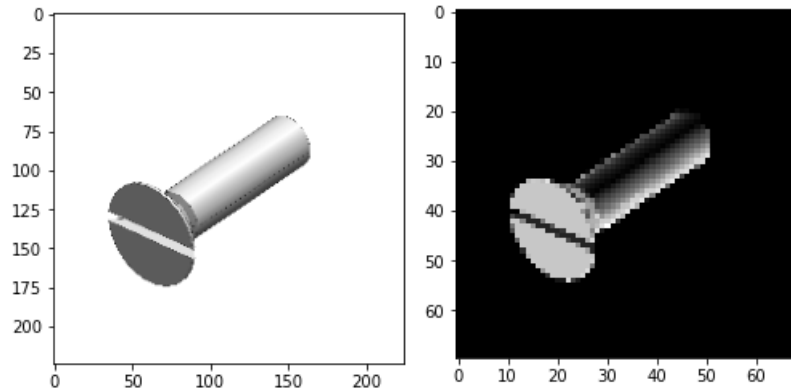
Η βάση δεδομένων που χρησιμοποιήθηκε στα νευρωνικά δίκτυα που επιλέξαμε αποτελείται από τρισδιάστατα μοντέλα CAD μηχανολογικών στοιχείων που πάρθηκε μέσω του ιστοτόπου GitHub και είναι κοινή για τα δύο δίκτυα.

Μια εικόνα αποτελείται από pixel. Στο μοντέλο RGB, κάθε pixel έχει τρία χρώματα στοιχεία, το κόκκινο, το πράσινο και το μπλε. Συνήθως κάθε στοιχείο μπορεί να κυμαίνεται από 0 (καθόλου χρώμα) έως 255 (πλήρης κορεσμός). Στην ψηφιακή επεξεργασία εικόνας, μια έγχρωμη εικόνα είναι ένας πίνακας pixel τριών επιπέδων, όπου κάθε στρώμα είναι ένας δισδιάστατος πίνακας που αντιπροσωπεύει το κόκκινο, το πράσινο ή το μπλε (Εικόνα 4.6).

			165	187	209	58	7
		14	125	233	201	98	159
253	144	120	251	41	147	204	
67	100	32	241	23	165	30	
209	118	124	27	59	201	79	
210	236	105	169	19	218	156	
35	178	199	197	4	14	218	
115	104	34	111	19	196		
32	69	231	203	74			

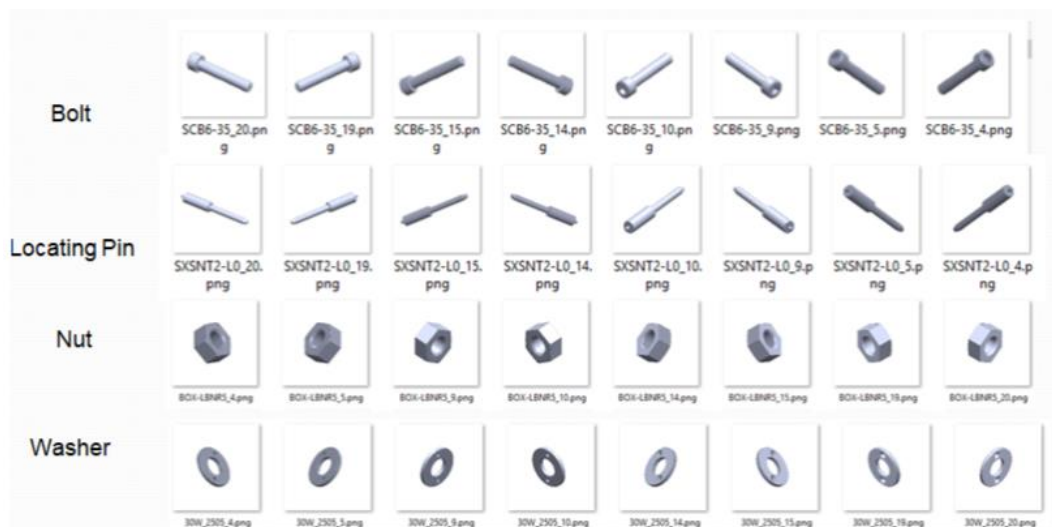
Εικόνα 4.6: Τρισδιάστατος πίνακας RGB. Κάθε στρώμα είναι ένας δισδιάστατος πίνακας τιμών pixel

Στην περίπτωση μας μια έγχρωμη εικόνα αποτελείται από  $224 \times 224 \times 3 = 150.528$  στοιχεία. Άμεσα μπορεί κάποιος να καταλάβει πως τα δεδομένα είναι υπέρογκα, συνεπώς για να μειώσουμε τον όγκο των δεδομένων που θα επεξεργαστούν τα NN's θα χρειαστεί να μικρύνουμε τις διαστάσεις (70 x 70) και να αλλάξουμε το χρώμα με εφαρμογή ενός φίλτρου σε ασπρόμαυρο (grayscale) των εικόνων (Εικόνα 4.7). Αν αλλάξουμε τις διαστάσεις των εικόνων δημιουργείτε νέος πίνακας με τα ίδια δεδομένα και διαφορετικό μέγεθος, έτσι δε θα επηρεάσουμε την διεξαγωγή των αποτελεσμάτων και θα διευκολύνουμε τη διαδικασία.



**Εικόνα 4.7:** Ενδεικτικό εξάρτημα, πριν και μετά τα φίλτρα από δεξιά προς τα αριστερά

Για τεχνικούς λόγους το σύνολο των δεδομένων (7616 εικόνες) απαρτίζουν έναν πίνακα [ X ] (Εικόνα 4.8). Ο πίνακας αυτός είναι τρισδιάστατος και εκτός από τις εικόνες περιέχει και τις διαστάσεις τους, ο πίνακας αυτός είναι της μορφής  $7616 \times 224 \times 224$ .



**Εικόνα 4.8:** Οπτικοποίηση των εξαρτημάτων που χρησιμοποιήθηκαν ως βάση δεδομένων

Ο δεύτερος πίνακας που θα δημιουργήσουμε είναι αυτός με τους μοναδικούς δείκτες (labels) που έχει η κάθε κλάση εξαρτημάτων (Εικόνα 4.9). Θα είναι ο πίνακας [ y ] με

μορφή 7616 x 1. (Π.χ. αν το έκτο στοιχείο της βάσης δεδομένων είναι παξιμάδι θα έχει δείκτη 2)

```
{0: 'bolt', 1: 'locatingpin', 2: 'nut', 3: 'washer'}
```

**Εικόνα 4.9:** Τοποθέτηση μηχανολογικών εξαρτημάτων με labels από 0 έως 3

Για τους λόγους της εκπαίδευσης χωρίσαμε τη βάση δεδομένων σε δύο κομμάτια. Το ένα κομμάτι θα είναι το κομμάτι της εκπαίδευσης που θα αποτελείται από το 80% της βάσης δεδομένων (training set), το άλλο κομμάτι είναι αυτό της αξιολόγησης που θα αποτελείται από το υπόλοιπο 20% (validation set) (Πίνακας 4.1). Ενσωματώνοντας επιπλέον ένα φίλτρο τυχαίας αναπαραγωγής δεδομένων για να αποφευχθεί η μεροληψία αποφεύγοντας το overfitting ή underfitting. Τα δεδομένα της εκπαίδευσης είναι αυτά από τα οποία το μοντέλο θα προσαρμόσει τα βάρη του κατάλληλα ώστε να "μάθει". Τα δεδομένα επαλήθευσης είναι κάτι καινούριο για το δίκτυο, δε τα έχει ξαναδεί, από αυτά θα υπολογιστεί η αποτελεσματικότητά του και θα πραγματοποιηθεί η αξιολόγησή του.

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                train_size=0.8,
                                                test_size=0.2,
                                                shuffle=True,
                                                random_state=1)
```

**Πίνακας 4.1:** Διαμοιρασμός του Dataset σε 80-20% για training και test.

#### 4.4 ΥΛΟΠΟΙΗΣΗ ΑΠΛΟΥ ΝΕΥΡΩΝΙΚΟΥ ΔΙΚΤΥΟΥ

Το ANN είναι ένα απλό πολυεπίπεδο νευρωνικό δίκτυο όπου κάθε νευρώνας του συνδέεται με όλους τους νευρώνες στο επόμενο επίπεδο. Στην κατασκευή του μοντέλου νευρωνικού δικτύου με τη βοήθεια του TensorFlow-keras Sequential έγινε η επιλογή των layers. Το δίκτυο αποτελείται μόνο από dense layers είναι ένα feedforward ANN αρχιτεκτονικής και περιέχει: ένα input layer, τρία hidden layers και ένα output layer (Πίνακας 4.2).

```
model = tf.keras.models.Sequential()

model.add(tf.keras.layers.Flatten()) # input layer

model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu)) # first hidden
layer
model.add(tf.keras.layers.Dense(256, activation=tf.nn.relu)) # second hidden
layer
model.add(tf.keras.layers.Dense(512, activation=tf.nn.relu)) # third hidden
layer
```

```

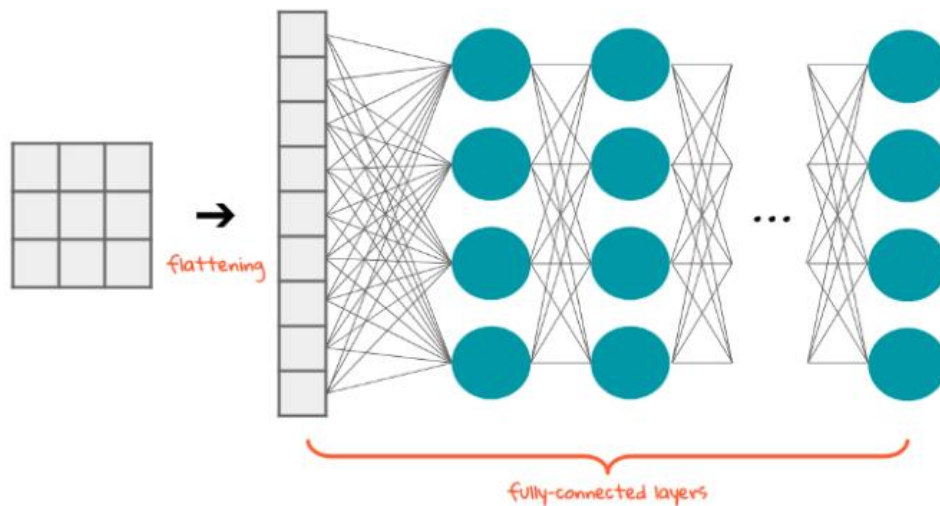
model.add(tf.keras.layers.Dense(4, activation=tf.nn.softmax)) #output layer

model.compile(optimizer = "Adam",
              loss = "sparse_categorical_crossentropy",
              metrics = ["accuracy"])

```

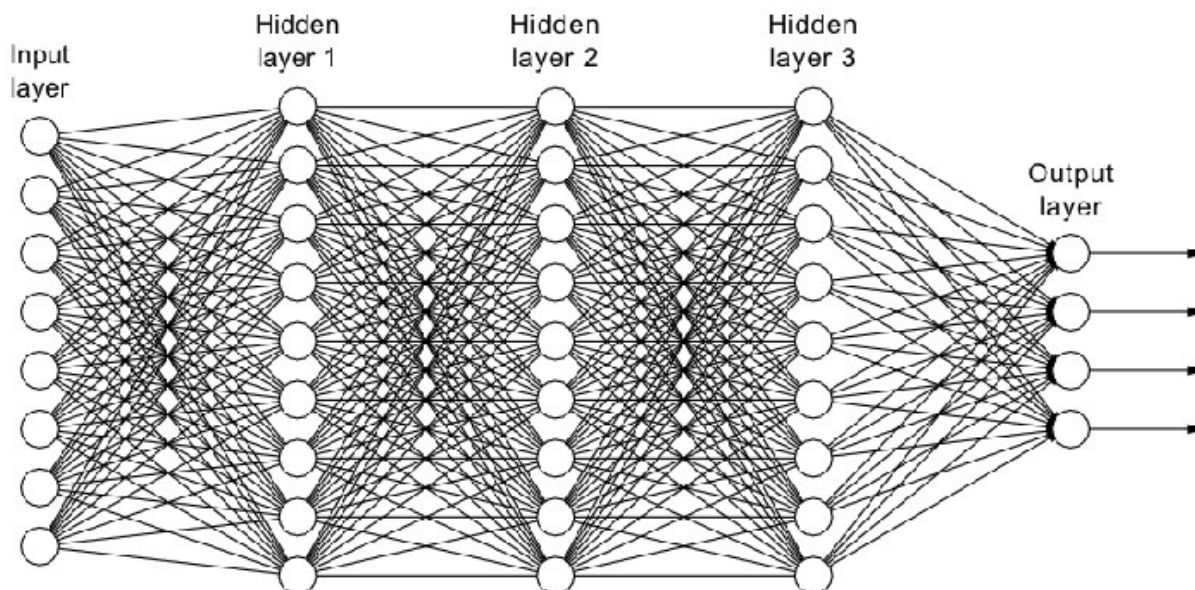
**Πίνακας 4.2:** Μοντέλο Νευρωνικού δικτύου με dense layers

Το input layer είναι ένα flatten-layer το οποίο λαμβάνει δεδομένα από τον τρισδιάστατο πίνακα [X] και τα μετατρέπει σε ένα μονοδιάστατο πίνακα, για να προχωρήσουν στα επόμενα layers (Εικόνα 4.10).



**Εικόνα 4.10:** Μετατροπή ενός τρισδιάστατου πίνακα σε μονοδιάστατο με το flatten-layer

Τα δεδομένα από το flatten layer περνούν στα τρία επόμενα hidden layers (Εικόνα 4.11) τα οποία είναι dense layers και αποτελούνται από 128 νευρώνες το πρώτο, 256 το δεύτερο και 512 το τρίτο στρώμα. Τέλος είναι το στρώμα εξόδου που περιέχει τέσσερις νευρώνες (έναν για κάθε κλάση) που είναι και αυτό ένα dense layer απόλυτα συνδεδεμένο με τα προηγούμενα.



**Εικόνα 4.11:** Νευρωνικό δίκτυο που αποτελείται από ένα input layer, τρία hidden layers και ένα output layer

Για κάθε κρυφό στρώμα σε αυτό το μοντέλο, επιλέγουμε τη RELU ως συνάρτηση ενεργοποίησης και την συνάρτηση Softmax ως συνάρτηση στο στρώμα εξόδου. Στο τέλος το μοντέλο συντίθεται με τον βελτιστοποιητή Adam και χρησιμοποιώντας την εντολή «sparse categorical cross-entropy» για τη κατασκευή της loss function (συνάρτηση απωλειών).

Το μοντέλο εκπαιδεύτηκε σε 20 epochs (εποχές) και batch size (δέσμες) 32. Κάθε epoch είναι ένα δείγμα δεδομένων που τροφοδοτείται προς τα εμπρός στο νευρωνικό δίκτυο. Τις περισσότερες φορές το σύνολο των δεδομένων δεν μπορεί να τροφοδοτηθεί στο νευρωνικό δίκτυο ταυτόχρονα με τη μία, οπότε πρέπει να χωριστεί σε δέσμες ή παρτίδες. Το μέγεθος της δέσμης υποδηλώνει το αριθμό των δειγμάτων εκπαίδευσης σε μία παρτίδα. Δηλαδή κάθε εποχή ή αλλιώς επανάληψη περιέχει 32 εικόνες από τις οποίες το μοντέλο εκπαιδεύεται.

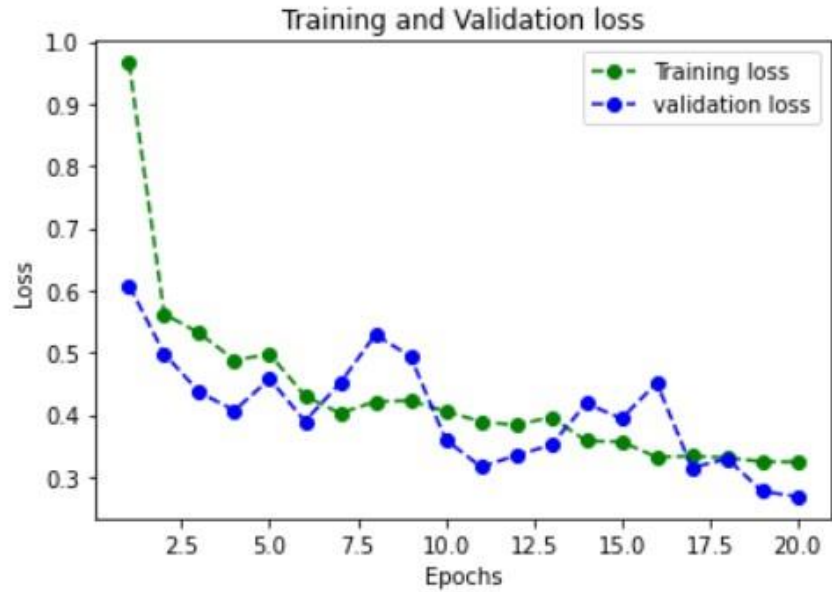
Στη συνέχεια ακολουθεί μια σύνοψη του μοντέλου που κατασκευάστηκε με σύμφωνα με τις προαναφερθέντες παραδοχές. (Εικόνα 4.12)

Layer (type)	Output Shape	Param #
flatten (Flatten)	(None, 4900)	0
dense (Dense)	(None, 128)	627328
dense_1 (Dense)	(None, 256)	33024
dense_2 (Dense)	(None, 512)	131584
dense_3 (Dense)	(None, 4)	2052
Total params: 793,988		
Trainable params: 793,988		
Non-trainable params: 0		

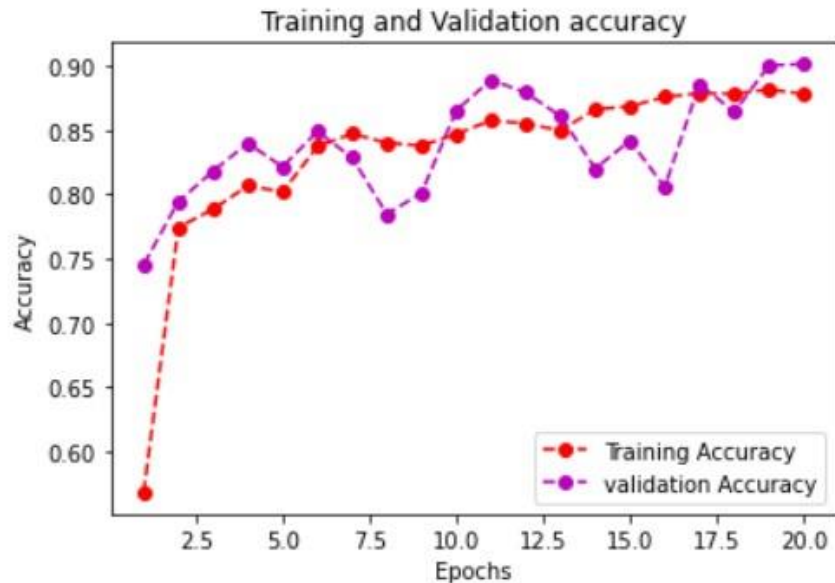
**Εικόνα 4.12:** Σύνοψη του μοντέλου ANN

#### 4.4.1 ΔΙΕΞΑΓΩΓΗ ΚΑΙ ΑΠΟΤΥΠΩΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΤΟΥ ANN

Με αναπροσαρμογή των βαρών του το νευρωνικό δίκτυο ελαχιστοποιεί το 'κόστος' (validation loss) και βελτιστοποιεί την ακρίβειά του. Όποτε ο αλγόριθμος κάνει λάθος προβλέψεις αναμένουμε αύξηση του validation loss. Μετά από 20 epochs μπορούμε να έχουμε πλέον μια οπτική του κόστους κατά τη διάρκεια αλλά και μετά το πέρας της εκπαίδευσης (Εικόνα 4.13), μπορούμε να εξάγουμε επιπλέον την ευστοχία του μοντέλου κατά την εκπαίδευση και την αξιολόγηση. (Εικόνα 4.14).



Εικόνα 4.13: Γράφημα Κόστους - Εποχών ANN



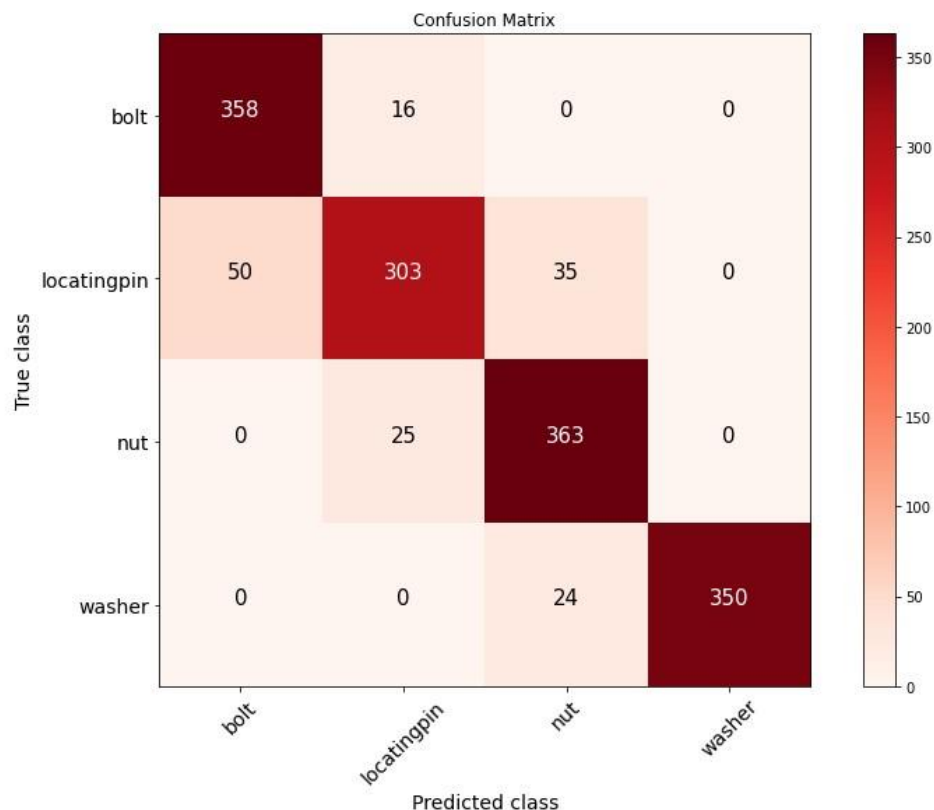
Εικόνα 4.14: Γράφημα Ακρίβειας - Εποχών ANN

Παρατηρούμε ότι το μοντέλο αυξάνει την ακρίβεια του με το πέρασμα των epochs και σε μερικά σημεία η ακρίβεια επαλήθευσης (validation accuracy) ξεπερνάει την ακρίβεια εκπαίδευσής του (training accuracy).

Αν θέλουμε μία ίσως πιο αναλυτική αποτύπωση των λαθών, μπορούμε ακολούθως να καταστρώσουμε έναν πίνακα γνωστό ως confusion matrix (CM) ή πίνακας σύγχυσης. Στον πίνακα αυτόν στον (x) άξονα αναφέρονται οι προβλέψεις κλάσεων (predicted classes) ενώ στον (y) άξονα οι πραγματικές κλάσεις (true classes). Η ακρίβεια αξιολόγησης από μόνη της μπορεί να είναι παραπλανητική εάν έχουμε άνισο αριθμό



παρατηρήσεων σε κάθε κλάση ή εάν έχουμε περισσότερες από δύο κλάσεις στο σύνολο δεδομένων.



**Εικόνα 4.15:** Confusion Matrix ANN, ύστερα της αξιολόγησης.

Απο το CM διακρίνουμε ποια κλάση αφορούν οι προβλέψεις, τα σωστά αποτελέσματα αναγράφονται διαγωνίως του κέντρου του πίνακα. Παρατηρούμε ότι, το μοντέλο εσφαλμένα πρόβλεψε 50 'bolt' (κοχλίες) ως 'locating pin' (πείρους), ενώ αντίθετα σωστά πρόβλεψε 358 'bolt', μετά το πέρας των 20 εποχών έχει κάνει συνολικά 150 λάθη.

#### 4.5 ΚΑΤΑΣΚΕΥΗ ΣΕΝΕΛΙΚΤΙΚΟΥ ΝΕΥΡΩΝΙΚΟΥ ΔΙΚΤΥΟΥ (CNN)

Το μοντέλο CNN (Convolutional Neural Network) είναι ένας τύπος μοντέλου βαθιάς μάθησης που διαθέτει ένα ή περισσότερα συνελκτικά στρώματα (conv2d layers). Στη Μηχανική Μάθηση χρησιμεύει συνηθέστερα στην αναγνώριση και επεξεργασία εικόνων, στην ταξινόμηση, την τμηματοποίηση και για άλλα αυτόματα συσχετισμένα δεδομένα. Κατά τη κατασκευή του CNN, το conv2d δέχεται ως είσοδο έναν πίνακα (εικόνα) και καθορίζει ως παράμετρο το μέγεθος των δισδιάστατων φίλτρων ή αλλιώς kernel που θα αλληλοεπιδράσουν με τον πίνακα εισόδου.

Στην περίπτωση μας, λαμβάνει στο στρώμα εισόδου του εικόνες στις οποίες εφαρμόζονται τρία φίλτρα συνέλιξης (convolution layers) με το καθένα να έχει διαστάσεις

[3x3] και μέγεθος φίλτρου 32, 64 και 128 pixel αντίστοιχα. Ανάμεσα σε κάθε συνελκτικό επίπεδο συμπεριλαμβάνεται και ένα maxpooling layer. Έπειτα εφαρμόζονται και δύο Dropout layers, ένα μετά το πρώτο convolution layer και ένα ανάμεσα από τα Dense layers με ρυθμό 0.2. Τα Dropout layers είναι ένας οικονομικός τρόπος επαναρύθμισης των βαρών δικτύου αποφεύγοντας το overfitting κατά την εκπαίδευση.

```
#Creation of a 3 layer CNN Sequential Model

model = Sequential()

model.add(Conv2D(32, (3,3), strides=(1, 1), padding='same', # first layer
input_shape=(IMG_SIZE, IMG_SIZE, 1)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

model.add(Conv2D(64, (3,3), strides=(1, 1), padding='same')) # second layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(128, (3,3), strides=(1, 1), padding='same')) # third layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())

model.add(Dense(256), Activation('relu'))
model.add(Dropout(0.2))

model.add(Dense(4)) # output layer
model.add(Activation('softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),metrics=['accuracy'])
```

**Πίνακας 4.3:** Μοντέλο συνελκτικού νευρωνικού δικτύου

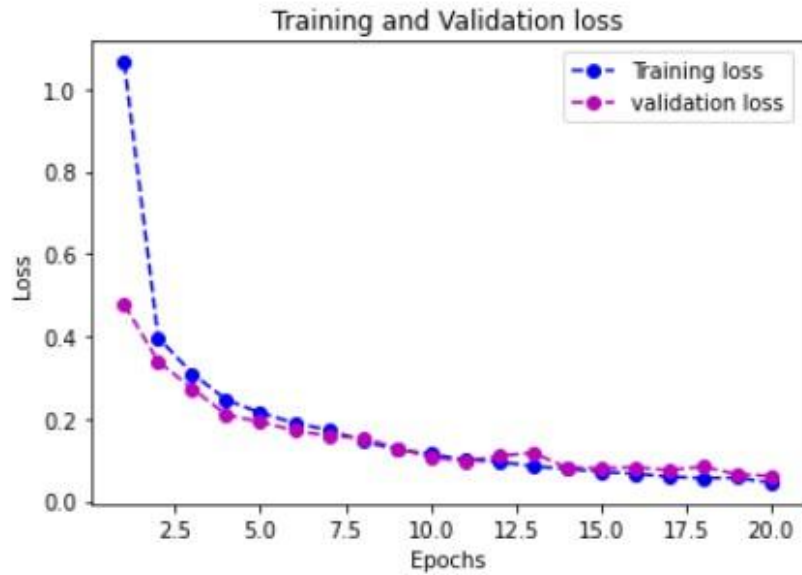
Μετά από τα επίπεδα συνέλιξης, βάζουμε ένα flatten layer στις εικόνες και προκύπτει ένας μονοδιάστατος πίνακας. Στη συνέχεια χρησιμοποιούμε ένα πυκνά συνδεδεμένο στρώμα με 256 νευρώνες και το στρώμα εξόδου με 4 νευρώνες ένας για κάθε κλάση. Για τη συνάρτηση ενεργοποίησης, επιλέγουμε τη ReLU στα κρυφά συνελκτικά στρώματα (Conv2D) και στο κρυφό πυκνό στρώμα (dense layer) ενώ για το στρώμα εξόδου τη Softmax στρώμα εξόδου. Ομοίως όπως και προηγουμένω το μοντέλο εκπαιδεύτηκε σε 20 epochs και batch size 32. Τέλος συντάσσουμε το μοντέλο με τον adam ως βελτιστοποιητή και την loss categorical cross-entropy για τη συνάρτηση απώλειας (loss function).

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 70, 70, 32)	320
activation (Activation)	(None, 70, 70, 32)	0
max_pooling2d (MaxPooling2D)	(None, 35, 35, 32)	0
dropout (Dropout)	(None, 35, 35, 32)	0
conv2d_1 (Conv2D)	(None, 35, 35, 64)	18496
activation_1 (Activation)	(None, 35, 35, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 17, 17, 64)	0
conv2d_2 (Conv2D)	(None, 17, 17, 128)	73856
activation_2 (Activation)	(None, 17, 17, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 128)	0
flatten (Flatten)	(None, 8192)	0
dense (Dense)	(None, 256)	2097408
dropout_1 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 4)	1028
activation_3 (Activation)	(None, 4)	0
Total params: 2,191,108		
Trainable params: 2,191,108		
Non-trainable params: 0		

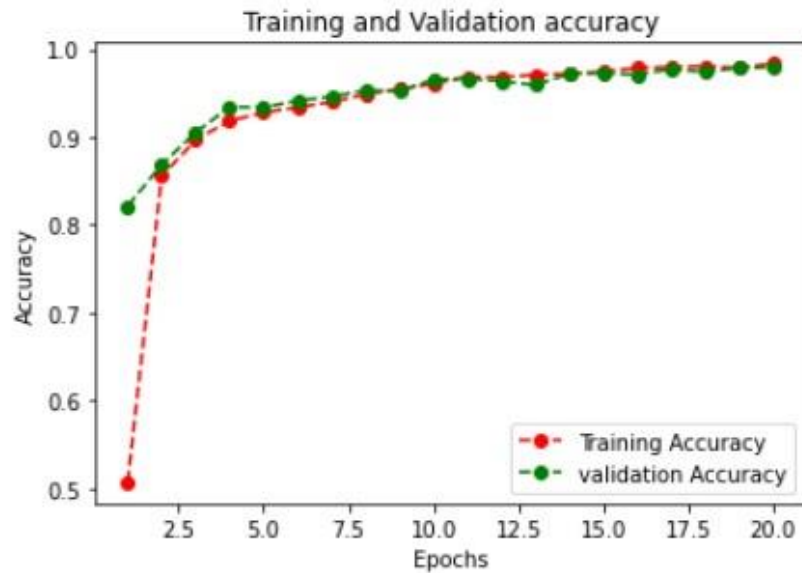
**Εικόνα 4.16:** Σύνοψη του μοντέλου CNN

#### 4.5.1 ΔΙΕΞΑΓΩΓΗ ΚΑΙ ΑΠΟΤΥΠΩΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ ΤΟΥ CNN

Με το πέρας των 20 εποχών η εκπαίδευση και αξιολόγηση ολοκληρώθηκαν. Δύο διαγράμματα μας αρκούν για να έχουμε πλήρη εικόνα του μοντέλου όπως παραπάνω, δηλαδή ένα γράφημα κόστους- εποχών (Εικόνα 4.17) ένα δεύτερο ακρίβειας - εποχών (4.18), επιπλέον παράγουμε και ένα Confusion Matrix για καλύτερη διαίσθηση των λαθών.

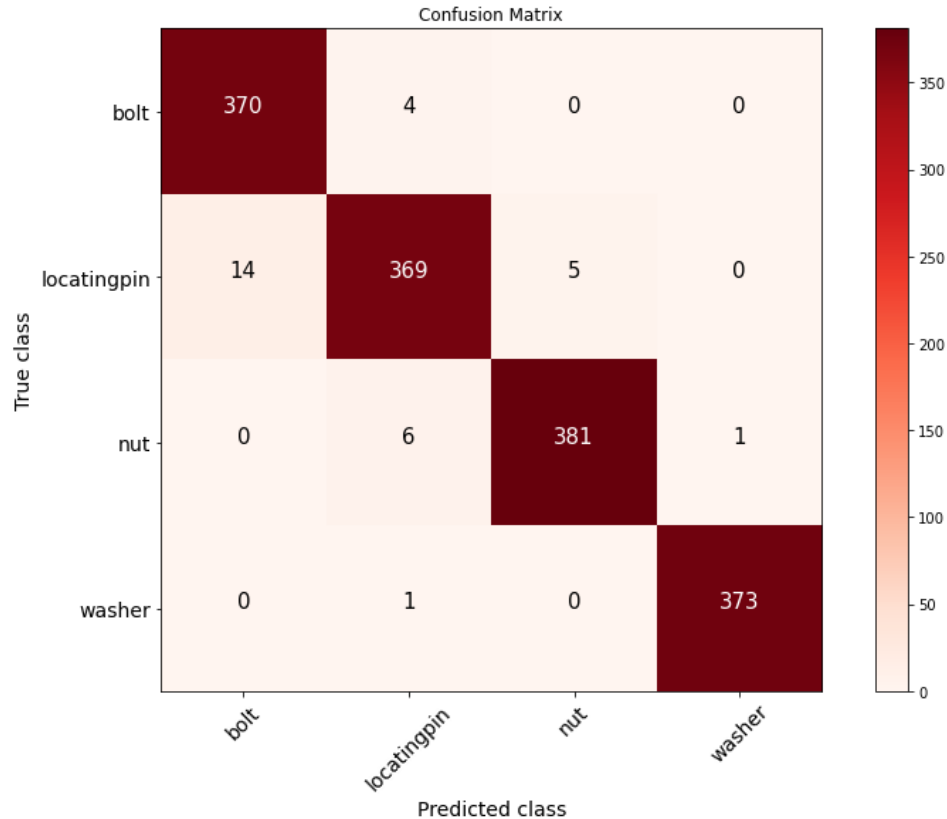


**Εικόνα 4.17:** Γράφημα Κόστους - Εποχών CNN



**Εικόνα 4.18:** Γράφημα Ακρίβειας - Εποχών CNN

Το επόμενο κριτήριο που θα λάβουμε υπόψιν για τη σύγκριση των αποτελεσμάτων μεταξύ των δύο νευρωνικών δικτύων είναι το confusion matrix στο οποίο παρατηρούμε 21% μείωση των λαθών συγκριτικά με το ANN (Εικόνα 4.19).



**Εικόνα 4.19:** Confusion Matrix CNN, ύστερα από την αξιολόγηση.

## 4.6 ΣΥΓΚΡΙΣΗ ΤΩΝ ΔΥΟ ΔΙΚΤΥΩΝ

Τώρα που έχουμε μία πλήρη εικόνα για τα δύο δίκτυα, ήρθε η στιγμή να βγάλουμε τα συμπεράσματά μας. Η σύγκρισή τους θα είναι πολύπλευρη, εξετάζοντάς τα από τρεις διαφορετικές πλευρές, πρώτα ως προς τον τρόπο λειτουργίας τους και στην συνέχεια ως προς τα αποτελέσματά τους.

### 4.6.1 ΣΥΓΚΡΙΣΗ ΩΣ ΠΡΟΣ ΤΟΝ ΤΡΟΠΟ ΛΕΙΤΟΥΡΓΙΑΣ

Το ANN χρησιμοποιεί βάρη και μια συνάρτηση ενεργοποίησης για το μεγαλύτερο μέρος της μεθόδου του. Ο καλύτερος τρόπος για να περιγράψει κανείς τον τρόπο λειτουργίας του ANN είναι ότι ανακατασκευάζει τεχνητά τον τρόπο λειτουργίας του νευρωνικού δικτύου ενός εγκεφάλου. Αφού κάνει κάτι λάθος, πηγαίνει πίσω και "αλλάζει" τον τρόπο σκέψης, όπως θα έκανε ένας άνθρωπος. Το ANN χρησιμοποιεί βάρη για να μάθει. Τα βάρη αλλάζουν μετά από κάθε επανάληψη μέσω του νευρώνα στο ANN. Το ANN επιστρέφει και αλλάζει τα βάρη ανάλογα με την ακρίβεια που υπολογίζεται από μια "συνάρτηση κόστους".

Συγκριτικά, δεν υπάρχει νευρώνας ή βάρη στο CNN. Το CNN αντ' αυτού εφαρμόζει πολλαπλά στρώματα στις εικόνες και χρησιμοποιεί φιλτράρισμα για να αναλύσει τις εισόδους εικόνας. Αυτά τα στρώματα είναι τα convolutional, pooling, dense. Ο σκοπός αυτών των στρωμάτων είναι να κατανοήσουν τα μοτίβα που μπορεί να "δει" το δίκτυο, να επεξεργαστούν την έξοδο των δεδομένων και να παρέχουν στην έξοδο ένα πίνακα n-διαστάσεων. Αυτός ο πίνακας χρησιμοποιείται για την παρατήρηση διακριτών χαρακτηριστικών και τη σύνδεσή τους με την παρεχόμενη εικόνα εισόδου. Στη συνέχεια δίνει στο χρήστη στην έξοδο του την ταξινόμηση των δεδομένων.

#### 4.6.2 ΣΥΓΚΡΙΣΗ ΤΩΝ ΕΙΣΟΔΩΝ ΤΟΥΣ

Το ANN επεξεργάζεται τις εισόδους με διαφορετικό τρόπο από το CNN. Ως αποτέλεσμα, το ANN αναφέρεται ως νευρωνικό δίκτυο Feed-Forward, επειδή οι εισόδοι επεξεργάζονται μόνο προς τα εμπρός. Λόγω της εξάρτησης από έγκυρες εισόδους δεδομένων, το ANN τείνει να είναι η λιγότερο δημοφιλής επιλογή κατά την ανάλυση εικόνων. Εν τω μεταξύ, το CNN λειτουργεί με συμβατό τρόπο για εικόνες ως δεδομένα εισόδου. Η χρήση φίλτρων σε μια εικόνα οδηγεί σε feature maps (χάρτες χαρακτηριστικών). Το CNN δεν επεξεργάζεται τα δεδομένα με κατεύθυνση προς τα εμπρός, αλλά αναφέρεται στα ίδια δεδομένα πολλές φορές κατά τη δημιουργία χαρτών.

#### 4.6.3 ΣΥΓΚΡΙΣΗ ΣΤΗΝ ΤΑΞΙΝΟΜΗΣΗ ΕΙΚΟΝΩΝ

Η αξιοπιστία του κάθε μοντέλου παρατηρούμε ότι είναι αρκετά υψηλή, πράγμα αναμενόμενο. Το CNN κατάφερε να αγγίξει 98% ευστοχία στις προβλέψεις του σε σχέση με το ANN που φτάνει έως 90%, εξάλλου αν με μια ματιά από τα διαγράμματα το CNN εκπέμπει μεγαλύτερη σιγουριά (Εικόνα 4.18). Αντίκτυπο της υψηλής ευστοχίας ήταν ο χρόνος ολοκλήρωσης των φάσεων εκπαίδευσης και αξιολόγησης, το ANN έκανε 21sec ενώ το CNN 25sec. Αυτό δικαιολογείται από τη διαφορά στον όγκο των εκπαιδευσιμών παραμέτρων, με το ANN έχει 793.988 παραμέτρους, την ίδια στιγμή που το CNN έχει 2.191.108. Από αυτά τα αποτελέσματα παρατηρούμε πως και τα δύο δίκτυα απέκτησαν φαινομενική προβλεπτική ικανότητα, με εμφανή νικητή όμως το CNN το οποίο κατάφερε να ξεπεράσει την ακρίβεια του ANN από την τέταρτη κιόλας εποχή.

Στην πραγματικότητα αν είχαμε εισάγει στα δίκτυα εικόνες διαφορετικές (π.χ. με διαφορετικό φόντο) θα παρατηρούσαμε μεγαλύτερη απόκλιση στην ακρίβεια. Αυτό θα μας οδηγούσε στην αναδιαμόρφωση του μοντέλου του ANN ώστε να μπορέσει να έχει καλύτερα αποτελέσματα, αυτό όμως θα σήμαινε και αύξηση των παραμέτρων του πιθανότατα περισσότερων από του CNN, με αποτέλεσμα την αύξηση του χρόνου εκπαίδευσης του δικτύου.

Εν κατακλείδι υπάρχουν συγκεκριμένες περιπτώσεις στις οποίες το ANN θα μπορούσε να προτιμηθεί έναντι του CNN και το αντίστροφο. Και τα δύο είναι μοναδικά στον τρόπο με τον οποίο λειτουργούν μαθηματικά, και αυτό τα κάνει να είναι καλύτερα

στην επίλυση συγκεκριμένων προβλημάτων. Σε γενικές γραμμές το CNN τείνει να είναι ένας πιο ισχυρός και ακριβής τρόπος επίλυσης προβλημάτων ταξινόμησης. Το ANN εξακολουθεί να είναι κυρίαρχο για προβλήματα όπου τα σύνολα δεδομένων είναι περιορισμένα και δεν περιέχουν εικόνες. Ωστόσο, λόγω της δυνατότητας του CNN να βλέπει τις εικόνες ως δεδομένα, είναι η πιο διαδεδομένη λύση για την υπολογιστική όραση και συναφή προβλήματα μηχανικής μάθησης.

## 5. ΤΕΛΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΕΡΑΙΤΕΡΩ ΜΕΛΕΤΗ

### 5.1 ΤΕΛΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΠΕΡΑΙΤΕΡΩ ΜΕΛΕΤΗ

Σε αυτή τη πτυχιακή εργασία προτείνεται μια μέθοδος ταξινόμησης μηχανολογικών εξαρτημάτων μέσω τεχνητών νευρωνικών δικτύων και διερευνώνται τεχνικές επιλογής βέλτιστης αρχιτεκτονικής. Τα αποτελέσματα δείχνουν ότι τα δίκτυα μπορούν να ταξινομήσουν τις εικόνες με αρκετά μεγάλη ακρίβεια σε πολύ μικρό χρονικό διάστημα, δεδομένων και των περιορισμένων υπολογιστικών πόρων. Όπως αναφέρθηκε και στο κεφάλαιο 3, η αποτελεσματικότητα ενός ΤΝΔ είναι ανάλογη της πολυπλοκότητας του, ενδεχομένως λοιπόν, με περισσότερο πειραματισμό να καταφθάναμε σε ακόμα καλύτερα αποτελέσματα.

Στη σημερινή εποχή οι βιομηχανίες αναζητούν νέες καινοτόμες λύσεις σε προβλήματα, για να επισπεύσουν την παραγωγή τους διατηρώντας τη βέλτιστη ποιότητα. Αυτό μπορεί να γίνει εφικτό μόνο με την ενσωμάτωση της τεχνητής νοημοσύνης στις γραμμές παραγωγής και ποιοτικού ελέγχου, όπως και είδαμε στο κεφάλαιο 4 με το CNN να αγγίζει 98% ευστοχία. Τα εντυπωσιακά αυτά αποτελέσματα δεν εντοπίζονται μόνο στον βιομηχανικό τομέα.

Σήμερα η τεχνητή νοημοσύνη έχει προχωρήσει πολύ και αποτελεί μέρος της ζωής μας. Στην πραγματικότητα βρίσκεται παντού επηρεάζοντας τις περισσότερες καθημερινές αποφάσεις μας. Χαρακτηριστικό παράδειγμα αποτελούν τα smartphones τα οποία χρησιμοποιώντας τη μηχανή πληροφοριών έχουν τη δυνατότητα να μας δώσουν οδηγίες ακόμα και για τη βέλτιστη διαδρομή που μπορούμε να ακολουθήσουμε ή την τιμή του κάθε προϊόντος που ενδεχομένως θέλουμε να αγοράσουμε. Ακόμα και η πρόβλεψη του καιρού γίνεται χάρη στην τεχνητή νοημοσύνη, καθώς και διάφορα είδη χειρουργείων πλέον γίνονται με χρήση έξυπνων robot, που επιτρέπουν στους γιατρούς να εκτελούν πολλούς τύπους πολύπλοκων διαδικασιών με μεγαλύτερη ακρίβεια, ευελιξία και έλεγχο από ό,τι είναι δυνατό με τις συμβατικές τεχνικές.

Τέλος προσθέτει ευφυΐα στα υπάρχοντα προϊόντα-μηχανήματα. Συνήθως η τεχνητή νοημοσύνη δεν πωλείται ως μεμονωμένη εφαρμογή. Πιθανότερο είναι οι ικανότητές της να βελτιώνουν τα προϊόντα που ήδη χρησιμοποιούνται όπως η πρόσθεση της ψηφιακής βοηθού Siri ως λειτουργία στα προϊόντα Apple νέας γενιάς. Στόχος της είναι η αυτοματοποίηση και η δημιουργία έξυπνων μηχανών που μπορούν να συνδυάσουν μεγάλες ποσότητες δεδομένων προκειμένου να βελτιώνουν πολλές διεργασίες στο σπίτι και στο χώρο εργασίας. Όσα έχουν ήδη επιτευχθεί μέχρι σήμερα θεωρούνται μόνο η κορυφή του παγόβουνου, καθώς ολοένα και περισσότερο εργατικό προσπαθεί να συμβάλλει στην εξέλιξη της τεχνητής νοημοσύνης.



## ΠΑΡΑΡΤΗΜΑ Ι (ANN)

```
import os
import glob
import numpy as np
import cv2
from skimage.io import imread
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sn
import tensorflow as tf
import keras
import random
from keras.utils import plot_model
from keras.layers import Dense, Activation, Flatten, Conv2D, MaxPooling2D
from keras.models import Sequential
from keras.utils import to_categorical
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
gpu = tf.config.list_physical_devices('GPU') #making sure the model runs on
gpu
tf.config.experimental.set_memory_growth(gpu[0], True)
print("you have", len(gpu), "GPU")
DATADIR = "C:\\Users\\TESTO\\Desktop\\ΣΧΟΛΗ\\ΠΤΥΧΙΑΚΗ\\dataset\\blnw-images-
224"
CATEGORIES = ['bolt', 'locatingpin', 'nut', 'washer']
os.chdir(DATADIR) #change the current directory to DATADIR
list_of_image = glob.glob("**/*.png") # Reads images
list_of_classes = glob.glob("**") # Reads classes
print("Total number of files " + str(len(list_of_image))) # Total data
print("Total Number of classes " + str(len(list_of_classes))) # Total Number
of classes
class_index = {}
for category in range(len(CATEGORIES)):
    class_index[category] = CATEGORIES[category]

class_index
y = []

for i, data in enumerate(list_of_classes, 0):
    list_of_images_in_folder = glob.glob(data+"/*.png")
    for j in list_of_images_in_folder:
        y.append(i)

y = np.asarray(y).reshape(-1,1) # Convert the list to numpy array
y.shape
image_list = []
IMG_SIZE = 70
for i,each in enumerate(list of image,1):
```

```

    im = imread(each, as_gray = True) # Convert images to gray and read as
an array
    new_array = cv2.resize(im, (IMG_SIZE, IMG_SIZE)) # Reshape images
    image_list.append(new_array) # Add to this list

X = np.asarray(image_list) # Convert the list to numpy array
X = X.reshape(-1, IMG_SIZE, IMG_SIZE, 1)
X.shape
plt.imshow(X[0].reshape(IMG_SIZE, IMG_SIZE), cmap = plt.cm.binary) #optical
example of a random image
plt.show() #optical example of a random image
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    train_size=0.8,
                                                    test_size=0.2,
                                                    shuffle=True,
                                                    random_state=1)

print("Shape of training data " + str(X_train.shape) )
print("Shape of testing data "+ str(X_test.shape) )
print("Shape of training label "+ str(y_train.shape) )
print("Shape of testing label "+ str(y_test.shape) )
model = tf.keras.models.Sequential()

model.add(tf.keras.layers.Flatten()) # input layer

model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu)) # first hidden
layer
model.add(tf.keras.layers.Dense(256, activation=tf.nn.relu)) # second hidden
layer
model.add(tf.keras.layers.Dense(512, activation=tf.nn.relu)) # third hidden
layer

model.add(tf.keras.layers.Dense(4, activation=tf.nn.softmax)) #output layer

model.compile(optimizer = "Adam",
              loss = "sparse_categorical_crossentropy",
              metrics = ["accuracy"])
no_epochs = 20
history = model.fit(X_train, y_train, epochs = no_epochs, batch_size=32,
                   validation_data = (X_test, y_test))

model.summary()
loss_train = history.history['loss']
loss_val = history.history['val_loss']
epochs = range(1, no_epochs+1)
plt.plot(epochs, loss_train, 'g', label='Training loss', marker = 'o',
linestyle="--")
plt.plot(epochs, loss_val, 'b', label='validation loss', marker = 'o',
linestyle="--")
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
loss_train = history.history['accuracy']

```

```

loss_val = history.history['val_accuracy']
epochs = range(1, no_epochs+1)
plt.plot(epochs, loss_train, 'r', label='Training Accuracy', marker = 'o',
linestyle="--")
plt.plot(epochs, loss_val, 'm', label='validation Accuracy', marker = 'o',
linestyle="--")
plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.show()
predictions = model.predict(X_test, batch_size=32, verbose=0)
y_classes = predictions.argmax(axis=-1)
cm = confusion_matrix(y_test, y_classes)
print(cm)
def plot_confusion_matrix(cm, classes,
                        normalize=False,
                        title="Confusion Matrix",
                        cmap = "Reds"):

    plt.figure(figsize=(11,8)) #size of the matrix
    plt.imshow(cm, interpolation="nearest" , cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, fontsize=14, rotation=45)
    plt.yticks(tick_marks, classes, fontsize=14)

    thresh = cm.max() / 2
    for i in range(cm.shape[0]):
        for j in range(cm.shape[1]):
            plt.text(j, i, cm[i, j],
                    horizontalalignment="center", fontsize=15, #numbers
inside the matrix
                    color="white" if cm[i, j]> thresh else "black")
    plt.tight_layout()
    plt.ylabel("True class", fontsize=14)
    plt.xlabel("Predicted class", fontsize=14)
plot_confusion_matrix(cm=cm,
                    classes=CATEGORIES,
                    title="Confusion Matrix")

```

## ΠΑΡΑΡΤΗΜΑ ΙΙ (CNN)

```
import os
import glob
import numpy as np
import cv2
from skimage.io import imread
import matplotlib.pyplot as plt
import pandas as pd
import random
import tensorflow as tf
import keras
import seaborn as sn
from keras.utils import plot_model
from keras.layers import Dense, Activation, Flatten, Conv2D, MaxPooling2D,
Dropout
from keras.models import Sequential
from keras.utils import to_categorical
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
gpu = tf.config.list_physical_devices('GPU') #making sure the model runs on
gpu
tf.config.experimental.set_memory_growth(gpu[0], True)
print("you have", len(gpu), "GPU")
DATADIR = "C:\\Users\\TESTO\\Desktop\\ΣΧΟΛΗ\\ΠΤΥΧΙΑΚΗ\\dataset\\blnw-images-
224"
CATEGORIES = ['bolt', 'locatingpin', 'nut', 'washer']
os.chdir(DATADIR) # Reads the dataset
list_of_image = glob.glob("**/*.png") # Reads images
list_of_classes = glob.glob("**") # Reads classes
print("Total number of files " + str(len(list_of_image))) # Total data
print("Total Number of classes " + str(len(list_of_classes))) # Total Number
of classes
class_index = {}
for category in range(len(CATEGORIES)):
    class_index[category] = CATEGORIES[category]

class_index
y = []

for i, data in enumerate(list_of_classes, 0):
    list_of_images_in_folder = glob.glob(data+"/*.png")
    for j in list_of_images_in_folder:
        y.append(i)

y = np.asarray(y).reshape(-1,1) # Convert the list to numpy array
y.shape
image_list = []
IMG_SIZE = 70
for i, each in enumerate(list_of_image, 1):
    im = imread(each, as_gray = True) # Convert images to gray and read as
an array
```

```

new_array = cv2.resize(im, (IMG_SIZE, IMG_SIZE)) # Reshape images
image_list.append(new_array) # Add to this list

X = np.asarray(image_list) # Convert the list to numpy array
X = X.reshape(-1, IMG_SIZE, IMG_SIZE, 1)
X.shape
plt.imshow(X[0].reshape(IMG_SIZE, IMG_SIZE), cmap = plt.cm.binary)
plt.show()
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    train_size=0.8,
                                                    test_size=0.2,
                                                    shuffle=True,
                                                    random_state=1)

print("Shape of training data " + str(X_train.shape) )
print("Shape of testing data "+ str(X_test.shape) )
print("Shape of training label "+ str(y_train.shape) )
print("Shape of testing label "+ str(y_test.shape) )
Y_train_one_hot = to_categorical(y_train)
Y_test_one_hot = to_categorical(y_test)
print(y_train.shape, Y_test_one_hot.shape)
#Creation of a 3 layer CNN Sequential Model
model = Sequential()

model.add(Conv2D(32, (3,3), strides=(1, 1), padding='same',
input_shape=(IMG_SIZE, IMG_SIZE, 1))) # first layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))

model.add(Conv2D(64, (3,3), strides=(1, 1), padding='same')) # second layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(128, (3,3), strides=(1, 1), padding='same')) # third layer
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())

model.add(Dense(256, activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(4)) # output layer
model.add(Activation('softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(), metrics=['accuracy'])
model.summary()
no_epochs = 20
history = model.fit(X_train, Y_train_one_hot,
                   epochs = no_epochs,
                   validation_data=(X_test, Y_test_one_hot),
                   verbose=1) #Actual Training of model
loss_train = history.history['loss']

```

```

loss_val = history.history['val_loss']
epochs = range(1,no_epochs+1)
plt.plot(epochs, loss_train, 'b', label='Training loss', marker = 'o',
linestyle="--")
plt.plot(epochs, loss_val, 'm', label='validation loss', marker = 'o',
linestyle="--")
plt.title('Training and Validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
loss_train = history.history['accuracy']
loss_val = history.history['val_accuracy']
epochs = range(1,no_epochs+1)
plt.plot(epochs, loss_train, 'r', label='Training Accuracy', marker = 'o',
linestyle="--")
plt.plot(epochs, loss_val, 'g', label='validation Accuracy', marker = 'o',
linestyle="--")
plt.title('Training and Validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
predictions = model.predict(X_test, verbose=1)
y_classes = predictions.argmax(axis=-1)
cm = confusion_matrix(y_test, y_classes)
print(cm)
def plot_confusion_matrix(cm, classes,
                        normalize=False,
                        title="Confusion Matrix",
                        cmap = "Reds"):

    plt.figure(figsize=(11,8)) #size of the matrix
    plt.imshow(cm, interpolation="nearest" , cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, fontsize=14, rotation=45)
    plt.yticks(tick_marks, classes, fontsize=14)

    thresh = cm.max() / 2
    for i in range(cm.shape[0]):
        for j in range(cm.shape[1]):
            plt.text(j, i, cm[i, j],
                    horizontalalignment="center", fontsize=15, #numbers
inside the matrix
                    color="white" if cm[i, j]> thresh else "black")
    plt.tight_layout()
    plt.ylabel("True class", fontsize=14)
    plt.xlabel("Predicted class", fontsize=14)
plot_confusion_matrix(cm=cm,
                    classes=CATEGORIES,
                    title="Confusion Matrix")

```

# ΒΙΒΛΙΟΓΡΑΦΙΑ

## ΒΙΒΛΙΟΓΡΑΦΙΑ:

Γεωργούλη Κ. , (2015) , «ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ Μια Εισαγωγική Προσέγγιση» , Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών, ανακτήθηκε από: :

[http://repfiles.kallipos.gr/html\\_books/93/01a-main.html#\\_idTextAnchor002](http://repfiles.kallipos.gr/html_books/93/01a-main.html#_idTextAnchor002)

Βλαχάβας Ι. , Κεφαλάς Π. , Βασιλειάδης Ν. , Κόκκορας Φ. , Σακελλαρίου Η., (2006), «Τεχνητή Νοημοσύνη» , Εκδόσεις Πανεπιστημίου Μακεδονίας , Γ' έκδοση

## ΔΙΑΔΙΚΤΥΑΚΕΣ ΠΗΓΕΣ :

1. <http://kelifos.physics.auth.gr/COURSES/neural/K1.pdf?fbclid=IwAR0LVLgF-ywCzB-unLNSPH4qrOaqVmerXp2dMj9-Yz6vJ6MQThZSFomjbtC>
2. <https://www.mdpi.com/2076-3417/10/17/6048/html>
3. <https://medium.com/@rinu.gour123/artificial-neural-network-for-machine-learning-structure-layers-2a275f73f473>
4. [https://www.researchgate.net/publication/336267803\\_Comprehensive\\_Review\\_of\\_Artificial\\_Neural\\_Network\\_Applications\\_to\\_Pattern\\_Recognition](https://www.researchgate.net/publication/336267803_Comprehensive_Review_of_Artificial_Neural_Network_Applications_to_Pattern_Recognition)
5. <https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPLBook2ndEd.pdf>
6. [https://www.slideshare.net/Ahmed\\_hashmi/neural-network-its-applications](https://www.slideshare.net/Ahmed_hashmi/neural-network-its-applications)
7. <https://www.allerin.com/blog/how-ai-is-propelling-aerospace-engineering-and-development>
8. [http://repfiles.kallipos.gr/html\\_books/93/04a-main.html](http://repfiles.kallipos.gr/html_books/93/04a-main.html)
9. [https://datacadamia.com/data\\_mining/bias\\_trade-off](https://datacadamia.com/data_mining/bias_trade-off)
10. <https://searchcio.techtarget.com/definition/artificial-neuron>
11. <https://ai.plainenglish.io/artificial-intelligence-vs-machine-learning-vs-deep-learning-whats-the-difference-dccce18efe7f>
12. [https://en.wikipedia.org/wiki/Recurrent\\_neural\\_network#/%20media%20/](https://en.wikipedia.org/wiki/Recurrent_neural_network#/%20media%20/)
13. <https://www.programmingsought.com/article/74024489636/>
14. <https://www.analyticssteps.com/blogs/what-are-different-loss-functions-used-optimizers-neural-networks>
15. <https://builtin.com/data-science/gradient-descent>
16. <https://towardsdatascience.com/an-introduction-to-gradient-descent-and-backpropagation-81648bdb19b2>
17. <https://pub.towardsai.net/do-you-understand-gradient-descent-and-backpropagation-most-dont-929d65f57a6c>
18. [https://www.researchgate.net/figure/Example-of-a-max-pooling-layer-the-maximum-value-out-of-a-specific-window-appears-in\\_fig4\\_333411007](https://www.researchgate.net/figure/Example-of-a-max-pooling-layer-the-maximum-value-out-of-a-specific-window-appears-in_fig4_333411007)
19. [https://www.researchgate.net/publication/350955402\\_Introduction\\_to\\_Convolutional\\_Neural\\_Networks](https://www.researchgate.net/publication/350955402_Introduction_to_Convolutional_Neural_Networks)
20. [https://alinlab.kaist.ac.kr/resource/Lec1\\_Introduction\\_to\\_NN.pdf](https://alinlab.kaist.ac.kr/resource/Lec1_Introduction_to_NN.pdf)
21. [https://www.inf.ufpr.br/todt/IAaplicada/CNN\\_Presentation.pdf](https://www.inf.ufpr.br/todt/IAaplicada/CNN_Presentation.pdf)

22. [https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture\\_10\\_3\\_deep\\_learning.pdf](https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture_10_3_deep_learning.pdf)
23. <https://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-m-hdstat-rnn-deep-learning.pdf>
24. <https://purnasaigudikandula.medium.com/a-beginner-intro-to-neural-networks-543267bda3c8>
25. <https://towardsdatascience.com/a-gentle-introduction-to-neural-networks-series-part-1-2b90b87795bc>
26. <https://www.smartsheet.com/neural-network-applications>
27. <https://www.athinodromio.gr/%CE%B5%CF%86%CE%B1%CF%81%CE%BC%CE%BF%CE%B3%CE%B5%CF%82-%CF%84%CE%B5%CF%87%CE%BD%CE%B7%CF%84%CE%AE%CF%82-%CE%BD%CE%BF%CE%B7%CE%BC%CE%BF%CF%83%CF%8D%CE%BD%CE%B7%CF%82-%CE%BA%CE%B1%CE%B9-%CE%BF%CE%B9/>
28. <https://www.analyticsvidhya.com/blog/2021/03/basics-of-neural-network/>
29. <https://docplayer.gr/32219797-Eisagogi-stin-ypologistiki-noimosyni.html>
30. <http://users.sch.gr/jenyk/index.php/artificialintelligence/ai-historicalreview/5-historicalroute>
31. <https://www.javatpoint.com/machine-learning>
32. <https://www.geomeletitiki.gr/%CE%B7-%CE%B3%CE%AD%CE%BD%CE%BD%CE%B7%CF%83%CE%B7-%CF%84%CE%B7%CF%82-%CF%84%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%CF%82-%CE%B1/>
33. <https://www.bdo.co.za/getattachment/Industries/Financial-Services/Emerging-Tech-Toolbox/Artificial-Intelligence-and-Machine-Learning/FS-AI-Whitepaper-design-July-2020.pdf>
34. <https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/>
35. <https://sebastianraschka.com/faq/docs/logisticregr-neuralnet.html>
36. <https://www.intechopen.com/chapters/61389>
37. <https://deepomatic.com/en/what-is-image-recognition>
38. [https://en.wikipedia.org/wiki/Activation\\_function](https://en.wikipedia.org/wiki/Activation_function)
39. [https://en.wikipedia.org/wiki/Softmax\\_function](https://en.wikipedia.org/wiki/Softmax_function)
40. [https://en.wikipedia.org/wiki/Sigmoid\\_function](https://en.wikipedia.org/wiki/Sigmoid_function)
41. [https://en.wikipedia.org/wiki/Rectifier\\_\(neural\\_networks\)](https://en.wikipedia.org/wiki/Rectifier_(neural_networks))
42. <https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-work-c7cad873ea7>
43. <https://builtin.com/data-science/gradient-descent>
44. <https://deeptai.org/machine-learning-glossary-and-terms/weight-artificial-neural-network>
45. [https://en.wikipedia.org/wiki/Multilayer\\_perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron)
46. <https://el.wikipedia.org/wiki/Perceptron>



47. <https://el.wikipedia.org/wiki/%CE%9D%CE%B5%CF%85%CF%81%CF%89%CE%BD%CE%B9%CE%BA%CF%8C%CE%B4%CE%AF%CE%BA%CF%84%CF%85%CE%BF>
48. <https://iopscience.iop.org/article/10.1088/1757-899X/688/3/033058/pdf>
49. <https://www.sciencedirect.com/science/article/pii/S1674775520301426>
50. [https://www.researchgate.net/publication/329418352\\_Application\\_of\\_artificial\\_intelligence\\_principles\\_in\\_mechanical\\_engineering](https://www.researchgate.net/publication/329418352_Application_of_artificial_intelligence_principles_in_mechanical_engineering)
51. <https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480>
52. [https://repository.kallipos.gr/bitstream/11419/3382/1/02\\_chapter\\_04.pdf](https://repository.kallipos.gr/bitstream/11419/3382/1/02_chapter_04.pdf)
53. <https://viso.ai/deep-learning/ann-and-cnn-analyzing-differences-and-similarities/?fbclid=IwAR13Z6h6EOPdSf7mKVL4HwTX-88xL4eTnSYtmEaiZuTU5YUiZmGCdBwTYRk>
54. <https://medium.com/kaggle-blog>
55. <https://pythonprogramminglanguage.com/machine-learning-tasks/>
56. <https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/>
57. <https://people.iese.ihu.gr/~kdiamant/MachineLearning/MachineLearningLesson01.pdf>
58. [http://www.cad.zju.edu.cn/home/zhx/csmath/lib/exe/fetch.php?media=2011:presentation\\_ml\\_by\\_ibrar.pdf](http://www.cad.zju.edu.cn/home/zhx/csmath/lib/exe/fetch.php?media=2011:presentation_ml_by_ibrar.pdf)
59. <https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neural-network/>
60. <https://www.tensorflow.org/guide>
61. <https://keras.io/>
62. <https://keras.io/guides/>
63. <https://www.quora.com/How-is-PyTorch-different-from-TensorFlow-What-are-the-advantages-of-using-one-vs-the-other-When-should-I-use-one-or-the-other>
64. <https://github.com/Promech2020/Mech-parts-classification-seq-model?fbclid=IwAR3-cJBGfmm-tlCgurHi-VzAF2SQrcTxdBptuqKL1HSpdtOg8hznPBqhGBY>
65. <https://www.embitel.com/blog/embedded-blog/artificial-intelligence-for-automotive-applications>
66. <https://www.it-jim.com/blog/applications-of-artificial-intelligence-in-automotive-industry/>
67. <https://builtin.com/artificial-intelligence/artificial-intelligence-automotive-industry>
68. <https://www.apriorit.com/dev-blog/728-ai-applications-automotive-industry>
69. <https://www.linkedin.com/pulse/how-ai-used-automotive-industry-application-murari-sharma>
70. <https://www.aviationtoday.com/2020/05/15/neural-networks-already-showing-future-potential-aerospace/>
71. <https://www.intechopen.com/chapters/61389>
72. [https://www.researchgate.net/publication/326054669\\_Artificial\\_Intelligence\\_Application\\_in\\_Machine\\_Condition\\_Monitoring\\_and\\_Fault\\_Diagnosis](https://www.researchgate.net/publication/326054669_Artificial_Intelligence_Application_in_Machine_Condition_Monitoring_and_Fault_Diagnosis)

73. <https://www.spotlightmetal.com/machine-learning-for-metal-casting--hype-or-opportunity-a-927713/>
74. <https://www.cio.com/article/3355239/making-automotive-manufacturing-smarter-with-ai.html>
75. <https://www.scnsoft.com/blog/iot-for-condition-monitoring>
76. Διαφάνεια ΤΕΧΝΗΤΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ από το μάθημα ΡΟΜΠΟΤΙΚΗ Πανεπιστημίου Πελοποννήσου, του κ. Θεοφάνη Αραβανή  
<https://eclass.uop.gr/modules/document/file.php/299/%CE%94%CE%99%CE%91%CE%9B%CE%95%CE%9E%CE%95%CE%99%CE%A3/7.%20%CE%91%CE%BD%CE%B1%CF%80%CE%B1%CF%81%CE%AC%CF%83%CF%84%CE%B1%CF%83%CE%B7%20%CE%93%CE%BD%CF%8E%CF%83%CE%B7%CF%82%20%CE%BA%CE%B1%CE%B9%20%CE%A3%CF%85%CE%BB%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CF%84%CE%B9%CE%BA%CE%AE.pdf>
77. Διαφάνεια ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ ΚΑΙ ΣΥΛΛΟΓΙΣΤΙΚΗ από το μάθημα ΡΟΜΠΟΤΙΚΗ Πανεπιστημίου Πελοποννήσου, του κ. Θεοφάνη Αραβανή  
<https://eclass.uop.gr/modules/document/file.php/299/%CE%94%CE%99%CE%91%CE%9B%CE%95%CE%9E%CE%95%CE%99%CE%A3/8.%20%CE%A4%CE%B5%CF%87%CE%BD%CE%B7%CF%84%CE%AC%20%CE%9D%CE%B5%CF%85%CF%81%CF%89%CE%BD%CE%B9%CE%BA%CE%AC%20%CE%94%CE%AF%CE%BA%CF%84%CF%85%CE%B1.pdf>
78. <https://morioh.com/p/15c995420be6>
79. <https://medium.com/analytics-vidhya/from-convolutional-neural-network-to-variational-auto-encoder-97694e86bb51>
80. <https://miaodi.github.io/machine%20learning/neural-network/>
81. <https://el.wikipedia.org/wiki/%CE%98%CE%B5%CF%89%CF%81%CE%AF%CE%B1%CF%84%CE%B7%CF%82%CF%80%CE%BF%CE%BB%CE%BB%CE%B1%CF%80%CE%BB%CE%AE%CF%82%CE%BD%CE%BF%CE%B7%CE%BC%CE%BF%CF%83%CF%8D%CE%BD%CE%B7%CF%82>
82. <https://stackoverflow.com/questions/34518656/how-to-interpret-loss-and-accuracy-for-a-machine-learning-model>
83. <https://intellipaat.com/community/368/how-to-interpret-loss-and-accuracy-for-a-machine-learning-model>

## ΠΗΓΕΣ ΕΙΚΟΝΩΝ

### Κεφάλαιο 1<sup>ο</sup>

1.1 Δοκιμασία Τούρινγκ, εικόνα απο [https://en.wikipedia.org/wiki/Turing\\_test](https://en.wikipedia.org/wiki/Turing_test)

1.2: Συζήτηση με το ELIZA, εικόνα από :

[https://en.wikipedia.org/wiki/ELIZA#/media/File:ELIZA\\_conversation.png](https://en.wikipedia.org/wiki/ELIZA#/media/File:ELIZA_conversation.png)

1.3 Κλασσικός προγραμματισμός αντί της Μηχανική Μάθηση, εικόνα απο: <https://datalya.com/blog/machine-learning/machine-learning-vs-traditional-programming-paradigm>

1.4 Απεικόνιση ενός απλού NN και ενός DNN, εικόνα από <http://www.deep-space.ch/index.php/en/10-technologies/9-deep-learning>

1.5 : Λειτουργία Ευφυούς Πράκτορα, εικόνα από

[https://www.researchgate.net/figure/Working-with-intelligent-agents\\_fig1\\_333907788](https://www.researchgate.net/figure/Working-with-intelligent-agents_fig1_333907788)

### Κεφάλαιο 2<sup>ο</sup>

2.1 Παραγωγή χυτοσιδήρου με επισυναπτώμενο νευρωνικό δίκτυο, εικόνα από:

<file:///C:/Users/user/Downloads/applsci-10-06048.pdf>

2.2 Απεικόνιση παρακολούθησης παραμέτρων κατάστασης, εικόνα από:

<https://www.scnsoft.com/blog/iot-for-condition-monitoring>

### Κεφάλαιο 3<sup>ο</sup>

3.1 Απεικόνιση Φυσικού νευρώνα, εικόνα από:

[http://repfiles.kallipos.gr/html\\_books/93/04a-main.html](http://repfiles.kallipos.gr/html_books/93/04a-main.html)

3.2 Μοντέλο τεχνητού νευρώνα [κύκλος] όπου,  $S_{1 \rightarrow 3}$  : είσοδοι και  $W_{1 \rightarrow 3}$  : βάρη, εικόνα

από: <http://kelifos.physics.auth.gr/COURSES/neural/K1.pdf?fbclid=IwAR0LVLgF-ywCzB-unLNSPH4qrOaqVmerXp2dMj9-Yz6vJ6MQThZSFomjbtC>

3.3 Γράφημα σιγμοειδούς συνάρτησης, εικόνα από:

[https://en.wikipedia.org/wiki/Sigmoid\\_function#/media/File:Logistic-curve.svg](https://en.wikipedia.org/wiki/Sigmoid_function#/media/File:Logistic-curve.svg)

3.4 Γράφημα Softmax, εικόνα από: [https://www.machinecurve.com/wp-content/uploads/2020/01/softmax\\_logits.png](https://www.machinecurve.com/wp-content/uploads/2020/01/softmax_logits.png)

3.5 Γραφική απεικόνιση ReLU, εικόνα από: [https://www.researchgate.net/figure/ReLU-activation-function\\_fig7\\_333411007](https://www.researchgate.net/figure/ReLU-activation-function_fig7_333411007)

3.6 Γραφική απεικόνιση ενός μη γραμμικού τυπικού τεχνητού νευρώνα, εικόνα από: <https://devepaper.com/why-do-the-activation-functions-of-neural-networks-use-non-linear-functions/>

3.7 Αρχιτεκτονική τεχνητού νευρωνικού δικτύου (ΤΝΔ), εικόνα από: [https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o\\_fig1\\_321259051](https://www.researchgate.net/figure/Artificial-neural-network-architecture-ANN-i-h-1-h-2-h-n-o_fig1_321259051)

3.8 Επιλογή γειτόνων για την ταξινόμηση της άγνωστης πράσινης παρατήρησης, εικόνα από: [https://en.wikipedia.org/wiki/Knearest\\_neighbors\\_algorithm#/media/File:KnnClassification.svg](https://en.wikipedia.org/wiki/Knearest_neighbors_algorithm#/media/File:KnnClassification.svg)

3.9 Σφάλμα γενίκευσης συναρτήσει της πολυπλοκότητας του μοντέλου ΤΝΔ, εικόνα από: <https://ravedata.in/machine-learning/the-overfitting-problem/>

3.10 Γραφική απεικόνιση της απότομης καθόδου, εικόνα από: <https://morioh.com/p/15c995420be6>

3.11 Αλγόριθμος οπισθοδιάδοσης διαισθητικά, εικόνα από: <https://www.i2tutorials.com/how-do-neural-networks-update-weights-and-biases-during-back-propagation/>

3.12 Αρχιτεκτονική CNN μοντέλου, εικόνα από: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

## Κεφάλαιο 4<sup>ο</sup>

4.1 Παράδειγμα πολλαπλασιασμού πινάκων, εικόνα από: <https://sausheong.github.io/posts/how-to-build-a-simple-artificial-neural-network-with-go/>

4.2 Το συνελικτικό φίλτρο εφαρμόζεται σε στρώμα εισόδου και φαίνονται τα αποτελέσματα στο στρώμα εξόδου, εικόνα από: <https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411>

- 4.3: Valid padding όπου στοιχεία που περισσεύουν απορρίπτονται, εικόνα από: <https://labs.bawi.io/deep-learning-convolutional-neural-networks-7992985c9c7b>
- 4.4: Zero padding όπου συμπληρώνονται μηδενικά στα περιθώρια του πίνακα ,εικόνα από: <https://labs.bawi.io/deep-learning-convolutional-neural-networks-7992985c9c7b>
- 4.5: Τύποι στρωμάτων συγκέντρωσης, εικόνα από: [https://www.researchgate.net/figure/Max-pooling-and-average-pooling\\_fig5\\_343675998](https://www.researchgate.net/figure/Max-pooling-and-average-pooling_fig5_343675998)
- 4.6: Τρισδιάστατος πίνακας RGB. Κάθε στρώμα είναι ένας δισδιάστατος πίνακας τιμών pixel, εικόνα από: [https://www.researchgate.net/figure/A-three-dimensional-RGB-matrix-Each-layer-of-the-matrix-is-a-two-dimensional-matrix\\_fig6\\_267210444](https://www.researchgate.net/figure/A-three-dimensional-RGB-matrix-Each-layer-of-the-matrix-is-a-two-dimensional-matrix_fig6_267210444)
- 4.7: Ενδεικτικό εξάρτημα, πριν και μετά τα φίλτρα από δεξιά προς τα αριστερά
- 4.8: Οπτικοποίηση των εξαρτημάτων που χρησιμοποιήθηκαν ως βάση δεδομένων
- 4.9: Διαχωρισμός μηχανολογικών εξαρτημάτων με labels από 0 έως 3
- 4.10: Μετατροπή ενός τρισδιάστατου πίνακα σε μονοδιάστατο με το flatten-layer, εικόνα από: <https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480>
- 4.11: Νευρωνικό δίκτυο που αποτελείται από ένα input layer, τρία hidden layers και ένα output layer: <https://miaodi.github.io/machine%20learning/neural-network/>
- 4.12: Σύνοψη του μοντέλου ANN
- 4.13: Γράφημα Κόστους - Εποχών ANN
- 4.14: Γράφημα Ακρίβειας - Εποχών ANN
- 4.15: Confusion Matrix ANN, ύστερα από την αξιολόγηση
- 4.16: Σύνοψη του μοντέλου CNN
- 4.17: Γράφημα Κόστους - Εποχών CNN
- 4.18: Γράφημα Ακρίβειας - Εποχών CNN
- 4.19: Confusion Matrix CNN, ύστερα από την αξιολόγηση.

## ΑΡΙΘΜΗΣΗ ΠΙΝΑΚΩΝ

Πίνακας 4.1: Διαμοιρασμός του Dataset σε 80-20% για training και test.

Πίνακας 4.2: Μοντέλο Νευρωνικού δικτύου με dense layers

Πίνακας 4.3: Μοντέλο συνελκτικού νευρωνικού δικτύου