Πανεπιστήμιο Δυτικής Αττικής

Τμήμα Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΑΝΑΓΝΩΡΙΣΗ ΣΥΝΑΙΣΘΗΜΑΤΩΝ ΜΕ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ»

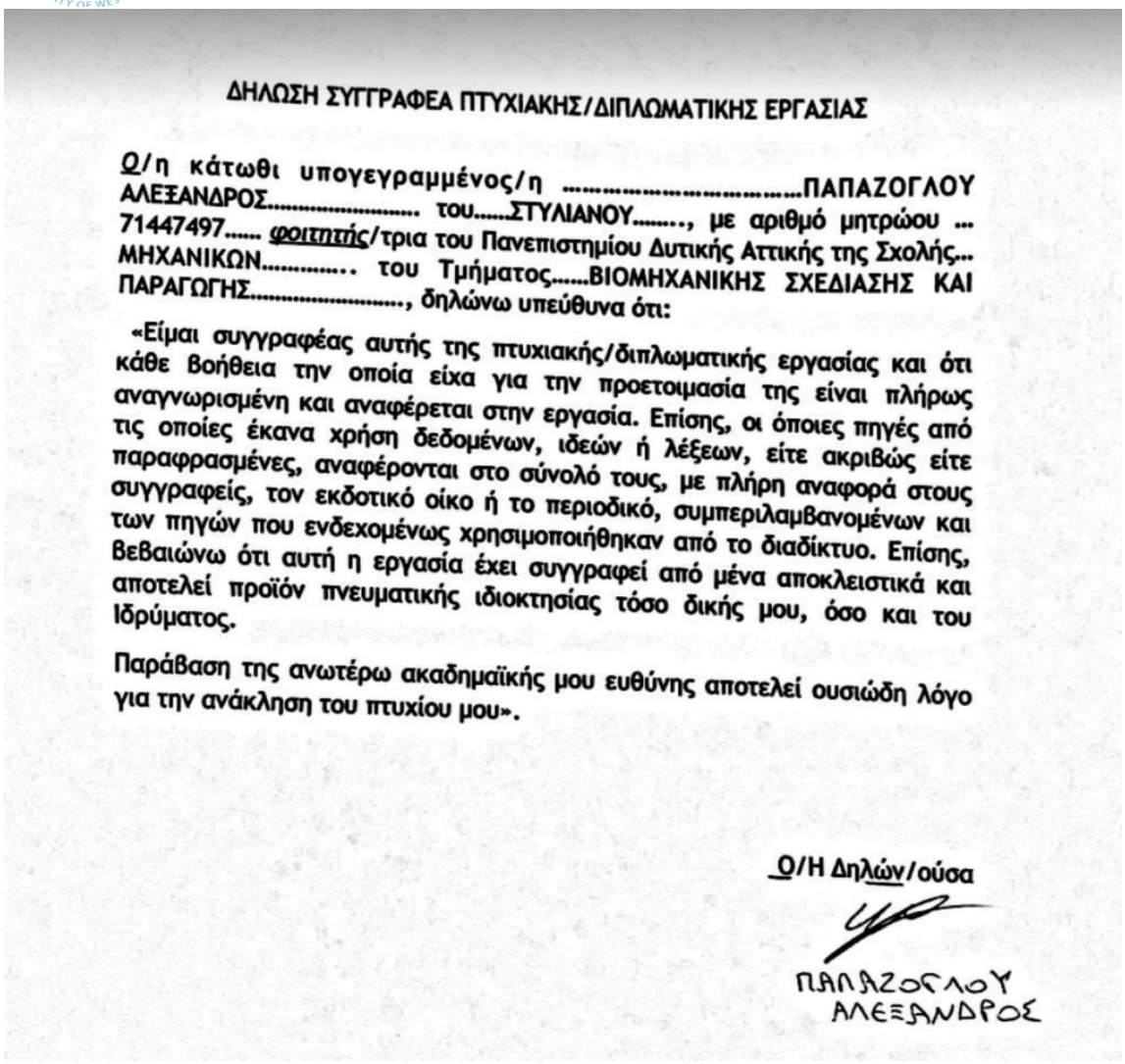
« ΠΑΠΑΖΟΓΛΟΥ ΑΛΕΞΑΝΔΡΟΣ »

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ : ΝΙΚΟΛΑΟΥ ΓΡΗΓΟΡΗΣ

ΑΘΗΝΑ,ΦΕΒΡΟΥΑΡΙΟΣ,2022

Η Διπλωματική/Πτυχιακή Εργασία εξετάστηκε από την κάτωθι Εξεταστική Επιτροπή:

ΟΝΟΜΑ ΕΠΩΝΥΜΟ	ΒΑΘΜΙΔΑ	ΨΗΦΙΑΚΗ ΥΠΟΓΡΑΦΗ
ΝΙΚΟΛΑΟΥ Γ.	ΛΕΚΤΟΡΑΣ	GRIGORIOS NIKOLAOU Digitally signed by GRIGORIOS NIKOLAOU Date: 2022.03.22 12:26:13 +02'00'
ΒΑΣΙΛΕΙΑΔΟΥ Σ.	ΕΠΙΚΟΥΡΗ ΚΑΘΗΓΗΤΡΙΑ	Soultana Vasileiadou Digitally signed by Soultana Vasileiadou Date: 2022.03.24 07:54:44 +02'00'
ΔΡΟΣΟΣ Χ.	ΕΔΙΠ	Christos Drosos Digitally signed by Christos Drosos Date: 2022.03.22 14:13:25 +02'00'



ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ/ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/η κάτωθι υπογεγραμμένος/η .....ΠΑΠΑΖΟΓΛΟΥ ΑΛΕΞΑΝΔΡΟΣ..... του.....ΣΤΥΛΙΑΝΟΥ....., με αριθμό μητρώου ... 71447497..... φοιτητής/τρια του Πανεπιστημίου Δυτικής Αττικής της Σχολής... ΜΗΧΑΝΙΚΩΝ..... του Τμήματος.....ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ....., δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της πτυχιακής/διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο/Η Δηλών/ούσα

ΠΑΠΑΖΟΓΛΟΥ ΑΛΕΞΑΝΔΡΟΣ

Προτεινόμενα μέλη 3μελούς επιτροπής εξέτασης:
1)_____ΓΡΗΓΟΡΗΣ ΝΙΚΟΛΑΟΥ_____
2)_____ΒΑΣΙΛΕΙΑΔΟΥ ΣΟΥΛΤΑΝΑ_____
3)_____ΧΡΗΣΤΟΣ ΔΡΟΣΟΣ_____

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου, που με με στήριξε κατά την διάρκεια εκπόνησης της εργασίας αλλά ακόμα περισσότερο επιθυμώ να ευχαριστήσω τον καθηγητή μου Γρηγόρη Νικολάου, για την συνεχή και άμεση βοήθεια που μου παρείχε σε ότι εμπόδιο συναντούσα σχετικά με την διπλωματική! Εκτιμώ τόσο την συνεργασία όσο και την παροχή των πολλών γνώσεων του!!!

ΠΑΠΑΖΟΓΛΟΥ ΑΛΕΞΑΝΔΡΟΣ

ΠΕΡΙΛΗΨΗ

Η δυνατότητα των ρομπότ να αναγνωρίζουν συναισθήματα θα παίξει καθοριστικό ρόλο στην ανάπτυξη και δημιουργία ρομπότ που θα ειδικεύονται στην παροχή ειδικών υπηρεσιών στα επερχόμενα χρόνια. Παρ' όλα αυτά, δεν έχουμε κατορθώσει την αναγνώριση μερικών συναισθημάτων σε ικανοποιητικό επίπεδο ακόμα. Για αυτόν τον λόγο δεν έχουν παρουσιαστεί ακόμα στην αγορά! Αλλά πρέπει να αναλογιστούμε το εξής, αν η αναγνώριση συναισθημάτων αποτελεί μια απαιτητική δοκιμασία για τους ανθρώπους, θα είναι ακόμα πιο δύσκολη για τα ρομπότ, δεδομένου ότι δεν μπορούν να νιώσουν συναισθήματα ή συμπόνια. Για την αναγνώριση συναισθημάτων χρησιμοποιούνται η βαθιά και η μηχανική μάθηση. Μπορούν όμως αυτές οι δύο να αναγνωρίσουν με καλή απόδοση τις διάφορες εκφράσεις του προσώπου/ συναισθήματα μας;

Στην παρούσα διπλωματική εργασία, μελετήθηκαν τα σύγχρονα κορυφαία μοντέλα που αναγνωρίζουν τα 6 βασικά συναισθήματα (χαρά, λύπη, φόβος, θυμός, έκπληξη ,αηδία αλλά και ουδετερότητα),και έγινε σύγκριση τους ως προς την αρχιτεκτονική τους, την αποδοτικότητα τους καθώς και την χρησιμότητα τους στην καθημερινότητα. Στην συνέχεια, έγινε χρήση μερικών εκ των προ εκπαιδευμένων μοντέλων τόσο στο σετ δεδομένων FER2013 όσο και σε φωτογραφίες δικές μου για να δούμε και να τεστάρουμε πόσο καλά λειτουργεί η συγκεκριμένη τεχνολογία. Επίσης επιχειρήθηκε η δημιουργία δικού μου μοντέλου με την χρήση νευρωνικών δικτύων και "μεταφοράς μάθησης" με τον στόχο ξανά της αναγνώρισης συναισθημάτων τόσο από φωτογραφίες όσο και σε πραγματικό χρόνο! Τα αποτελέσματα ήταν σχετικά μέτρια καθώς δεν είχα τον απαραίτητο/ιδανικό εξοπλισμό για την δημιουργία και εν συνεχεία εκπαίδευση μοντέλου!

ΛΕΞΕΙΣ-ΚΛΕΙΔΙΑ

Τεχνητή νοημοσύνη, μηχανική μάθηση, βαθιά μάθηση, συνελικτικά νευρωνικά δίκτυα, αναγνώριση συναισθημάτων, Deepface, VggFace , FaceNet, OpenFace, "μεταφοράς μάθησης"

«EMOTION RECOGNITION WITH MACHINE LEARNING

«ALEXANDROS PAPAZOGLOU»

ABSTRACT

The ability of robots to recognize emotions will be vital for the creation and development of robots that specialize in providing special services in the coming years. However, we have not achieved the recognition of certain emotions in a sufficiently effective way yet. Therefore, they haven't been introduced in the market. But we need to consider this, if emotion recognition is a very challenging task even for people, it is even more for robots, since they do not feel any emotion and can't empathize. For the sake of emotion recognition, Deep learning and Machine learning are being used. However, can Deep learning and Machine learning recognize our emotions/facial expressions efficiently?

In the present thesis, the current state-state-of-the-art pre-trained models that recognize the 6 basic emotions (joy, sadness, fear, anger, surprise, disgust, and neutrality) were studied and compared with regard to their architecture, efficiency and handiness in everyday life. Moreover, some of the pre-trained models were used both in the FER2013 dataset as well as in personal photographs, to see and test how well this technology works. Also, i tried to create my own model using convolutional neural networks (CNN) and "transfer learning" with the common goal of recognizing emotions from photos and in real time(video)! The results were relatively modest as I did not have the necessary/ideal equipment for creating and training my model!

KEYWORDS

Artificial Intelligence, Machine learning, Deep learning, CNN, emotion recognition, Deepface, VggFace , FaceNet, OpenFace, transfer learning

ΕΙΣΑΓΩΓΗ

Τα τελευταία χρόνια τα υπηρεσιακά ρομπότ έχουν βελτιώσει σημαντικά την απόδοση τους, με βάση αυτό μπορούμε με επιφύλαξη να πούμε πως , όπως και με τα βιομηχανικά ρομπότ, έτσι και με τα υπηρεσιακά θα υπάρξει στα επερχόμενα χρόνια μια "επανάσταση" ! Προτού όμως προκύψει η επανάσταση αυτή, είναι απαραίτητο τα ρομπότ(ειδικά τα ανθρωποειδή) να αντιλαμβάνονται τα συναισθήματα μας έτσι ώστε να προσαρμόζονται καλύτερα στις ανάγκες μας.

Η τεχνητή νοημοσύνη πλέον, έχει μεγάλη συμμετοχή σε κάθε πτυχή της ανθρώπινης ζωής. Οι περισσότερες τεχνολογίες προσαρμόζονται στις ανθρώπινες ανάγκες και η τεχνητή νοημοσύνη είναι αυτή που καθιστά την προσαρμογή αυτή δυνατή!

Οι "τεχνολογίες" / τεχνικές αυτές χρησιμοποιούνται σε αλγόριθμους οι οποίοι με την σειρά τους έχουν σαν στόχο την αναγνώριση των ανθρώπινων συναισθημάτων! Όταν οι άνθρωποι προσπαθούν να επικοινωνήσουν μεταξύ τους, ένα μεγάλο ποσοστό το κάνει δίχως λεκτική επικοινωνία. Πολλές μελέτες έχουν αποδείξει πως οι εκφράσεις του προσώπου έχουν μια αλληλένδετη σχέση με τα ανθρώπινα συναισθήματα. Μπορούμε και καταλαβαίνουμε ο ένας τον άλλο μέσω της δυνατότητας μας να εντοπίζουμε και να αναγνωρίζουμε αυτά τα συναισθήματα. Ο κύριος στόχος αυτού του τμήματος της τεχνητής νοημοσύνης είναι να χρησιμοποιήσει τεχνικές μάθησης έτσι ώστε να δώσει την δυνατότητα στην μηχανή να αναγνωρίζει τα συναισθήματα.

Τα συναισθήματα παίζουν μεγάλο ρόλο στον τομέα της ψυχολογίας αλλά και στον τομέα της διαχείρισης και προώθησης προϊόντων στην αγορά, μιας που πρέπει να γνωρίζουν οι επιχειρήσεις ότι οι πελάτες είναι ικανοποιημένοι. Τα ρομπότ όμως δεν μπορούν να νιώσουν τα συναισθήματα αυτά, ούτε έχουν την δυνατότητα να αναγνωρίσουν τι νιώθει ένα άτομο σε σύγκριση με αυτά. Οι διαθέσιμες πληροφορίες είναι μεγάλες "μήτρες" που αντιπροσωπεύουν τις υπάρχουσες εικόνες και άλλες επιπλέον πληροφορίες που μπορεί να λαμβάνουν από αισθητήρες ή μικρόφωνα. Από εκεί ξεκινάει η δουλεία. Γι' αυτόν τον λόγο, η αναγνώριση των συναισθημάτων από έναν υπολογιστή είναι περίπλοκη.

Για να το πετύχουμε αυτό πρέπει να ανακαλύψουμε, τα πραγματικά συναισθήματα που νιώθει το ανθρώπινο όν, ποια τεχνολογία είναι ιδανική για να τα καταγράψουμε και τέλος ποια μοντέλα και αλγόριθμοι είναι κατάλληλοι. Λόγω της ύπαρξης όλων αυτών των μεταβλητών , το “πρόβλημα" αυτό δεν έχει λυθεί ικανοποιητικά ακόμη.

Σε αυτή την διπλωματική , ερευνούμε και δοκιμάζουμε μερικά από τα κορυφαία μοντέλα μηχανικής και βαθιάς μάθησης στο σετ δεδομένων FER2013 και σε μερικές προσωπικές φωτογραφίες. Παρουσιάζουμε την δομή και αρχιτεκτονική πίσω από τα μοντέλα αυτά, την επεξηγούμε , την εφαρμόζουμε σε ένα δικό μας μοντέλο και στο τέλος παρουσιάζουμε τα αποτελέσματα μας και αναλύουμε μελλοντικές εφαρμογές αλλά και πιθανές/ αναγκαίες βελτιώσεις!

ΣΤΑΔΙΑ ΔΙΠΛΩΜΑΤΙΚΗΣ

1) **Επισκόπηση της βιβλιογραφίας**, παρουσιάζονται οι βιβλιογραφικές πηγές της εργασίας.

2) **Παρουσίαση θεωρητικού υπόβαθρου**, αναφέροντας βασικές έννοιες, τεχνολογίες και γνώσεις που συνείσφεραν στην εκπόνηση της εργασίας.

3) **Σχεδιασμός/ Αρχιτεκτονική,** παρουσιάζεται η δομή και ο κώδικας της εργασίας.

4) **Σχολιασμός και Αξιολόγηση.**

5)**Συμπεράσματα.**

6) **Παραπομπές**

1)ΕΠΙΣΚΟΠΗΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑΣ

Στο συγκεκριμένο κεφάλαιο παρουσιάζεται συνοπτικά η βιβλιογραφική ανασκόπηση της διπλωματικής εργασίας.

Πέρα από βιβλία, βασιστήκαμε σε μερικές πήγες του διαδικτύου (που θα αναγράφονται στην βιβλιογραφία) αλλά και πάνω στο **Kaggle**, μέσω του οποίου είχαμε την δυνατότητα να μάθουμε αλλά και να δοκιμάσουμε σχεδόν όλα τα μοντέλα και τους αλγόριθμους τόσο της μηχανικής όσο και της βαθιάς μάθησης. Επίσης , στην βιβλιογραφία παρουσιάζεται 1 διπλωματική εργασία , στην οποία βασιστήκαμε πάνω σχετικά με το κομμάτι των διάφορων τεχνολογιών που χρησιμοποιούνται κατά την διάρκεια αναγνώρισης συναισθημάτων και την σε βάθος επεξήγηση τους! Τέλος , σημαντικό να αναφερθεί είναι, ότι μια από τις κυριότερες πηγές πληροφοριών αποτέλεσαν δύο μαθήματα που παρακολούθησα κατά την 5ετή φοίτηση μου στο πανεπιστήμιο, με τίτλους **Τεχνητή Νοημοσύνη** και **Μηχανική μάθηση και επεξεργασία μεγάλων δεδομένων**! Τα βιβλία αυτά και οι διαφάνειες των μαθημάτων κρίθηκαν σκόπιμο να χρησιμοποιηθούν ως πηγές στην εργασία καθώς προσφέρουν μία κατανοητή και πλήρης παρουσίαση των τεχνολογιών που χρησιμοποιήθηκαν, σε ικανοποιητικό βαθμό που εξυπηρετούσε τον σκοπό της εργασίας. Το γενικότερο συμπέρασμα που προκύπτει από την μελέτη της βιβλιογραφίας είναι ότι αν και οι τεχνολογίες και τα περιβάλλοντα ανάπτυξης που χρησιμοποιούνται απαντώνται στις περισσότερες περιπτώσεις (Keras και Tensorflow), τα νευρωνικά δίκτυα χρησιμοποιούνται για πληθώρα εφαρμογών και υπάρχουν πολλά μοντέλα και τεχνικές που μπορούν να ακολουθηθούν. Η ανίχνευση συναισθημάτων είναι αρκετά δημοφιλής τομέας και τα θέματα των ερευνών καλύπτουν ένα ευρύ πεδίο!

2) ΑΝΑΓΝΩΡΙΣΗ ΣΥΝΑΙΣΘΗΜΑΤΩΝ ΠΡΟΣΩΠΟΥ

Η αναγνώριση συναισθημάτων προσώπου είναι η τεχνολογία που αναλύει τις εκφράσεις του προσώπου τόσο από στατικές εικόνες όσο και από βίντεο για να αποκαλύψει πληροφορίες σχετικά με τη συναισθηματική κατάσταση του ατόμου. Η πολυπλοκότητα των εκφράσεων του προσώπου, η πιθανή χρήση της τεχνολογίας σε οποιοδήποτε πλαίσιο και η εμπλοκή νέων τεχνολογιών όπως η τεχνητή νοημοσύνη εγείρουν σημαντικές εξελίξεις στον συγκεκριμένο τομέα.

Τί είναι η αναγνώριση συναισθημάτων προσώπου[FER];

Η αναγνώριση συναισθημάτων προσώπου είναι μια τεχνολογία που χρησιμοποιείται για την ανάλυση συναισθημάτων από διαφορετικές πηγές, όπως εικόνες και βίντεο. Ανήκει στην οικογένεια των τεχνολογιών που συχνά αναφέρονται ως «συναισθηματικός υπολογισμός», ένα πολυεπιστημονικό πεδίο έρευνας για τις δυνατότητες των υπολογιστών να αναγνωρίζουν και να ερμηνεύουν ανθρώπινα συναισθήματα και συναισθηματικές καταστάσεις και συχνά βασίζεται σε τεχνολογίες Τεχνητής Νοημοσύνης.

Οι εκφράσεις του προσώπου είναι μορφές μη λεκτικής επικοινωνίας, παρέχοντας συμβουλές για ανθρώπινα συναισθήματα. Για δεκαετίες, η αποκωδικοποίηση τέτοιων εκφράσεων συναισθημάτων υπήρξε ερευνητικό ενδιαφέρον στον τομέα της ψυχολογίας (Ekman and Friesen 2003; Lang et al. 1993) αλλά και στο πεδίο αλληλεπίδρασης ανθρώπινου υπολογιστή (Cowie et al. 2001; Abdat et. al. 2011). Πρόσφατα, η υψηλή διάδοση των καμερών και η τεχνολογική πρόοδος στη βιομετρική ανάλυση, η μηχανική μάθηση και η αναγνώριση προτύπων έχουν παίξει εξέχοντα ρόλο στην ανάπτυξη της τεχνολογίας FER.

Πολλές εταιρείες, από γίγαντες τεχνολογίας όπως η NEC ή η Google έως μικρότερες, όπως η Affectiva ή η Eyeris επενδύουν στην τεχνολογία, γεγονός που δείχνει την αυξανόμενη σημασία της. Υπάρχουν επίσης πολλές πρωτοβουλίες προγράμματος έρευνας και καινοτομίας της ΕΕ Horizon20201 που διερευνούν τη χρήση της τεχνολογίας.

Η ανάλυση FER περιλαμβάνει τρία βήματα:

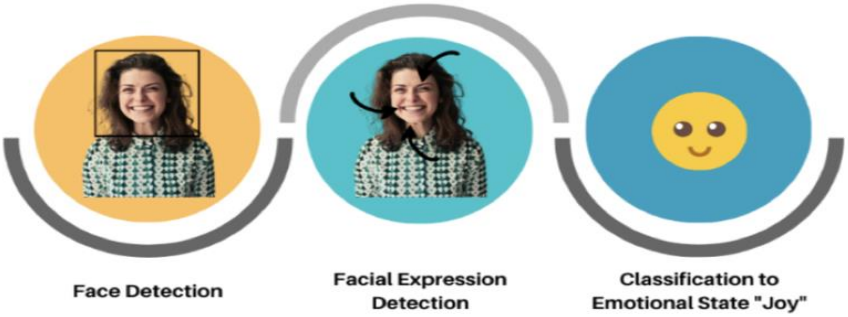
α) αφαίρεση προσώπου,

β) ανίχνευση έκφρασης πρόσωπο

γ) ταξινόμηση έκθεσης σε συναισθηματική κατάσταση (Εικόνα 1)

Η ανίχνευση συναισθημάτων βασίζεται στην ανάλυση των θέσεων ορόσημου του προσώπου (π.χ. τέλος της μύτης, φρύδια). Επιπλέον, στα βίντεο, αναλύονται επίσης οι αλλαγές σε αυτές τις θέσεις, προκειμένου να εντοπιστούν οι συσπάσεις σε μια ομάδα μυών του προσώπου. Ανάλογα με τον αλγόριθμο, οι εκφράσεις του προσώπου μπορούν να ταξινομηθούν σε βασικά συναισθήματα (π.χ. θυμός, αηδία, φόβος, χαρά, λύπη και έκπληξη) ή σύνθετα συναισθήματα (π.χ. ευτυχώς λυπημένος, ευχάριστα έκπληκτος, ευτυχισμένος απογοητευμένος, δυστυχώς φοβισμένος, δυστυχώς θυμωμένος, δυστυχώς έκπληκτος) . Σε άλλες περιπτώσεις, οι εκφράσεις του προσώπου θα μπορούσαν να συνδεθούν με τη φυσιολογική ή ψυχική κατάσταση του νου (π.χ. κούραση ή πλήξη).

Η πηγή των εικόνων ή των βίντεο που χρησιμεύουν ως αλγόριθμοι FER ποικίλλουν από κάμερες παρακολούθησης έως κάμερες τοποθετημένες κοντά σε διαφημιστικές οθόνες στα καταστήματα, καθώς και σε μέσα κοινωνικής δικτύωσης και υπηρεσίες ροής ή προσωπικές συσκευές. Το FER μπορεί επίσης να συνδυαστεί με βιομετρική ταυτοποίηση. Η ακρίβειά του μπορεί να βελτιωθεί με την τεχνολογία που αναλύει διαφορετικούς τύπους πηγών, όπως π.χ. φωνή, κείμενο, δεδομένα υγείας από αισθητήρες ή μοτίβα ροής αίματος που συνάγονται από την εικόνα. Οι πιθανές χρήσεις του FER καλύπτουν ένα ευρύ φάσμα εφαρμογών, παραδείγματα των οποίων παρατίθενται εδώ σε ομάδες, ανά τομέα εφαρμογής.



ΕΦΑΡΜΟΓΕΣ

Παροχή εξατομικευμένων υπηρεσιών:

• ανάλυση συναισθημάτων για την εμφάνιση εξατομικευμένων μηνυμάτων σε έξυπνα περιβάλλοντα

• παρέχουν εξατομικευμένες συστάσεις π.χ. για μουσική επιλογή ή πολιτιστικό υλικό

• να αναλύσει τις εκφράσεις του προσώπου για να προβλέψει την ατομική αντίδραση στις ταινίες

Ανάλυση συμπεριφοράς πελατών και διαφήμιση :

• ανάλυση των συναισθημάτων των πελατών ενώ οι αγορές εστιάζουν είτε σε προϊόντα είτε στη διάταξή τους μέσα στο κατάστημα

• διαφημιστική σήμανση σε σιδηροδρομικό σταθμό χρησιμοποιώντας σύστημα αναγνώρισης και παρακολούθησης προσώπου για σκοπούς μάρκετινγκ

Φροντίδα υγείας:

• ανίχνευση αυτισμού ή νευροεκφυλιστικών ασθενειών

• πρόβλεψη ψυχωτικών διαταραχών ή κατάθλιψης σε προσδιορίζουν τους χρήστες που χρειάζονται βοήθεια

• πρόληψη αυτοκτονίας

• ανίχνευση κατάθλιψης σε ηλικιωμένους

• Παρατήρηση των συνθηκών των ασθενών κατά τη διάρκεια της θεραπείας

Εργασία:

• να βοηθήσει στη λήψη αποφάσεων των προσλήψεων

• εντοπισμός μη ενδιαφερόμενων υποψηφίων σε μια συνέντευξη

• παρακολουθεί τις διαθέσεις και την προσοχή των εργαζομένων

Εκπαίδευση :

• παρακολουθεί την προσοχή των μαθητών

• ανίχνευση συναισθηματικής αντίδρασης σε άτομα που διδάσκονται

• σχεδιασμός συναισθηματικού συστήματος διδασκαλίας

• ανίχνευση συμμετοχής στη διαδικτυακή μάθηση

ΠΟΙΕΣ ΕΙΝΑΙ ΟΙ ΠΡΟΚΛΗΣΕΙΣ ΑΝΑΓΝΩΡΙΣΗΣ ΣΥΝΑΙΣΘΗΜΑΤΩΝ;

Όπως κάθε αναπτυσσόμενη τεχνολογία, η αναγνώριση συναισθημάτων δεν είναι τέλεια και έχει τις ατέλειες και τις προκλήσεις της. Μία από τις προκλήσεις είναι ότι τα σύνολα δεδομένων επισημαίνονται από τους ανθρώπους και διαφορετικά άτομα μπορούν να διαβάζουν και να ερμηνεύουν τα συναισθήματα με διαφορετικούς τρόπους. Επίσης, ορισμένες ορατές οπτικές ενδείξεις, όπως τα φρυγμένα φρύδια, μπορεί να σημαίνουν άλλα συναισθήματα εκτός από το θυμό, και άλλες ενδείξεις μπορεί να είναι λεπτές υποδείξεις θυμού, αν και δεν είναι εμφανείς. Ένα άλλο ζήτημα που αντιμετωπίζει αυτή η τεχνολογία είναι κατά τον εντοπισμό συναισθημάτων από ανθρώπους διαφορετικών χρωμάτων. . Αυτό σημαίνει ότι τα σύνολα εκπαίδευσης πρέπει να είναι πιο διαφορετικά και οι ειδικοί κάνουν ήδη ό, τι μπορούν για να το διορθώσουν.

ΤΟ ΗΘΙΚΟ ΠΡΟΒΛΗΜΑ ΤΗΣ ΑΝΑΓΝΩΡΙΣΗΣ ΣΥΝΑΙΣΘΗΜΑΤΩΝ

Εκτός από τις προκλήσεις που επιβάλλει, η αναγνώριση συναισθημάτων φαίνεται να προκαλεί επίσης ηθικά προβλήματα, καθώς μερικές φορές μπορεί να εισβάλει στον προσωπικό χώρο. Υπάρχουν μέρη όπου η αναγνώριση συναισθημάτων απαγορεύεται από το νόμο, καθώς δεν το "εκτιμούν" όλοι όταν η τεχνητή νοημοσύνη χρησιμοποιεί τα συναισθήματά τους. Υπάρχουν επίσης μέρη, όπως η Καλιφόρνια, όπου απαγορεύεται στις αρχές να χρησιμοποιούν τέτοια τεχνολογία, επειδή παραβιάζει το δικαίωμα των πολιτών και μπορεί να τους κάνει να είναι επιφυλακτικοί απέναντι σε αυτές!

3) ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

Γενικά η μηχανική μάθηση δεν χρησιμοποιείται μονάχα για αναγνώριση εικόνων, αλλά έχει ακόμα πολλές εφαρμογές σε διάφορους τομείς όπως οι ιατρικές διαγνώσεις, συστήματα προβλέψεων και οικονομικές υπηρεσίες. Η αναγνώριση φωνής είναι ένα πολύ διάσημο παράδειγμα βαθιάς μάθησης. Σε αυτό το τμήμα , θα επεξηγήσουμε τα βασικά των δύο παραπάνω ιδεών.

ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ

Ως κύριο σκοπό η μηχανική μάθηση έχει την μελέτη αναγνώρισης και κατανόησης μοτίβων από υπολογιστές. Καθιστά δυνατό, το να μαθαίνουν οι μηχανές δίχως να είναι απόλυτα προγραμματισμένες.

Αυτή η "αρχή" λειτουργεί με αλγόριθμους που μπορούν να δώσουν σχετικά ευρήματα ή συμπεράσματα που λαμβάνονται από ένα σύνολο δεδομένων, χωρίς ο άνθρωπος να γράφει οδηγίες ή κώδικες για αυτό. Ο σκοπός αυτής της "αρχής" είναι ότι οι άνθρωποι και οι μηχανές εργάζονται χέρι με χέρι. Ακριβώς αυτό κάνουν οι αλγόριθμοι, επιτρέπουν στις μηχανές να εκτελούν εργασίες, τόσο γενικές όσο και ειδικές.

Η διαδικασία εκμάθησης γίνεται από ταξινομητές. Ένας ταξινομητής είναι ένας αλγόριθμος που, λαμβάνοντας ως είσοδο ορισμένες πληροφορίες ενός αντικειμένου, είναι σε θέση να υποδείξει την κατηγορία ή την τάξη στην οποία ανήκει μεταξύ ενός περιορισμένου αριθμού πιθανών κλάσεων.

Διαλέξαμε 1 ταξινομητή από τις πιο αντιπροσωπευτικές οικογένειες αλγορίθμων για αξιολόγηση!

Multilayer Perceptron [MLP]

Το πολυστρωματικό perceptron είναι ένα τεχνητό νευρωνικό δίκτυο που σχηματίζεται από πολλαπλά στρώματα, με τέτοιο τρόπο ώστε να έχει την ικανότητα να επιλύει προβλήματα που δεν είναι γραμμικά διαχωρίσιμα, που είναι ο κύριος περιορισμός του perceptron[13] . Το πολυστρωματικό Perceptron μπορεί να συνδεθεί πλήρως ή τοπικά.

Ένα πολυστρωματικό perceptron έχει τρία στρώματα κόμβων: ένα στρώμα εισόδου, ένα κρυφό στρώμα και ένα στρώμα εξόδου. Εκτός από τον κόμβο εισόδου, κάθε κόμβος χρησιμοποιεί μια μη γραμμική συνάρτηση ενεργοποίησης, η οποία επιτρέπει την ταξινόμηση δεδομένων που δεν διαχωρίζονται γραμμικά. Το MLP χρησιμοποιεί μια τεχνική εποπτευόμενης μάθησης που ονομάζεται **back-propagation**.

Το **backpropagation** είναι μια μέθοδος που χρησιμοποιείται για τον υπολογισμό της κλίσης που είναι απαραίτητο να εφαρμοστεί στα βάρη ή στους συντελεστές των κόμβων στο δίκτυο. Ο αλγόριθμος είναι ο εξής:

- Ενεργοποίηση του μοντέλου δίνοντας μια τυχαία τιμή στα βάρη.
- Υπολογισμός των τιμών εξόδου.
- Υπολογισμός συνάρτησης απώλειας(λαμβανόμενη έξοδος ,λιγότερο επιθυμητή έξοδος.
- Αναστροφή των σφαλμάτων της εξόδου προς την είσοδο.
- Επανυπολογισμός των τιμών της εισόδου και επαναλαμβάνουμε τον αλγόριθμο εώς ότου σταθεροποιηθούν οι τιμές μας!

ΒΑΘΙΑ ΜΑΘΗΣΗ

Η βαθιά μάθηση είναι ένας κλάδος της τεχνητής νοημοσύνης που ασχολείται με τη μίμηση του τρόπου τον οποίο οι άνθρωποι χρησιμοποιούν για να αποκτήσουν ένα συγκεκριμένο είδος γνώσης. Είναι ένας τρόπος αυτοματοποίησης της προγνωστικής ανάλυσης. Βασίζεται κυρίως σε ένα βιολογικό μοντέλο του εγκεφάλου που προτάθηκε από τους βραβευμένους με Νόμπελ Hubel και Wiesel το 1959. Σχηματίζεται από στρώματα και νευρώνες που μαθαίνουν από τα κατώτερα στρώματα μέχρι τα ανώτερα με την ικανότητα να εξάγουν αφηρημένες έννοιες.

Οι αλγόριθμοι βαθιάς μάθησης έχουν μια αυξανόμενη ιεραρχία πολυπλοκότητας. Τι εννοούμε με αυτό.. Κάθε αλγόριθμος εφαρμόζει έναν μη γραμμικό μετασχηματισμό στην είσοδό του και χρησιμοποιεί αυτό που μαθαίνει για να δημιουργήσει ένα στατιστικό μοντέλο ως έξοδο. Υπάρχουν επαναλήψεις μέχρι η έξοδος να έχει ένα αποδεκτό επίπεδο ακρίβειας.

Στην βαθιά μάθηση, το μοντέλο μαθαίνει από μόνο του “ανακαλύπτοντας" τις σχέσεις μεταξύ των μεταβλητών! Οι άνθρωποι δεν παρεμβαίνουν μιας που το μοντέλο έχει την ικανότητα να κάνει την επιλογή των χαρακτηριστικών!

Μερικές φορές είναι περίπλοκο να εξάγουμε υψηλού-επιπέδου “αφηρημένα χαρακτηριστικά" από μη επεξεργασμένα δεδομένα, Η βαθιά μάθηση μας “λύνει τα χέρια" όσον αφορά αυτό το πρόβλημα με το να εκφράζει αυτά τα πολύπλοκα χαρακτηριστικά σε πιο απλούς συνδυασμούς αυτών.

Στην παρούσα εργασία, εστιάζουμε στα συνελικτικά νευρωνικά δίκτυα(CNN), εφόσον ο συγκεκριμένος τύπος δικτύου είναι ιδιαίτερα αποτελεσματικός όσον αφορά τον εντοπισμό εικόνων και την ταξινόμηση εργασιών!

Convolutiolan neural networks[CNN]

Σε αυτό το είδος βαθιάς μάθησης, χρησιμοποιούνται ειδικά νευρωνικά δίκτυα γνωστά ως συνελικτικά. Όπως όλα τα νευρωνικά δίκτυα, και αυτά έχουν στρώματα εισόδου, κρυμμένα επίπεδα και επίπεδα εξόδου, αλλά επίσης αυτού του είδους τα δίκτυα έχουν δύο τύπους κρυφών επιπέδων: στρώματα συνελεύσεων(convolution layers) και συγκέντρωσης(pooling layers).

Convolution layers

Αυτά τα κρυμμένα στρώματα ασχολούνται με την εκμάθηση τοπικών προτύπων σε μικρά δισδιάστατα παράθυρα. Είναι στρώματα που χρησιμεύουν στον εντοπισμό οπτικών χαρακτηριστικών σε εικόνες όπως γραμμές, άκρες και σταγόνες χρώματος. Αυτά τα στρώματα επιτρέπουν την εκμάθηση μιας ιδιότητας της εικόνας σε ένα συγκεκριμένο σημείο και είναι σε θέση να την αναγνωρίσουν οπουδήποτε στη δική τους εικόνα.

Τα συνελικτικά στρώματα είναι σε θέση να μάθουν διαφορετικά στοιχεία πιο πολύπλοκα σύμφωνα με όσα μαθαίνονταν στο προηγούμενο στρώμα. Ένα παράδειγμα θα ήταν ότι ένα στρώμα μαθαίνει διαφορετικούς τύπους γραμμών που εμφανίζονται στην εικόνα και το επόμενο στρώμα είναι σε θέση να μάθει στοιχεία της εικόνας που αποτελούνται από διαφορετικές γραμμές που έχουν μάθει στο προηγούμενο στρώμα. Η ικανότητα να το κάνει αυτό επιτρέπει στα δίκτυα αυτά να μαθαίνουν με αποτελεσματικό τρόπο οπτικές έννοιες που γίνονται όλο και πιο πολύπλοκες.

Τα συνελικτικά νευρωνικά δίκτυα λειτουργούν σε 3 άξονες, πλάτος, ύψος και το “κανάλι”. Στην είσοδο, το κανάλι θα μπορούσε να είναι 1 εάν η εικόνα εισόδου είναι σε γκρι κλίμακα ή 3 για έγχρωμη εικόνα RGB, μία για κάθε χρώμα (κόκκινο, πράσινο και μπλε). Μέσα στο δίκτυο, το βάθος είναι συνήθως μεγαλύτερο από 3.

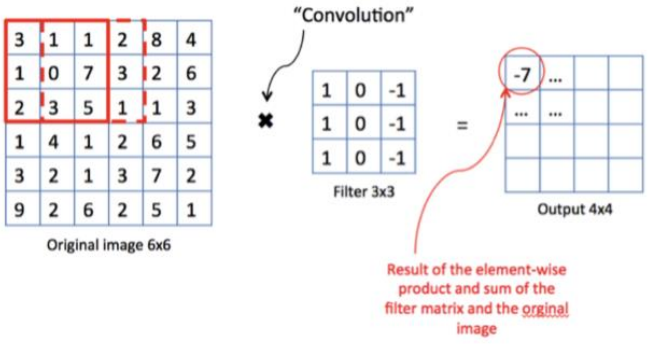
Δεν θα είναι όλοι οι νευρώνες εισόδου συνδεδεμένοι με όλους τους νευρώνες του συνελικτικού στρώματος . Γίνεται μέσω μικρών γειτονικών περιοχών του χώρου των νευρώνων εισόδου που αποθηκεύει τα pixel της εικόνας.

Η έξοδος είναι το αποτέλεσμα της εφαρμογής του φίλτρου Kernel στην αρχική εικόνα. Το αποτέλεσμα είναι το προϊόν και το άθροισμα της μήτρας kernel που εφαρμόζεται στην αρχική εικόνα σε κάθε διαφάνεια. Το μέγεθος εξόδου αλλάζει ανάλογα με το μέγεθος του kernel. Για παράδειγμα, το Σχήμα 3 δείχνει ένα φίλτρο που εντοπίζει οριζόντιες ακμές. Κάθε τιμή εξόδου είναι το αποτέλεσμα του προϊόντος και σύνοψη της αρχικής εικόνας με το φίλτρο, στην πρώτη διαφάνεια:

$$3 + 1 + 2 – 1 – 7 – 5 = -7$$

Ο έξοδος έχει μέγεθος 4x4 γιατί στην εικόνα εισόδου 6x6 μπορούμε να βάλουμε το φίλτρο 3x3 σε 4x4 θέσεις. Η μήτρα αρχίζει να ταξιδεύει μέσα από όλους τους νευρώνες στην πάνω αριστερή πλευρά και κινείται προς τα δεξιά. Όταν τελειώσουμε με μια σειρά, συνεχίζουμε με τη σειρά από κάτω.

Η εφαρμογή διαφορετικών φίλτρων είναι δυνατή για τον εντοπισμό διαφορετικών δυνατοτήτων.



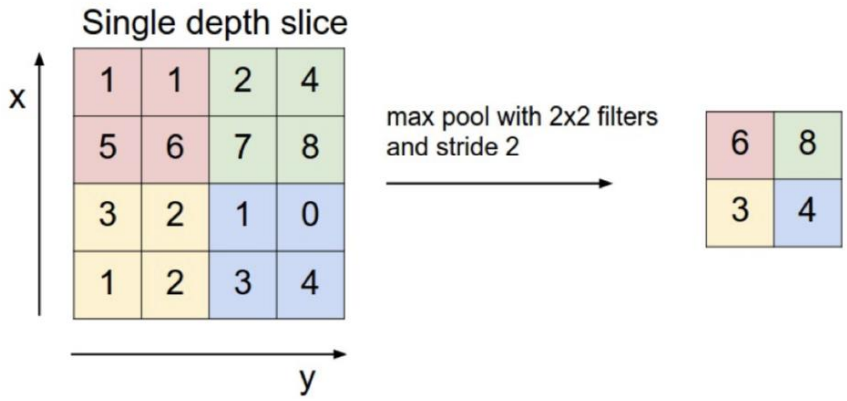
Σχήμα 3: Παράδειγμα συνέλιξης

Pooling layers

Ο συγκεκριμένος τύπος στρωμάτων συγκεντρώνει τους νευρώνες σε ομάδες. Με την διαδικασία αυτή αποκτάμε τις πιο σχετικές ιδιότητες των δεδομένων εισόδου και εξαλείφει τα περιττά δεδομένα που μπορεί να είναι παραπλανητικά.

Υπάρχουν διάφορες μορφές συγκέντρωσης. Μια "μέση συγκέντρωση" που λαμβάνει τη μέση τιμή της ομάδας των νευρώνων ή αλλιώς "μέγιστη συγκέντρωση" που επιλέγει τη μέγιστη τιμή κάθε ομάδας.

Σε αυτό το στρώμα, όταν ομαδοποιούνται οι νευρώνες, η είσοδος μειώνεται. Μειώνεται ανάλογα με το μέγεθος των ομαδοποιήσεων. Για παράδειγμα, εάν οι ομάδες έχουν μέγεθος 2x2, στην έξοδο θα λάβουμε το ένα τέταρτο του μεγέθους των δεδομένων εισόδου. Στο σχήμα 4 φαίνεται ότι, ακόμη και με αυτή η τεχνική μετασχηματισμού, η "χωρική" σχέση εξακολουθεί να διατηρείται.



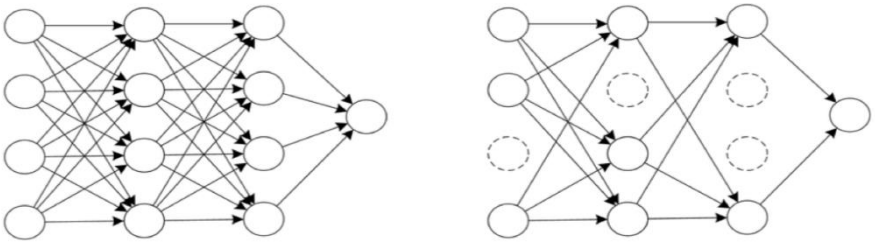
Σχήμα 4: Παράδειγμα Pooling

Μείωση overfitting : "Dropout"

Οι αρχιτεκτονικές βαθιάς μάθησης θα μπορούσαν να έχουν εκατομμύρια παραμέτρους, μιας που κάθε στρώμα συνδέεται με συνδέσεις βάρους. Αυτά τα μοντέλα με πολλές παραμέτρους μπορούν να εμφανίσουν εύκολα overfit στα δεδομένα εκπαίδευσης. Για να αποφύγουμε αυτό το πρόβλημα, υπάρχουν ορισμένες μέθοδοι που βοηθούν στη μείωση του.

Μία απ' αυτές τις μεθόδους είναι το dropout, ο όρος αναφέρεται στην εγκατάλειψη μονάδων σε ένα νευρωνικό δίκτυο όπως φαίνεται στο σχήμα 5. Η πτώση των μονάδων αφαιρεί τους νευρώνες από το δίκτυο, συμπεριλαμβανομένων των συνδέσεων βαρών τους. Το Dropout χρησιμοποιεί μια σταθερή πιθανότητα για να διατηρηθεί , ενώ η επιλογή των μονάδων που θα απορριφθούν είναι τυχαία.

Τα νευρωνικά δίκτυα που χρησιμοποιούν την μέθοδο dropout μπορούν να εκπαιδευτούν χρησιμοποιώντας μια προσαρμοστική μέθοδο εκμάθησης. Η διαφορά είναι ότι για κάθε εκπαίδευση μιας μίνι παρτίδας, η προώθηση και η αναστροφή(backpropagation) γίνονται σε ένα αραιωμένο δίκτυο που πρέπει να εγκαταλείψει μονάδες.



Σχήμα 5 : σχέδιο απόδοσης dropout

Συναρτήσεις ενεργοποίησης

Στα νευρωνικά δίκτυα, η συνάρτηση ενεργοποίησης ενός κόμβου καθορίζει την έξοδο ενός κόμβου ανάλογα με την είσοδο ή το σύνολο εισόδων. Για κάθε νευρώνα, η είσοδος υπολογίζεται με τις συναρτήσεις ενεργοποίησης και στη συνέχεια το αποτέλεσμα αποστέλλεται ως έξοδος. Ανάλογα με αυτήν την έξοδο, η συνάρτηση ενεργοποίησης αποφασίζει εάν οι συνδέσεις αυτού του νευρώνα θα ενεργοποιοιηθούν ή όχι. Σε αυτήν την εργασία, έχει χρησιμοποιηθεί ο τύπος συνάρτησης ενεργοποίησης "διορθωμένης γραμμικής μονάδας (Rectified linear units) , άλλα θα αναφέρουμε και έναν ακόμα που χρησιμοποιείται το ίδιο πολύ την σιγμοειδή (Sigmoid)!

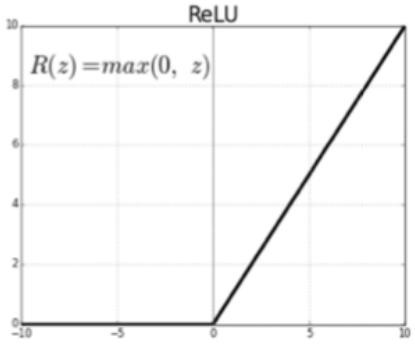
Rectified Linear Units [ReLU]

Τα τελευταία χρόνια, οι διορθωμένες γραμμικές μονάδες (ReLU) αποτελούν την πιο χρησιμοποιούμενη συνάρτηση ενεργοποίησης χάρη στην απλότητάς τους. Η ReLU έχει αποδειχθεί ότι κάνει την εκπαίδευση γρηγορότερα σε σύγκριση με άλλες συναρτήσεις ενεργοποίησης. Η συνάρτηση ενεργοποίησης ReLU επιστρέφει 0 αν η έξοδος είναι μικρότερη από 0 και επιστρέφει την ίδια τιμή της εισόδου εάν η είσοδος είναι μεγαλύτερη από 0.

Αν και μπορεί να το λέει το όνομά της, αυτή η συνάρτηση ενεργοποίησης δεν είναι γραμμική, αφού για όλες τις αρνητικές τις τιμές μετατοπίζεται στο 0. Η εξίσωση που την αντιπροσωπεύει είναι η ακόλουθη.

$$f(x) = max(0, x)$$

Και η γραφική τους αναπαράσταση φαίνεται στο παρακάτω σχήμα:



Σχήμα 6: Γραφική αναπαράσταση ReLU

Sigmoid

Η μη γραμμική φύση της σιγμοειδούς συνάρτησης είναι ιδανική για νευρωνικά δίκτυα. Η σιγμοειδής συνάρτηση μετατρέπει τις εισαγόμενες τιμές σε μια κλίμακα (0,1), όπου οι υψηλές τιμές έχουν τείνουν ασυμπτωτικά στο 1 και οι πολύ χαμηλές τιμές τείνουν ασυμπτωτικά στο 0. Η εξίσωση της είναι η ακόλουθη:

$$Accuracy = \frac{1}{1 - e^{-x}}$$

Τα χαρακτηριστικά της είναι :

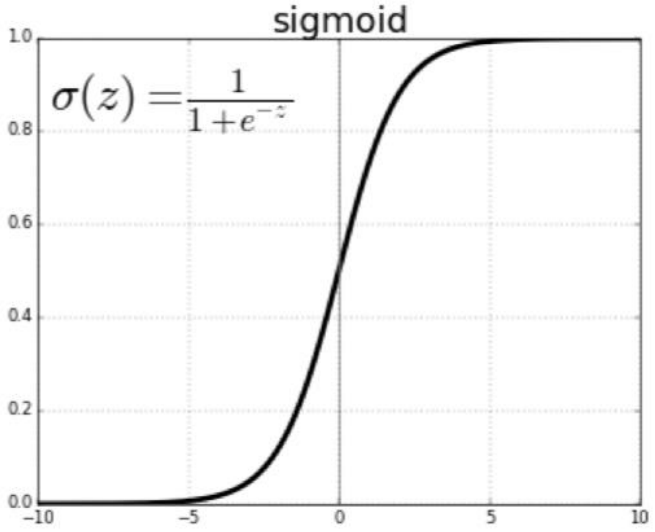
i) Αργή σύγκλιση.

ii) Δεν έχει ως κέντρο το 0.

iii) Είναι οριοθετημένη μεταξύ του 0 και του 1.

iv) Έχει καλή απόδοση στο τελευταίο στρώμα.

Γενικά αποκαλείται Σιγμοειδής(sigmoid) μιας που σχηματίζεται, όπως θα δούμε και στη γραφική παρακάτω, ένα S.

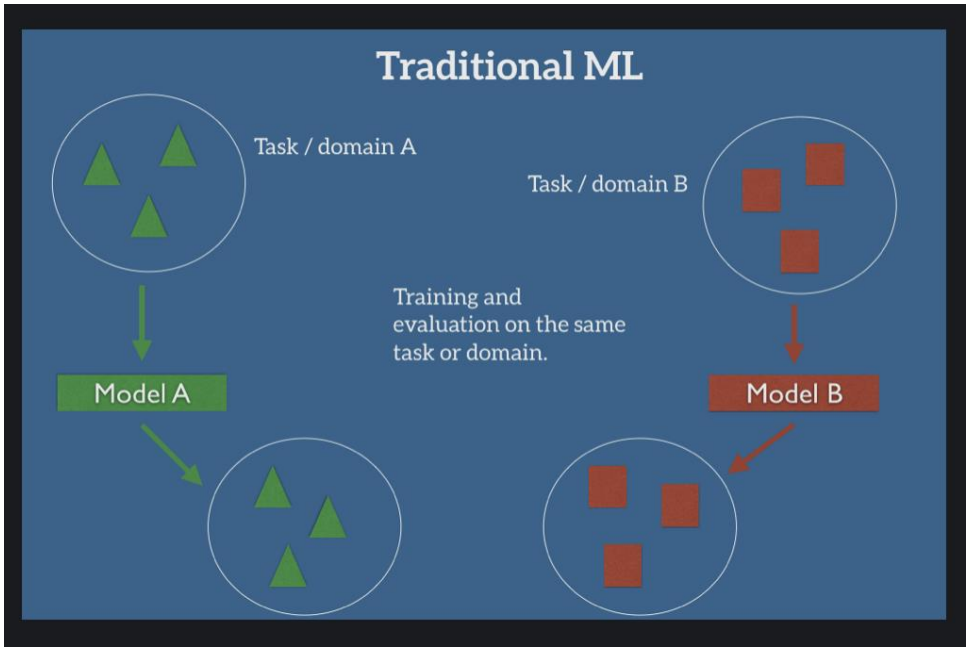


Σχήμα 7 : Γραφική αναπαράσταση σιγμοειδής συνάρτησης

TRANSFER LEARNING

Τί εστί transfer learning;

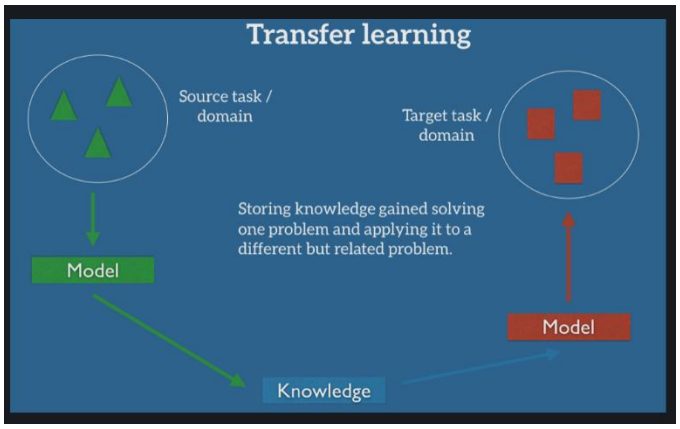
Στο κλασικό σενάριο εποπτευόμενης μάθησης (της μηχανικής μάθησης), εάν σκοπεύουμε να εκπαιδεύσουμε ένα μοντέλο για κάποια εργασία και τομέα Α, υποθέτουμε ότι μας παρέχονται δεδομένα με ετικέτα για την ίδια εργασία και τομέα. Αυτό μπορούμε να το δούμε καθαρά στο Σχήμα 8, όπου η εργασία και ο τομέας των δεδομένων εκπαίδευσης και δοκιμών του μοντέλου μας Α είναι ίδια . Αργότερα θα ορίσουμε λεπτομερέστερα τι ακριβώς είναι μια εργασία και ένας τομέας). Προς το παρόν, ας υποθέσουμε ότι μια εργασία είναι ο στόχος που επιδιώκει να επιτύχει το μοντέλο μας, π.χ. αναγνωρίζουν αντικείμενα σε εικόνες και ένας τομέας είναι από όπου προέρχονται τα δεδομένα μας, π.χ. εικόνες που ελήφθησαν στα καφενεία του Σαν Φρανσίσκο[7].



Σχήμα 8: Παραδοσιακό setup εποπτευόμενης μάθησης στην ΜΜ.

Τώρα μπορούμε να εκπαιδεύσουμε ένα μοντέλο Α σε αυτό το σύνολο δεδομένων και περιμένουμε να έχει καλή απόδοση σε "αόρατα" δεδομένα της ίδιας εργασίας και τομέα. Σε άλλη περίσταση, όταν δίνονται δεδομένα για κάποια άλλη εργασία ή τομέα Β, απαιτούμε ξανά δεδομένα με ετικέτα της ίδιας εργασίας ή τομέα που μπορούμε να χρησιμοποιήσουμε για να εκπαιδεύσουμε ένα νέο μοντέλο Β έτσι ώστε να μπορούμε να περιμένουμε να έχει καλή απόδοση σε αυτά τα δεδομένα.Το παραδοσιακό υπόδειγμα εποπτευόμενης μάθησης καταρρέει όταν δεν έχουμε επαρκή δεδομένα με ετικέτα για την εργασία ή τον τομέα που μας ενδιαφέρει να εκπαιδεύσουμε ένα αξιόπιστο μοντέλο. Αν θέλουμε να εκπαιδεύσουμε ένα μοντέλο για τον εντοπισμό πεζών σε νυχτερινές εικόνες, θα μπορούσαμε να εφαρμόσουμε ένα μοντέλο που έχει εκπαιδευτεί σε παρόμοιο τομέα, π.χ. στις εικόνες της ημέρας. Στην πράξη, ωστόσο, βιώνουμε συχνά επιδείνωση ή κατάρρευση της απόδοσης καθώς το μοντέλο έχει κληρονομήσει την προκατάληψη των δεδομένων εκπαίδευσης και δεν ξέρει πώς να γενικευτεί στον νέο τομέα[7].

Αν θέλουμε να εκπαιδεύσουμε ένα μοντέλο για να εκτελέσει μια νέα εργασία, όπως ανίχνευση ποδηλατών, δεν μπορούμε καν να επαναχρησιμοποιήσουμε ένα υπάρχον μοντέλο, καθώς οι ετικέτες μεταξύ των εργασιών διαφέρουν. Το Transfer learning μας επιτρέπει να αντιμετωπίσουμε αυτά τα σενάρια αξιοποιώντας τα ήδη υπάρχοντα επισημασμένα δεδομένα κάποιας σχετικής εργασίας ή τομέα. Προσπαθούμε να αποθηκεύσουμε αυτήν τη γνώση που αποκτήθηκε για την επίλυση της εργασίας προέλευσης στον τομέα προέλευσης(πηγής) και την εφαρμόζουμε στο πρόβλημα που μας ενδιαφέρει, όπως φαίνεται στο σχήμα 9.



Σχήμα 9 : Transfer learning setup

Στην πράξη, επιδιώκουμε να μεταφέρουμε όση περισσότερη γνώση μπορούμε από τη ρύθμιση της πηγής στο στόχο ή τον τομέα μας. Αυτή η γνώση μπορεί να λάβει διάφορες μορφές ανάλογα με τα δεδομένα: μπορεί να σχετίζεται με τον τρόπο σύνθεσης των αντικειμένων για να μας επιτρέπει να εντοπίζουμε ευκολότερα νέα αντικείμενα. μπορεί να αφορά τις γενικές λέξεις που χρησιμοποιούν οι άνθρωποι για να εκφράσουν τις απόψεις τους κ.λπ.

FACENET

Η αναγνώριση προσώπου είναι η εργασία αναγνώρισης και επαλήθευσης ατόμων με βάση τις εικόνες προσώπου. Το FaceNet είναι ένα σύστημα αναγνώρισης προσώπου που αναπτύχθηκε το 2015 από τους ερευνητές της Google Florian Schroff, Dmitry Kalenichenko και James Philbin σε μια εργασία με τίτλο FaceNet: A Unified Embedding for Face Recognition and Clustering.

Αναγνώριση προσώπου χρησιμοποιώντας το μοντέλο FaceNet

Η αναγνώριση προσώπου είναι η διαδικασία αναγνώρισης ενός ατόμου από μια ψηφιακή εικόνα ή ένα βίντεο. Αυτό είναι ένα πρόβλημα αντιστοίχισης 1: K. Μια περίπτωση χρήσης για αυτό θα μπορούσε να είναι η επισήμανση της παρουσίας των εργαζομένων όταν ένας εργαζόμενος εισέρχεται στο κτίριο αναζητώντας τις κωδικοποιήσεις προσώπου τους στη βάση δεδομένων.

Το μοντέλο FaceNet αναμένει μια είσοδο μεγέθους 160x160x3 ως είσοδο και εξάγει ένα διάνυσμα ενσωμάτωσης προσώπου με μήκος 128. Αυτή η ενσωμάτωση προσώπου περιέχει πληροφορίες που περιγράφουν τα σημαντικά χαρακτηριστικά ενός προσώπου. Στη συνέχεια, το FaceNet βρίσκει την ετικέτα τάξης της ενσωμάτωσης του προσώπου κατάρτισης που έχει την ελάχιστη απόσταση L2 με την ενσωμάτωση της όψης -στόχου. Είναι η μικρότερη απόσταση μεταξύ δύο σημείων σε ένα N διαστατικό χώρο γνωστό και ως Ευκλείδειο χώρο. Το μοντέλο CNN χρησιμοποιεί τη συνάρτηση Triplet Loss. Αυτή η λειτουργία επιστρέφει μια μικρότερη τιμή για παρόμοιες εικόνες και μεγαλύτερη τιμή για ανόμοιες εικόνες. Τα στοιχεία ενός δικτύου FaceNet περιγράφονται στις ακόλουθες ενότητες.

ΕΙΣΑΓΩΓΗ ΕΙΚΟΝΑΣ

Το σετ εκπαίδευσης αποτελείται από μικρογραφίες προσώπων “κομμένες” σε μέγεθος 160x160 από τις εικόνες. Εκτός από τη μετάφραση και την κλιμάκωση, δεν απαιτούνται άλλες ευθυγραμμίσεις με τα “κομμένα" πρόσωπα.

DEEP CNN

Υπάρχουν δύο διαφορετικές αρχιτεκτονικές που περιγράφονται στο FaceNet:Deep CNN Βασισμένο στην Αρχιτεκτονική Zeiler και Fergus NetworkΑρχή αρχιτεκτονικής μοντέλου βασισμένο στο GoogLeNetΟι δύο αρχιτεκτονικές διαφέρουν ως προς τον αριθμό των παραμέτρων που χρησιμοποιούνται και τις λειτουργίες πλωτού σημείου ανά δευτερόλεπτο (FLOPS). Το FLOPS είναι ένα τυπικό μέτρο απόδοσης του υπολογιστή που απαιτεί υπολογισμούς κυμαινόμενου σημείου. Η ακρίβεια του μοντέλου είναι μεγαλύτερη με μεγαλύτερα FLOPS. Ένα δίκτυο με χαμηλότερα FLOPS λειτουργεί πιο γρήγορα και καταναλώνει λιγότερη μνήμη, αλλά έχει χαμηλότερη ακρίβεια.

NN1: Deep CNN Βασισμένο στην αρχιτεκτονική του δικτύου Zeiler και Fergus

layer	size-in	size-out	kernel	param	FLPS
conv1	220×220×3	110×110×64	7×7×3, 2	9K	115M
pool1	110×110×64	55×55×64	3×3×64, 2	0	
rnorm1	55×55×64	55×55×64		0	
conv2a	55×55×64	55×55×64	1×1×64, 1	4K	13M
conv2	55×55×64	55×55×192	3×3×64, 1	111K	335M
rnorm2	55×55×192	55×55×192		0	
pool2	55×55×192	28×28×192	3×3×192, 2	0	
conv3a	28×28×192	28×28×192	1×1×192, 1	37K	29M
conv3	28×28×192	28×28×384	3×3×192, 1	664K	521M
pool3	28×28×384	14×14×384	3×3×384, 2	0	
conv4a	14×14×384	14×14×384	1×1×384, 1	148K	29M
conv4	14×14×384	14×14×256	3×3×384, 1	885K	173M
conv5a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv5	14×14×256	14×14×256	3×3×256, 1	590K	116M
conv6a	14×14×256	14×14×256	1×1×256, 1	66K	13M
conv6	14×14×256	14×14×256	3×3×256, 1	590K	116M
pool4	14×14×256	7×7×256	3×3×256, 2	0	
concat	7×7×256	7×7×256		0	
fc1	7×7×256	1×32×128	maxout p=2	103M	103M
fc2	1×32×128	1×32×128	maxout p=2	34M	34M
fc7128	1×32×128	1×1×128		524K	0.5M
L2	1×1×128	1×1×128		0	
total				140M	1.6B

NN2: Αρχή αρχιτεκτονικής μοντέλου με βάση το GoogleNet

type	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj (p)	params	FLOPS
conv1 (7×7×3, 2)	112×112×64	1							9K	119M
max pool + norm	56×56×64	0						m 3×3, 2		
inception (2)	56×56×192	2		64	192				115K	360M
norm + max pool	28×28×192	0						m 3×3, 2		
inception (3a)	28×28×256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28×28×320	2	64	96	128	32	64	L_2, 64p	228K	179M
inception (3c)	14×14×640	2	0	128	256,2	32	64,2	m 3×3,2	398K	108M
inception (4a)	14×14×640	2	256	96	192	32	64	L_2, 128p	545K	107M
inception (4b)	14×14×640	2	224	112	224	32	64	L_2, 128p	595K	117M
inception (4c)	14×14×640	2	192	128	256	32	64	L_2, 128p	654K	128M
inception (4d)	14×14×640	2	160	144	288	32	64	L_2, 128p	722K	142M
inception (4e)	7×7×1024	2	0	160	256,2	64	128,2	m 3×3,2	717K	56M
inception (5a)	7×7×1024	2	384	192	384	48	128	L_2, 128p	1.6M	78M
inception (5b)	7×7×1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1×1×1024	0								
fully conn	1×1×128	1							131K	0.1M
L2 normalization	1×1×128	0								
total									7.5M	1.6B

Ενσωμάτωση προσώπου

Οι ενσωματώσεις προσώπου μεγεθών 1 × 1 × 128 δημιουργούνται από το στρώμα κανονικοποίησης L2 του βαθιού CNN. Αυτή η ενσωμάτωση χρησιμοποιείται στην επαλήθευση προσώπου και στην ομαδοποίηση προσώπου.

Triplet Loss

Οι ενσωματώσεις των ίδιων προσώπων ονομάζονται θετικές και οι ενσωματώσεις διαφορετικών προσώπων ονομάζονται αρνητικές. Το πρόσωπο που αναλύεται ονομάζεται άγκυρα. Για τον υπολογισμό της απώλειας, σχηματίζεται ένα τρίδυμο που αποτελείται από μια άγκυρα, μια θετική και μια αρνητική ενσωμάτωση και αναλύονται οι Ευκλείδειες αποστάσεις τους. Ο μαθησιακός στόχος του FaceNet είναι να ελαχιστοποιήσει την απόσταση μεταξύ μιας άγκυρας και ενός θετικού και να μεγιστοποιήσει την απόσταση μεταξύ της άγκυρας και ενός αρνητικού. Ένα εκπαιδευτικό τρίδυμο περιέχει τρία δείγματα: άγκυρα, θετικά και αρνητικά (A, P, N). Ο στόχος του δικτύου ενσωμάτωσης τριπλής απώλειας είναι να μάθει μια ενσωμάτωση έτσι ώστε

$$(||F(A) - F(P)|| + margin) < ||F(A) - F(N)||$$

$$\sum_{i}^{N}[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha$$

$$x_i - Itrepresentsanimage$$

$$f(x_i) - Itrepresentstheembeddingofanimage$$

$$\alpha - Itrepresentsthemarginbetweenpositiveandnegativepairs$$

Triplet Selection

Η επιλογή των σωστών ζευγών εικόνων είναι εξαιρετικά σημαντική καθώς θα υπάρχουν πολλά ζεύγη εικόνων που θα ικανοποιούν αυτήν την προϋπόθεση. Το μοντέλο δεν θα μάθει πολλά από αυτά και θα συγκλίνει επίσης αργά εξαιτίας αυτού. Για να διασφαλιστεί η γρήγορη σύγκλιση, είναι ζωτικής σημασίας να επιλέξετε τρίδυμα που παραβιάζουν τον περιορισμό του τριπλού.

$$Argmax||f(x_i^a) - f(x_i^p)||_2^2 - Eqn(1)$$

$$Argmin||f(x_i^a) - f(x_i^n)||_2^2 - Eqn(2)$$

Eq (1) σημαίνει ότι δεδομένης μιας εικόνας άγκυρας του ατόμου A, θέλουμε να βρούμε μια θετική εικόνα του A έτσι ώστε η απόσταση μεταξύ αυτών των δύο εικόνων να είναι μεγαλύτερη. Eq (2) σημαίνει ότι δεδομένης μιας εικόνας αγκύρωσης A, θέλουμε να βρούμε μια αρνητική εικόνα έτσι ώστε η απόσταση μεταξύ αυτών των δύο εικόνων να είναι η μικρότερη. Έτσι, απλώς επιλέγουμε τα σκληρά θετικά και τα αρνητικά εδώ. Αυτή η προσέγγιση μας βοηθά να επιταχύνουμε τη σύγκλιση καθώς το μοντέλο μας μαθαίνει χρήσιμες αναπαραστάσεις. Οι εφευρέτες του FaceNet χρησιμοποίησαν μίνι παρτίδες αποτελούμενες από 40 θετικές και τυχαία επιλεγμένες αρνητικές ενσωματώσεις. Οι ελάχιστες και οι μέγιστες αποστάσεις υπολογίστηκαν για κάθε μίνι παρτίδα για να δημιουργηθούν τρίδυμα.

Επαλήθευση προσώπου επαλήθευση προσώπου συγκρίνει τις ενσωματώσεις προσώπου όλων των εκπαιδευμένων εικόνων με τη δεδομένη εικόνα για να βρείτε αντιστοιχίσεις. Η εύρεση αν δύο εικόνες ανήκουν στο ίδιο άτομο είναι αντιστοίχιση 1: 1.

Ομαδοποίηση προσώπου

Η ομαδοποίηση προσώπων είναι η διαδικασία ομαδοποίησης εικόνων του ίδιου ατόμου μαζί για άλμπουμ. Απαντά στην ερώτηση "Υπάρχουν παρόμοια πρόσωπα;" Οι ενσωματώσεις προσώπων μπορούν να εξαχθούν και ένας αλγόριθμος ομαδοποίησης όπως τα K-μέσα μπορεί να χρησιμοποιηθεί για να ομαδοποιήσει τα πρόσωπα του ίδιου ατόμου μαζί.

Συμπέρασμα

Τα ορόσημα του προσώπου μπορούν να χρησιμοποιηθούν για την ευθυγράμμιση του, για παρακολούθηση προσώπων σε βίντεο και για μέτρηση συναισθημάτων. Μπορούν επίσης να βοηθήσουν στη διάγνωση ιατρικών καταστάσεων. . Το FaceNet μπορεί να αναγνωρίσει πρόσωπα ακόμη και αν το άτομο φορά χειρουργική μάσκα

4)ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΠΕΡΙΒΑΛΛΟΝΤΑ ΑΝΑΠΤΥΞΗΣ

TensoFlow

Το TensorFlow είναι μια end-to-end βιβλιοθήκη(πλατφόρμα) ανοικτού κώδικα που χρησιμοποιείται για μηχανική μάθηση. Διαθέτει ένα ολοκληρωμένο, ευέλικτο οικοσύστημα εργαλείων, βιβλιοθηκών και κοινοτικών πόρων που επιτρέπει στους ερευνητές να προωθήσουν την τελευταία λέξη της τεχνολογίας και οι προγραμματιστές να δημιουργήσουν και να αναπτύξουν εύκολα εφαρμογές που λειτουργούν με μηχανική μάθηση.

Το TensorFlow αναπτύχθηκε αρχικά από ερευνητές και μηχανικούς που εργάζονται στην ομάδα του Google Brain στον οργανισμό Google Machine Intelligence Research για τη διεξαγωγή μηχανικής μάθησης και έρευνας για νευρωνικά δίκτυα. Το σύστημα είναι αρκετά γενικό ώστε να μπορεί να εφαρμοστεί σε μεγάλη ποικιλία άλλων τομέων.Το TensorFlow παρέχει σταθερά API Python και C ++, καθώς και μη εγγυημένο API (backward compatible) για άλλες γλώσσες.

Η χρήση αυτής της βιβλιοθήκης μας επιτρέπει την γρήγορη υλοποίηση μαθηματικών τύπων και αλγόριθμων που χρησιμοποιούνται στον τομέα των συνελικτικών νευρωνικών δικτύων, με αποτέλεσμα τον σχεδιασμό, την κατασκευή και την εκπαίδευση μοντέλων βαθιάς μάθησης. Οι υπολογισμοί γίνονται με γραφήματα ροής δεδομένων (data flow graphs). Τα γραφήματα αυτά αποτελούνται από κόμβους και ακμές, όπου οι κόμβοι είναι μαθηματικές λειτουργίες ενώ οι ακμές είναι τα δεδομένα. Τα δεδομένα είναι συνήθως τανυστές (tensors)[8].

Σχήμα 10: TensorFlow logo

Keras

Το Keras είναι ένα API βαθιάς μάθησης γραμμένο σε Python, που “τρέχει” πάνω από την πλατφόρμα μηχανικής μάθησης TensorFlow. Αναπτύχθηκε με έμφαση στη δυνατότητα γρήγορου πειραματισμού. Το να μπορείς να πηγαίνεις από την ιδέα στο αποτέλεσμα όσο το δυνατόν γρηγορότερα είναι το κλειδί για να κάνεις καλή έρευνα[8].

Το Κεράς είναι:

- Απλό - αλλά όχι απλοϊκό.
- Μειώνει το γνωστικό φορτίο του προγραμματιστή για να σας αφήσει να εστιάσετε στα μέρη του προβλήματος που έχουν πραγματικά σημασία.
- Είναι ευέλικτο.
- Υιοθετεί την αρχή της προοδευτικής αποκάλυψης της πολυπλοκότητας: οι απλές ροές εργασίας πρέπει να είναι γρήγορες και εύκολες, ενώ οι αυθαίρετα προηγμένες ροές εργασίας θα πρέπει να είναι δυνατές μέσω μιας σαφούς πορείας που βασίζεται σε όσα έχουμε ήδη μάθει.
- Είναι ισχυρό.
- Παρέχει απόδοση και επεκτασιμότητα στη βιομηχανία: χρησιμοποιείται από οργανισμούς και εταιρείες όπως η NASA, το YouTube ή το Waymo.

Σχήμα 11: Keras logo

Anaconda-Jupyter Notebook

Η εφαρμογή Jupyter Notebook μάς επιτρέπει να δημιουργούμε και να επεξεργαζόμαστε έγγραφα που εμφανίζουν την είσοδο και την έξοδο ενός σεναρίου Python ή R γλώσσας. Μόλις αποθηκευτούν, μπορουμέ να μοιραστούμε αυτά τα αρχεία και με άλλους. ΣΗΜΕΙΩΣΗ: Η γλώσσα Python και R περιλαμβάνονται από προεπιλογή, αλλά με προσαρμογή, το Notebook μπορεί να εκτελέσει πολλά άλλα περιβάλλοντα πυρήνα[14]. Ο κώδικας της διπλωματικής γράφτηκε αποκλειστικά στην συγκεκριμένη πλατφόρμα.

Σχήμα 12: Jupyter Notebook logo

Deepface

Το Deepface είναι ένα ελαφρύ πλαίσιο αναγνώρισης προσώπου και ανάλυσης χαρακτηριστικών προσώπου (ηλικία, φύλο, συναίσθημα και φυλή) για την python. Πρόκειται για ένα υβριδικό πλαίσιο αναγνώρισης προσώπου που συνδυάζει μοντέρνα μοντέλα: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace και Dlib. Αυτά τα μοντέλα έφτασαν και πέρασαν την ακρίβεια ανθρώπινου επιπέδου[9]. Η βιβλιοθήκη βασίζεται κυρίως στα TensorFlow και Keras.

Σχήμα 13: Deepface logo

OpenCV

Το OpenCV είναι μία βιβλιοθήκη ανοικτού κώδικα που χρησιμοποιείται κυρίως στην μηχανική μάθηση και την όραση υπολογιστών. Παρέχει υποδομές και επιταχύνει την αναγνώριση εικόνων και αντικειμένων σε πραγματικό χρόνο. Οι αλγόριθμοι που περιλαμβάνονται σε αυτήν την βιβλιοθήκη επιτρέπουν – μεταξύ άλλων – την αναγνώριση προσώπου και αντικειμένων, αφαίρεση κόκκινων ματιών, παρακολούθηση της κίνησης του ματιού και γενικότερα παρακολούθηση κίνησης αντικειμένων[15].

Σχήμα 14: OpenCV logo

ΣΕΤ ΔΕΔΟΜΕΝΩΝ [DATASET]

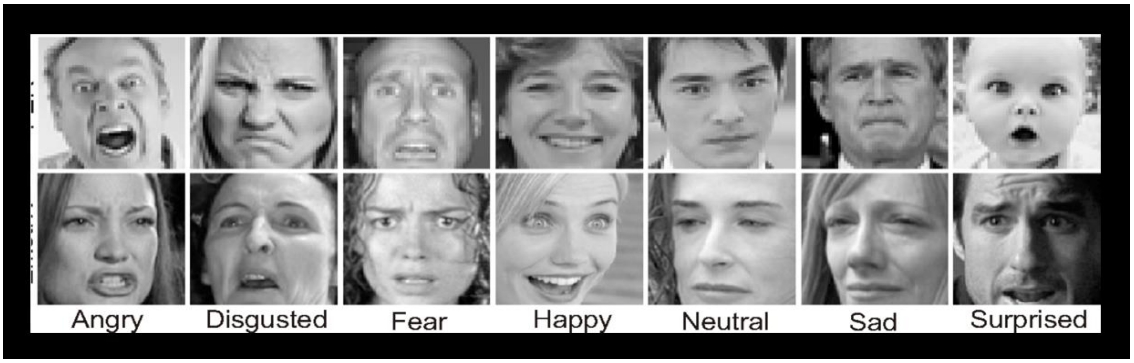
Για να μπορέσουμε να εκπαιδεύσουμε το μοντέλο, είναι απαραίτητο να έχουμε σετ δεδομένων που έχει καλά ταξινομημένα τα συναισθήματα. Επίσης σημαντικό είναι να έχει μεγάλο αριθμό δεδομένων έτσι ώστε να μπορέσουμε να εκπαιδεύσουμε με βέλτιστο τρόπο το μοντέλο. Παρακάτω θα παρουσιάσουμε το σετ δεδομένων FER2013 που χρησιμοποιήθηκε κατά την διάρκεια εκπόνησης της εργασίας.

FER2013

Τα δεδομένα αποτελούνται από εικόνες 48x48 pixel σε κλίμακα του γκρι προσώπων. Τα πρόσωπα έχουν καταχωρηθεί αυτόματα έτσι ώστε το πρόσωπο να βρίσκεται στο κέντρο και να καταλαμβάνει περίπου τον ίδιο χώρο σε κάθε εικόνα.

Η βάση δεδομένων δημιουργήθηκε χρησιμοποιώντας το API αναζήτησης εικόνας Google και τα πρόσωπα καταχωρήθηκαν αυτόματα, κάθε εικόνα ταξινομείται ανάλογα με το συναίσθημα που εμφανίζεται στην έκφραση του προσώπου σε μία από τις επτά κατηγορίες του Κατηγορικού μοντέλου.

Παρακάτω ακολουθεί παράδειγμα!



Σχήμα 15: Παράδειγμα FER2013

5)ΣΧΕΔΙΑΣΜΟΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ

Στο συγκεκριμένο κεφάλαιο θα αναλύσουμε την αρχιτεκτονική των μοντέλων της εργασίας, τον κώδικα των 3 εφαρμογών /project μου , καθώς και το πως λειτουργούν!

I) Alex-CNN-model / EDA

Στο 1ο project, χρησιμοποιώντας το σετ δεδομένων FER2013 , επιχείρησα να κάνω μια επεξηγηματική ανάλυση (EDA) όσον αφορά την αναγνώριση συναισθημάτων μέσω ενός συνελικτικού δικτύου. Παρακάτω θα αναλυθεί βήμα βήμα ο κώδικας , καθώς και τα στάδια της ανάλυσης!

ΒΙΒΛΙΟΘΗΚΕΣ

In [1]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
```

In [5]:
```
pip install mlxtend
```

```
Requirement already satisfied: mlxtend in /opt/anaconda3/lib/python3.8/site-packages (0.18.0)
Requirement already satisfied: numpy>=1.16.2 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (1.18.5)
Requirement already satisfied: matplotlib>=3.0.0 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (3.2.2)
Requirement already satisfied: scikit-learn>=0.20.3 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (0.2
3.1)
Requirement already satisfied: joblib>=0.13.2 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (0.16.0)
Requirement already satisfied: setuptools in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (49.2.0.post20
200714)
Requirement already satisfied: pandas>=0.24.2 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (1.0.5)
Requirement already satisfied: scipy>=1.2.1 in /opt/anaconda3/lib/python3.8/site-packages (from mlxtend) (1.5.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /opt/anaconda3/lib/python3.8/site-packages
(from matplotlib>=3.0.0->mlxtend) (2.4.7)
Requirement already satisfied: cycler>=0.10 in /opt/anaconda3/lib/python3.8/site-packages (from matplotlib>=3.0.0->ml
xtend) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /opt/anaconda3/lib/python3.8/site-packages (from matplotlib>=3.0.
0->mlxtend) (1.2.0)
Requirement already satisfied: python-dateutil>=2.1 in /opt/anaconda3/lib/python3.8/site-packages (from matplotlib>=3
.0.0->mlxtend) (2.8.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/anaconda3/lib/python3.8/site-packages (from scikit-learn>
=0.20.3->mlxtend) (2.1.0)
Requirement already satisfied: pytz>=2017.2 in /opt/anaconda3/lib/python3.8/site-packages (from pandas>=0.24.2->mlxte
nd) (2020.1)
Requirement already satisfied: six in /opt/anaconda3/lib/python3.8/site-packages (from cycler>=0.10->matplotlib>=3.0.
0->mlxtend) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

In [6]:
```
from keras import models
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from keras.optimizers import RMSprop, Adam
from tensorflow.keras.utils import to_categorical
```

Αρχικά φορτώνουμε τις στάνταρ και απαραίτητες βιβλιοθήκες(pandas, numpy, matplotlib) και πακέτα της Keras και sklearn. Επειδή αντιμετώπισα ένα πρόβλημα με την φόρτωση των πακέτων Keras ήταν αναγκαστικό πριν να τρέξω την εντολή " pip install mlxtend"!

PATH

Στην συνέχεια ορίζουμε το μονοπάτι εισόδου, την “διεύθυνση" που βρίσκονται δηλαδή τα αρχεία μας και δείχνουμε ποια είναι.

```
In [7]: path = '/Users/user/Desktop/facial expression cnn 2.0/'
        os.listdir(path)

Out[7]: ['resnet50_weights_tf_dim_ordering_tf_kernels.h5',
         'icml_face_data.csv',
         'mobilenet_1_0_224_tf_no_top.h5',
         'model_weights_resnet.h5',
         'test.csv',
         'efficientnetb3_notop.h5',
         'mask_rcnn_coco.h5',
         'resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5',
         'model_weights_vgg16.h5',
         'train.csv',
         'example_submission.csv']
```

ΦΟΡΤΩΣΗ ΔΕΔΟΜΕΝΩΝ

Φορτώνουμε τα δεδομένα των εικόνων με ετικέτες και τα εμφανίζουμε.

```
In [8]: data = pd.read_csv(path+'icml_face_data.csv')

In [9]: data.head()

Out[9]:
```

	emotion	Usage	pixels
0	0	Training	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...
1	0	Training	151 150 147 155 148 133 111 140 170 174 182 15...
2	2	Training	231 212 156 164 174 138 161 173 182 200 106 38...
3	4	Training	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...
4	6	Training	4 0 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...

ΣΥΝΑΡΤΗΣΕΙΣ

Ορίζουμε μερικές βοηθητικές συναρτήσεις για την προετοιμασία και την "χάραξη" των δεδομένων.

In [10]:
```
def prepare_data(data):
    """ Προετοιμάζουμε τα δεδομένα για μοντελοποίηση.
        είσοδος: πλαίσια δεδομένων με ετικέτες και δεδομένα pixel
        έξοδος : πίνακας εικόνων και ετικετών """

    image_array = np.zeros(shape=(len(data), 48, 48))
    image_label = np.array(list(map(int, data['emotion'])))

    for i, row in enumerate(data.index):
        image = np.fromstring(data.loc[row, ' pixels'], dtype=int, sep=' ')
        image = np.reshape(image, (48, 48))
        image_array[i] = image

    return image_array, image_label

def plot_examples(label=0):
    fig, axs = plt.subplots(1, 5, figsize=(25, 12))
    fig.subplots_adjust(hspace = .2, wspace=.2)
    axs = axs.ravel()
    for i in range(5):
        idx = data[data['emotion']==label].index[i]
        axs[i].imshow(train_images[idx][:,:,0], cmap='gray')
        axs[i].set_title(emotions[train_labels[idx].argmax()])
        axs[i].set_xticklabels([])
        axs[i].set_yticklabels([])

def plot_all_emotions():
    fig, axs = plt.subplots(1, 7, figsize=(30, 12))
    fig.subplots_adjust(hspace = .2, wspace=.2)
    axs = axs.ravel()
    for i in range(7):
        idx = data[data['emotion']==i].index[i]
        axs[i].imshow(train_images[idx][:,:,0], cmap='gray')
        axs[i].set_title(emotions[train_labels[idx].argmax()])
        axs[i].set_xticklabels([])
        axs[i].set_yticklabels([])
```

Ορίζουμε μερικές βοηθητικές συναρτήσεις για την προετοιμασία και την “χάραξη” των δεδομένων.

```
def plot_image_and_emotion(test_image_array, test_image_label, pred_test_labels, image_number):
    """ Συνάρτηση που σχεδίαζει την  εικόνα και συγκρίνει τα αποτελέσματα της πρόβλεψης με την ετικέτα """

    fig, axs = plt.subplots(1, 2, figsize=(12, 6), sharey=False)

    bar_label = emotions.values()

    axs[0].imshow(test_image_array[image_number], 'gray')
    axs[0].set_title(emotions[test_image_label[image_number]])

    axs[1].bar(bar_label, pred_test_labels[image_number], color='orange', alpha=0.7)
    axs[1].grid()

    plt.show()

def plot_compare_distributions(array1, array2, title1='', title2=''):
    df_array1 = pd.DataFrame()
    df_array2 = pd.DataFrame()
    df_array1['emotion'] = array1.argmax(axis=1)
    df_array2['emotion'] = array2.argmax(axis=1)

    fig, axs = plt.subplots(1, 2, figsize=(12, 6), sharey=False)
    x = emotions.values()

    y = df_array1['emotion'].value_counts()
    keys_missed = list(set(emotions.keys()).difference(set(y.keys())))
    for key_missed in keys_missed:
        y[key_missed] = 0
    axs[0].bar(x, y.sort_index(), color='orange')
    axs[0].set_title(title1)
    axs[0].grid()

    y = df_array2['emotion'].value_counts()
    keys_missed = list(set(emotions.keys()).difference(set(y.keys())))
    for key_missed in keys_missed:
        y[key_missed] = 0
    axs[1].bar(x, y.sort_index())
    axs[1].set_title(title2)
    axs[1].grid()

    plt.show()
```

ΕΠΙΣΚΟΠΗΣΗ

```
In [13]: data[' Usage'].value_counts()

Out[13]: Training       28709
         PrivateTest     3589
         PublicTest      3589
         Name:  Usage, dtype: int64
```

Μέσω της επισκόπησης ενημερωνόμαστε σχετικά με την φύση των δεδομένων. Στην δική μας περίπτωση παρατηρούμε πως έχουμε 28.709 φωτογραφίες με τις οποίες εκπαιδευτεί το μοντέλο και 3.589 για να το τεστάρουμε.

ΠΡΟΕΤΟΙΜΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

- Ορίζουμε τα συναισθήματα.

```
In [14]: emotions = {0: 'Angry', 1: 'Disgust', 2: 'Fear', 3: 'Happy', 4: 'Sad', 5: 'Surprise', 6: 'Neutral'}
```

- Και ορίζουμε εξίσου τα δεδομένα μας σε δεδομένα εκπαίδευσης και δεδομένα δοκιμής [training data – test data].

```
In [15]: train_image_array, train_image_label = prepare_data(data[data[' Usage']=='Training'])
         val_image_array, val_image_label = prepare_data(data[data[' Usage']=='PrivateTest'])
         test_image_array, test_image_label = prepare_data(data[data[' Usage']=='PublicTest'])
```

- Έπειτα reshape και scale στις εικόνες.

```
In [16]: train_images = train_image_array.reshape((train_image_array.shape[0], 48, 48, 1))
         train_images = train_images.astype('float32')/255
         val_images = val_image_array.reshape((val_image_array.shape[0], 48, 48, 1))
         val_images = val_images.astype('float32')/255
         test_images = test_image_array.reshape((test_image_array.shape[0], 48, 48, 1))
```

- Ακολουθεί η κωδικοποίηση της τιμής στόχου.

```
In [17]: train_labels = to_categorical(train_image_label)
         val_labels = to_categorical(val_image_label)
         test_labels = to_categorical(test_image_label)
```

ΠΑΡΑΔΕΙΓΜΑΤΑ

Με την εντολή plot_all_emotions() θα εμφανίσουμε 7 φωτογραφίες , 1 για κάθε συναίσθημα.

```
In [18]: plot_all_emotions()
```

Γράφοντας plot_examples(label= λ) και αλλάζοντας το λ=0,1,2,3,4,5,6 θα μας εμφανίζονται κάθε φορά 5 φωτογραφίες που θα αντιστοιχούν στο ίδιο συναίσθημα.

- ***ΝΕΥΡΑ[λ=0]***

```
In [19]: plot_examples(label=0)
```

- ***ΑΗΔΙΑ[λ=1]***

```
In [20]: plot_examples(label=1)
```

- ***ΦΟΒΟΣ[λ=2]***

```
In [21]: plot_examples(label=2)
```

- ***ΧΑΡΑ[λ=3]***

```
In [22]: plot_examples(label=3)
```

- ***ΛΥΠΗ[λ=4]***

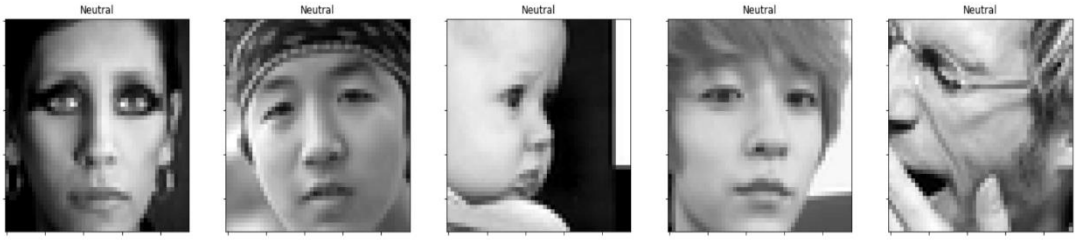
```
In [23]: plot_examples(label=4)
```

- ***ΕΚΠΛΗΞΗ[λ=5]***

```
In [24]: plot_examples(label=5)
```

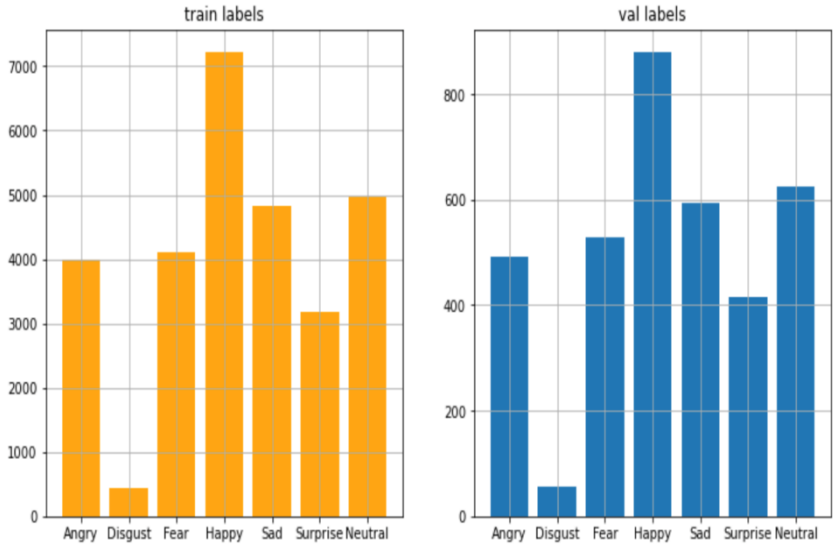
- ***ΟΥΔΕΤΕΡΟΤΗΤΑ[λ=6]***

```
In [25]: plot_examples(label=6)
```



ΚΑΤΑΝΟΜΗ ΕΤΙΚΕΤΩΝ

```
In [26]: plot_compare_distributions(train_labels, val_labels, title1='train labels', title2='val labels')
```



ΒΑΡΗ "ΤΑΞΗΣ"

Υπολογίζουμε τα βάρη τάξης από την κατανομή των ετικετών.

```
In [27]: class_weight = dict(zip(range(0, 7), (((data[data[' Usage']=='Training']['emotion']
         .value_counts()).sort_index())/len(data[data[' Usage']=='Training']['emotion'])).to
         list()))
```

```
In [28]: class_weight
```

```
Out[28]: {0: 0.1391549688251071,
          1: 0.01518687519593159,
          2: 0.14270786164617366,
          3: 0.2513149186666202,
          4: 0.16823992476226968,
          5: 0.11045316799609878,
          6: 0.17294228290779895}
```

ΜΟΝΤΕΛΟ

Ορίζουμε ένα απλό CNN μοντέλο και βλέπουμε με συντομία τα χαρακτηριστικά του

[model.summary()]

In [29]:
```
model = models.Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(48, 48, 1)))
model.add(MaxPool2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPool2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dense(7, activation='softmax'))
```

In [30]:
```
model.compile(optimizer=Adam(lr=1e-3), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
/opt/anaconda3/lib/python3.8/site-packages/tensorflow/python/keras/optimizer_v2/optimizer_v2.py:374: UserWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  warnings.warn(
```

In [31]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 46, 46, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 23, 23, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 21, 21, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 10, 10, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 8, 8, 64)          36928
_________________________________________________________________
flatten (Flatten)            (None, 4096)              0
_________________________________________________________________
dense (Dense)                (None, 64)                262208
_________________________________________________________________
dense_1 (Dense)              (None, 7)                 455
=================================================================
Total params: 318,407
Trainable params: 318,407
Non-trainable params: 0
```

Ξεκινάμε την εκπαίδευση του μοντέλου για 12 εποχές και πετυχαίνουμε test accuracy 53,9% την οποία εμφανίζουμε με την εντολή print('test accuracy:', test_acc).

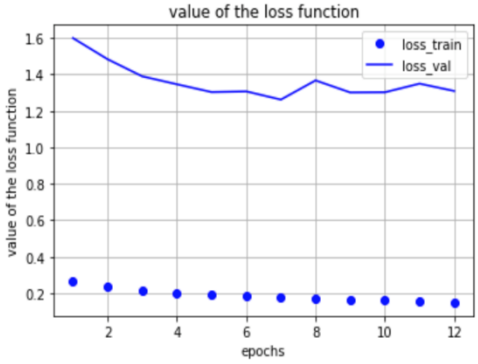
In [26]:
```
test_loss, test_acc = model.evaluate(test_images, test_labels)
print('test caccuracy:', test_acc)
```

```
113/113 [==============================] - 2s 15ms/step - loss: 1.4217 - accuracy:
0.5391
test caccuracy: 0.5391473770141602
```

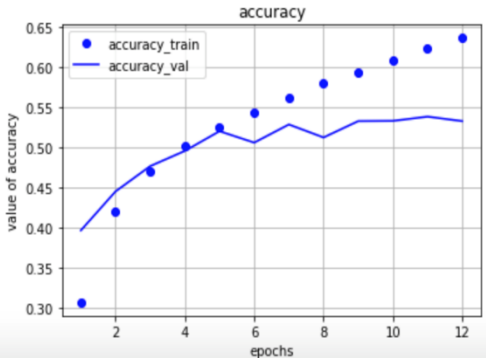
ΑΝΑΛΥΣΗ ΣΥΓΚΛΙΣΗΣ

Μέσω της σύγκλισης καθορίζουμε πόσα στοιχεία απαιτούνται σε ένα μοντέλο για να διασφαλιστεί ότι τα αποτελέσματα μιας ανάλυσης δεν επηρεάζονται από την αλλαγή του μεγέθους του πλέγματος.

In [36]:
```
loss = history.history['loss']
loss_val = history.history['val_loss']
epochs = range(1, len(loss)+1)
plt.plot(epochs, loss, 'bo', label='loss_train')
plt.plot(epochs, loss_val, 'b', label='loss_val')
plt.title('value of the loss function')
plt.xlabel('epochs')
plt.ylabel('value of the loss function')
plt.legend()
plt.grid()
plt.show()
```



In [37]:
```
acc = history.history['accuracy']
acc_val = history.history['val_accuracy']
epochs = range(1, len(loss)+1)
plt.plot(epochs, acc, 'bo', label='accuracy_train')
plt.plot(epochs, acc_val, 'b', label='accuracy_val')
plt.title('accuracy')
plt.xlabel('epochs')
plt.ylabel('value of accuracy')
plt.legend()
plt.grid()
plt.show()
```

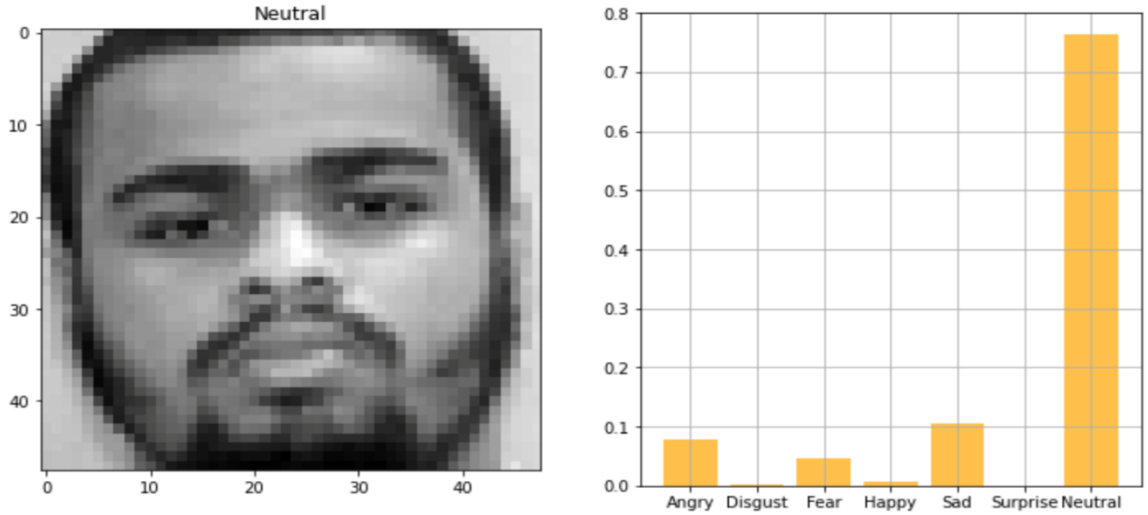


ΑΝΑΛΥΣΗ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Παίρνοντας τυχαία μια φωτογραφία από τις κατηγορίες συναισθημάτων (ΟΥΔΕΤΕΡΟΤΗΤΑ και ΝΕΥΡΑ) του FER2013 δείχνουμε πόσο καλά την προβλέπει το μοντέλο μας. Και συγκρίνουμε επιπλέον τις ετικέτες δοκιμής με τις πρόβλεψης.

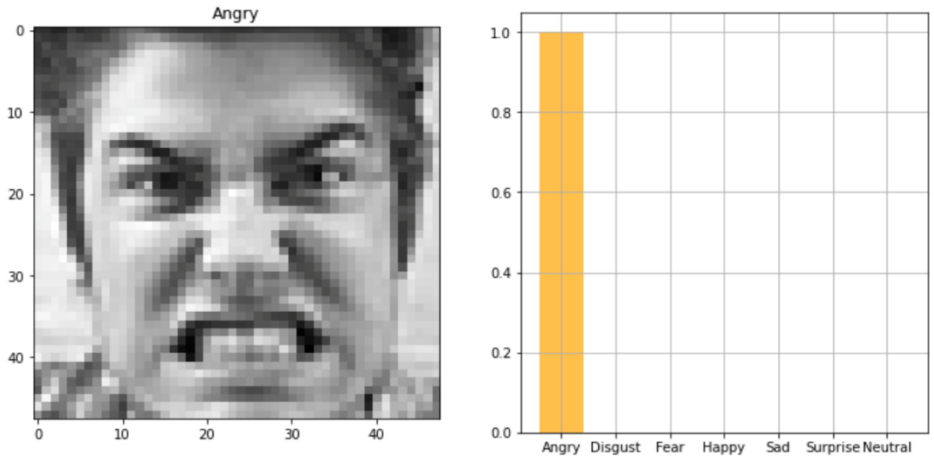
◊ ***ΟΥΔΕΤΕΡΟΤΗΤΑ***

```
In [54]: plot_image_and_emotion(test_image_array, test_image_label, pred_test_labels, 106)
```



◊ ***ΝΕΥΡΑ***

```
In [55]: plot_image_and_emotion(test_image_array, test_image_label, pred_test_labels, 40)
```



ΑΝΑΛΥΣΗ ΛΑΘΟΣ ΠΡΟΒΛΕΨΗΣ

Δεδομένου ότι πετύχαμε ακρίβεια 53% στο test set είναι πολύ σημαντικό να δώσουμε έμφαση και στις λάθος προβλέψεις. Θέλουμε να εξάγουμε διάφορες λεπτομέρειες έτσι ώστε να βελτιώσουμε στο μέλλον το μοντέλο.

In [57]:
```
df_compare = pd.DataFrame()
df_compare['real'] = test_labels.argmax(axis=1)
df_compare['pred'] = pred_test_labels.argmax(axis=1)
df_compare['wrong'] = np.where(df_compare['real']!=df_compare['pred'], 1, 0)
```

In [58]:
```
conf_mat = confusion_matrix(test_labels.argmax(axis=1), pred_test_labels.argmax(axis=1))

fig, ax = plot_confusion_matrix(conf_mat=conf_mat,
                                show_normed=True,
                                show_absolute=False,
                                class_names=emotions.values(),
                                figsize=(8, 8))
fig.show()
```

```
<ipython-input-58-73a2c2fbda11>:8: UserWarning: Matplotlib is currently using module://ipykernel.pylab.backend_inline
, which is a non-GUI backend, so cannot show the figure.
  fig.show()
```

HYPERPARAMETER TUNING[CONVNET]

Αρχικά το convnet είναι ένα ειδικό είδος νευρωνικών δικτύων που περιέχει τουλάχιστον ένα συνελικτικό στρώμα. Μια τυπική δομή convnet λαμβάνει μια εικόνα, την περνάει μέσα από μια σειρά συνελικτικών στρωμάτων, μη γραμμικά επίπεδα ενεργοποίησης, συγκέντρωση (downsampling) και ένα πλήρως συνδεδεμένο επίπεδο για την έξοδο των ετικετών ταξινόμησης.Ένα δίκτυο διανομής διαφέρει από ένα κανονικό νευρωνικό δίκτυο με τη χρήση του συνελικτικού στρώματος. Σε ένα κανονικό νευρωνικό δίκτυο, χρησιμοποιούμε ολόκληρη την εικόνα για να το εκπαιδεύσουμε. Λειτουργεί καλά για απλή κεντραρισμένη εικόνα (για παράδειγμα, κεντρογραφημένη ψηφιακή εικόνα), αλλά δεν αναγνωρίζει την εικόνα με πιο περίπλοκες παραλλαγές (για παράδειγμα, μια γάτα που τρέχει στην αυλή). Το να έχουμε περισσότερα κρυμμένα στρώματα για να μάθουμε αφηρημένα χαρακτηριστικά θα βοηθούσε αλλά είναι μάλλον μη πρακτικό καθώς χρειαζόμαστε πάρα πολλούς νευρώνες για να εκπαιδεύσουμε και να αποθηκεύσουμε στη μνήμη.Από την άλλη πλευρά, ένα convnet αναγνωρίζει ένα αντικείμενο αναζητώντας πρώτα χαρακτηριστικά χαμηλού επιπέδου όπως ακμές, γραμμές και καμπύλες και στη συνέχεια δημιουργώντας πιο αφηρημένα χαρακτηριστικά (για παράδειγμα τι κάνει τη γάτα γάτα) μέσω σειράς στρεπτικών στρώσεων. Κατά τη διάρκεια της διαδικασίας εκμάθησης στο στρώμα περιστροφής, το δίκτυο μαθαίνει μεμονωμένα μέρη ενός αντικειμένου με κοινά φίλτρα (ή ανιχνευτές χαρακτηριστικών) σε μικρές περιοχές της εικόνας και τα προσθέτει για να δημιουργήσει αφηρημένες λειτουργίες. Η χρήση κοινών φίλτρων μειώνει σε μεγάλο βαθμό την πραγματική εκμάθηση παραμέτρων.

Hyperparameter tuning

(Οι συγκεκριμένες "υπερπαράμετροι” επιλέχθηκαν με βάση τις παραμέτρους ενός απλού πειραματικού cnn που μας έδωσε την ευκαιρία να αξιοποιήσουμε τον “απλό” εξοπλισμό που έχουμε και να παράγουμε παρ' όλα αυτά ένα αποτέλεσμα! Παρακάτω θα εξηγήσουμε ποίες είναι οι "υπερπαράμετροι” και τι “ρόλο” έχουν!)

Ο συντονισμός των “υπερπαραμέτρων” για ένα βαθύ νευρωνικό δίκτυο είναι τόσο δύσκολος όσο και αργός όσον αφορά την εκπαίδευση σε ένα βαθύ νευρωνικό δίκτυο και υπάρχουν αριθμητικές παράμετροι για διαμόρφωση. Σε αυτό το μέρος, εξετάζουμε εν συντομία τις “υπερπαραμέτρους” για το convnet.

1)Learning rate,

Το ποσοστό εκμάθησης ελέγχει πόσο ενημερώνεται το βάρος στον αλγόριθμο βελτιστοποίησης. Μπορούμε να χρησιμοποιήσουμε σταθερό ρυθμό μάθησης, σταδιακά μειούμενο ρυθμό εκμάθησης, μεθόδους που βασίζονται στην ορμή ή ρυθμούς προσαρμογής μάθησης, ανάλογα με την επιλογή βελτιστοποιητή όπως SGD, Adam, Adagrad, AdaDelta ή RMSProp.

2) Αριθμός Εποχών[12 εποχές]

Ο αριθμός των εποχών είναι ο αριθμός των φορών που ολόκληρο το σετ εκπαίδευσης περνά μέσα από το νευρωνικό δίκτυο. Θα πρέπει να αυξήσουμε τον αριθμό των εποχών μέχρι να δούμε ένα μικρό κενό μεταξύ του σφάλματος δοκιμής και του λάθους κατάρτισης.

3) Batch Size (μέγεθος παρτίδας)

Η "μίνι παρτίδα" είναι συνήθως προτιμότερη στη διαδικασία εκμάθησης του convnet. Ένα εύρος από 16 έως 128 είναι μια καλή επιλογή για δοκιμή. Πρέπει να σημειώσουμε ότι το convnet είναι ευαίσθητο στο μέγεθος της παρτίδας.

4) Συνάρτηση Ενεργοποίησης[relu]

Η συνάρτηση ενεργοποίησης εισάγει τη μη γραμμικότητα στο μοντέλο. Συνήθως, ο ανορθωτής λειτουργεί καλά με το convnet. Άλλες εναλλακτικές λύσεις είναι το sigmoid, tanh και άλλες λειτουργίες ενεργοποίησης που μειώνονται κατά την εργασία.

5) Αριθμός κρυφών στρωμάτων και μονάδων

Είναι συνήθως καλό να προσθέσουμε περισσότερα επίπεδα έως ότου το σφάλμα δοκιμής σταματά να βελτιώνεται. Το μειονέκτημα είναι ότι είναι υπολογιστικά "δαπανηρό" να εκπαιδεύσεις το δίκτυο. Η κατοχή μικρής ποσότητας μονάδων μπορεί να οδηγήσει σε ανεπαρκή προσαρμογή, ενώ η ύπαρξη περισσότερων μονάδων συνήθως δεν είναι επιβλαβής με την κατάλληλη τακτοποίηση.

6)Εκκίνηση βάρους[weight initialization]

Πρέπει να αρχικοποιούμε τα βάρη με μικρούς τυχαίους αριθμούς για να αποτρέψουμε τους νεκρούς νευρώνες, αλλά όχι πολύ μικρά για να αποφύγουμε τη μηδενική κλίση. Η ομοιόμορφη κατανομή συνήθως λειτουργεί καλά.

7)Dropout for regularization

Το dropout είναι μια προτιμώμενη τεχνική κανονικοποίησης για την αποφυγή υπερβολικής προσαρμογής σε βαθιά νευρωνικά δίκτυα. Η μέθοδος απλώς ρίχνει μονάδες στο νευρωνικό δίκτυο σύμφωνα με την επιθυμητή πιθανότητα. Μια προεπιλεγμένη τιμή 0,5 είναι μια καλή επιλογή για δοκιμή.

EVALUATION METRICS

Μέσω αυτών καταλάβαμε πως το μοντέλο μπορεί και σίγουρα χρειάζεται να βελτιωθεί στο μέλλον είτε μέσω εκπαίδευσης με περισσότερα δεδομένα είτε με περισσότερες εποχές!

```
In [63]: # Recall
         from sklearn.metrics import recall_score
         recall_score(test_labels,pred_test_labels, average=None)

Out[63]: array([0.25053533, 0.        , 0.72580645, 0.34636872, 0.18070444,
                0.37349398, 0.09390445])

In [64]: # Precision
         from sklearn.metrics import precision_score
         precision_score(test_labels,pred_test_labels, average=None)

Out[64]: array([0.32054795, 0.        , 0.18126888, 0.75980392, 0.40136054,
                0.37990196, 0.4453125 ])

In [66]: from sklearn.metrics import f1_score
         f1_score(test_labels, pred_test_labels, average=None)

Out[66]: array([0.28125   , 0.        , 0.29008864, 0.47582502, 0.24920803,
                0.37667072, 0.15510204])
```

Είναι σημαντικό όμως να εξηγήσουμε τις παραπάνω μεθόδους αξιολόγησης , καθώς και το πότε τις χρησιμοποιούμε!

Accuracy,Precision,Recall :

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

A. Accuracy

Η ακρίβεια είναι η βασική μέτρηση ταξινόμησης. Είναι αρκετά εύκολο να την καταλάβουμε. Είναι κατάλληλη τόσο για δυαδικά προβλήματα όσο και για προβλήματα ταξινόμησης πολλαπλών τάξεων.

Accuracy = (TP+TN)/(TP+FP+FN+TN)

Η ακρίβεια είναι το ποσοστό των πραγματικών αποτελεσμάτων μεταξύ του συνολικού αριθμού των περιπτώσεων που εξετάστηκαν.

Πότε την χρησιμοποιούμε;

Η ακρίβεια είναι μια έγκυρη επιλογή αξιολόγησης για προβλήματα ταξινόμησης τα οποία είναι καλά ισορροπημένα και δεν είναι "στραβά" ή χωρίς τάξη.

B. Precision

Ας ξεκινήσουμε με το precision, το οποίο απαντά στην ακόλουθη ερώτηση: ποιο ποσοστό των προβλεπόμενων θετικών είναι πραγματικά θετικό;

Precision = (TP)/(TP+FP)

Πότε το χρησιμοποιούμε;

Το precision είναι μια έγκυρη επιλογή μέτρησης αξιολόγησης την οποία χρησιμοποιούμε όταν θέλουμε να είμαστε πολύ σίγουροι για την πρόβλεψή μας. Για παράδειγμα: Εάν χτίζουμε ένα σύστημα για να προβλέψουμε εάν πρέπει να μειώσουμε το πιστωτικό όριο σε έναν συγκεκριμένο λογαριασμό, θέλουμε να είμαστε πολύ σίγουροι για την πρόβλεψή μας ή μπορεί να έχει ως αποτέλεσμα τη δυσαρέσκεια των πελατών.

Γ. Recall

Ένα άλλο πολύ χρήσιμο μέτρο είναι η ανάκληση, η οποία απαντά σε μια διαφορετική ερώτηση: ποια αναλογία των πραγματικών θετικών είναι σωστά ταξινομημένη;

Recall = (TP)/(TP+FN)

Πότε το χρησιμοποιούμε;

Η ανάκληση είναι μια έγκυρη επιλογή μέτρησης αξιολόγησης όταν θέλουμε να αποτυπώσουμε όσο το δυνατόν περισσότερα θετικά. Για παράδειγμα: Εάν χτίζουμε ένα σύστημα για να προβλέψουμε αν ένα άτομο έχει καρκίνο ή όχι, θέλουμε να συλλάβουμε την ασθένεια ακόμη και αν δεν είμαστε πολύ σίγουροι.

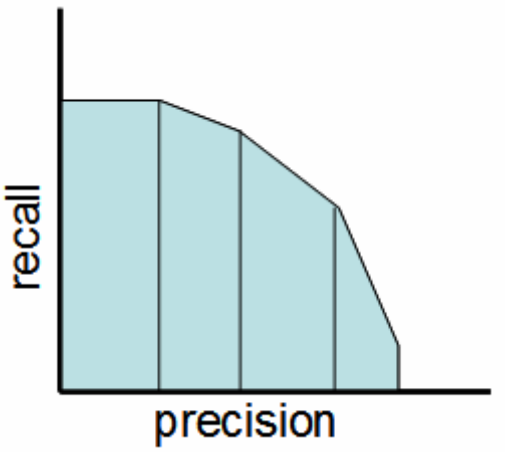
2. F1 score :

Η βαθμολογία F1 είναι ένας αριθμός μεταξύ 0 και 1 και είναι ο αρμονικός μέσος όρος ακρίβειας και ανάκλησης.

Ας ξεκινήσουμε με ένα δυαδικό πρόβλημα πρόβλεψης. Προβλέπουμε αν ένας αστεροειδής θα χτυπήσει τη γη ή όχι.Αν λοιπόν πούμε «Όχι» για ολόκληρο το σετ προπόνησης, η ακρίβειά μας θα είναι 0. Ποια είναι η ανάκληση της θετικής μας τάξης; Είναι μηδέν. Ποια είναι η ακρίβεια; Είναι πάνω από 99%.Και ως εκ τούτου, η βαθμολογία F1 είναι επίσης 0. Και έτσι γνωρίζουμε ότι ο ταξινομητής που έχει ακρίβεια 99% είναι βασικά χωρίς αξία για την περίπτωσή μας. Και ως εκ τούτου λύνει το πρόβλημά μας.

<u>Πότε το χρησιμοποιούμε;</u>

Θέλουμε να έχουμε ένα μοντέλο με καλή ακρίβεια και ανάκληση.



Ανταλλαγή ακριβείας-ανάκλησης

Με απλά λόγια, η βαθμολογία F1 διατηρεί μια ισορροπία μεταξύ της ακρίβειας και της ανάκλησης για τον ταξινομητή μας. Εάν η ακρίβειά μας είναι χαμηλή, η F1 είναι χαμηλή και εάν η ανάκληση είναι χαμηλή και πάλι η βαθμολογία F1 μας είναι χαμηλή.

Πχ. Εάν είστε αστυνομικός επιθεωρητής και θέλετε να πιάσετε εγκληματίες, θέλετε να είστε σίγουροι ότι το άτομο που πιάνετε είναι εγκληματίας (Precision) και θέλετε επίσης να συλλάβετε όσο το δυνατόν περισσότερους εγκληματίες γίνεται (Ανάκληση). Το F1 score πετυχαίνει αυτήν την αντιστάθμιση!

<u>Και κάπως έτσι τελείωσε η πρώτη εφαρμογή/ project !</u>

II)Face Emotion Recognition- TensforFlow – Transfer Learning

Όπως προαναφέρθηκε και στο θεωρητικό υπόβαθρο, η μέθοδος της " μεταφοράς μάθησης" είναι ιδιαίτερα χρήσιμη! Σε περίπτωση που θέλουμε να κάνουμε "σωστή δουλεία" με γρήγορο και εγγυημένο αποτέλεσμα ενώ δεν έχουμε ούτε τον απαραίτητο εξοπλισμό μα ούτε αρκετό χρόνο, αυτή είναι η κατάλληλη μέθοδος!

ΕΙΣΑΓΩΓΗ ΒΙΒΛΙΟΘΗΚΩΝ ΚΑΙ ΟΡΙΣΜΟΣ ΤΑΞΕΩΝ

In [1]:
```
import tensorflow as tf
import cv2
```

In [6]:
```
import os
import matplotlib.pyplot as plt
import numpy as np
```

In [28]:
```
img_array = cv2.imread("/Users/user/Desktop/images/train/0/43.jpg")
```

In [29]:
```
img_array.shape
```

Out[29]:
```
(48, 48, 3)
```

In [30]:
```
plt.imshow(img_array)
```

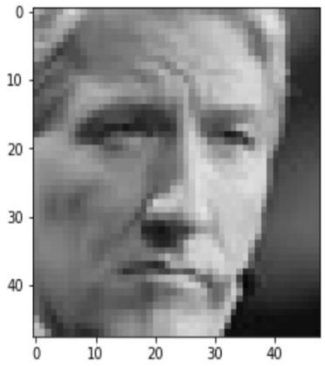
Out[30]:
```
<matplotlib.image.AxesImage at 0x147aa8b20>
```

In [31]:
```
Datadirectory = "/Users/user/Desktop/images/train"
```

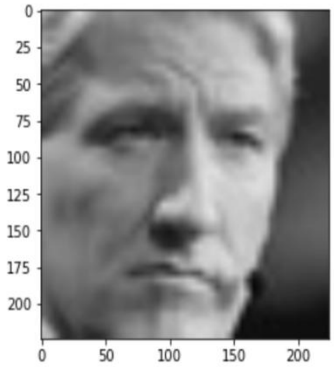
In [32]:
```
Classes = ["0","1","2","3","4","5","6"]
```

Στην συνέχεια ήθελα να δείξω όλες τις εικόνες άλλα δεδομένου ότι ήταν πολλές χρησιμοποίησα την εντολή break και έδειξα μονάχα την πρώτη φωτογραφία από τον φάκελο! Αλλάζω επίσης το μέγεθος των εικόνων λόγω ImageNet!

```
In [33]: for category in Classes:
             path = os.path.join(Datadirectory, category)
             for img in os.listdir(path):
                 img_array = cv2.imread(os.path.join(path, img))
                 plt.imshow(cv2.cvtColor(img_array, cv2.COLOR_BGR2RGB))
                 plt.show()
                 break
             break
```




```
In [34]: img_size = 224
         new_array = cv2.resize(img_array,(img_size,img_size))
         plt.imshow(cv2.cvtColor(new_array, cv2.COLOR_BGR2RGB))
         plt.show()
```




```
In [35]: new_array.shape

Out[35]: (224, 224, 3)
```

ΑΝΑΓΝΩΣΗ ΟΛΩΝ ΤΩΝ ΕΙΚΟΝΩΝ-ΜΕΤΑΤΡΟΠΗ ΤΟΥΣ ΣΕ ΠΙΝΑΚΑ

Ορίζω εξίσου το X και το y που αντιστοιχούν στα δεδομένα / ετικέτες!

ΑΝΑΓΝΩΣΗ ΟΛΩΝ ΤΩΝ ΕΙΚΟΝΩΝ-ΜΕΤΑΤΡΟΠΗ ΤΟΥΣ ΣΕ ΠΙΝΑΚΑ

In [36]:
```
train_Data = []

def create_train_Data():
    for category in Classes:
        path = os.path.join(Datadirectory, category)
        class_num = Classes.index(category) ##0, 1 ## Ετικέτα
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path,img))
                new_array= cv2.resize(img_array, (img_size,img_size))
                train_Data.append([new_array,class_num])
            except Exception as e:
                pass
```

In [38]:
```
create_train_Data()
```

In [39]:
```
print(len(train_Data))
```

```
8068
```

In [40]:
```
temp = np.array(train_Data)
```

In [41]:
```
temp.shape
```

Out[41]:
```
(8068, 2)
```

In [42]:
```
import random

random.shuffle(train_Data)
```

In [43]:
```
X = [] ## δεδομένα / χαρακτηριστικά
y = []  ## ετικέτα

for features,label in train_Data:
    X.append(features)
    y.append(label)

X= np.array(X).reshape(-1, img_size, img_size, 3) ## μετατροπή σε 4 διαστάσεις
```

In [44]:
```
y[3]
```

Out[44]:
```
5
```

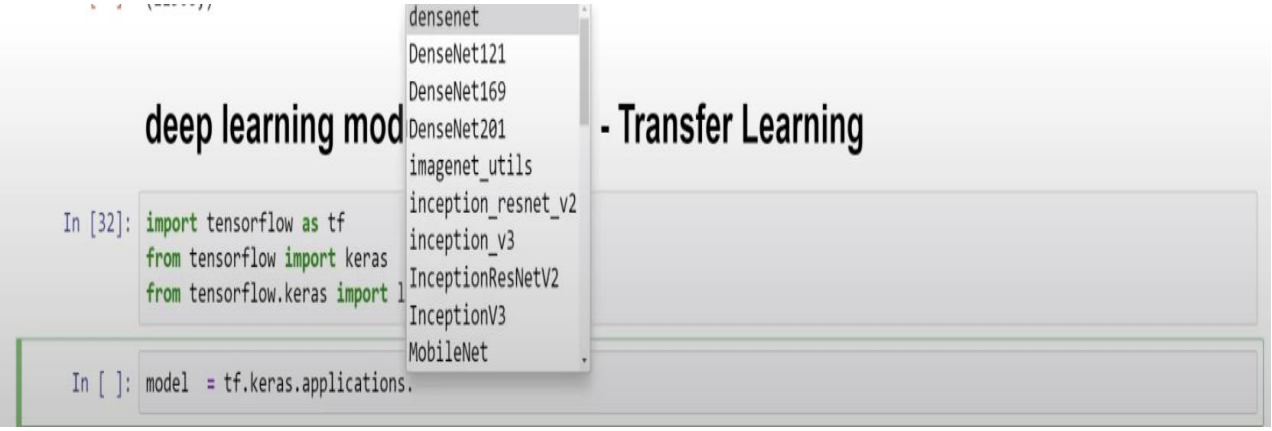
In [45]:
```
Y = np.array(y)
```

In [46]:
```
Y.shape
```

Out[46]:
```
(8068,)
```

DEEP LEARNING MODEL [TRAINING] - TRANSFER LEARNING

Εδώ εισάγουμε το έτοιμο, ήδη εκπαιδευμένο μοντέλο MobileNetV2 και δείχνουμε παράλληλα και ποια είναι τα άλλα διαθέσιμα μοντέλα!



TRANSFER LEARNING- TUNING

Στις παρακάτω γραμμές κώδικα γίνονται τα εξής:

- Tuning στα βάρη έτσι ώστε να ξεκινούν από το προηγούμενο check point.
- Ορίζουμε το νέο μοντέλο και βλέπουμε μια περίληψη του.
- Το εκπαιδεύουμε για 30 εποχές , πετυχαίνοντας ακρίβεια 82,4%.
- Το αποθηκεύουμε και στην συνεχεία το τεστάρουμε.

TRANSFER LEARNING- TUNING

In [50]:
```
base_input = model.layers[0].input
```

In [51]:
```
base_output = model.layers[-2].output
```

In [52]:
```
base_output
```

Out[52]:
```
<KerasTensor: shape=(None, 1280) dtype=float32 (created by layer 'global_average_pooling2d')>
```

In [53]:
```
final_output = layers.Dense(128)(base_output) #adding new layer , after the output of global pooling layer
final_output = layers.Activation('relu')(final_output) # activation function
final_output = layers.Dense(64)(final_output)
final_output = layers.Activation('relu')(final_output)
final_output = layers.Dense(7, activation='softmax')(final_output) # my classes are 07
```

In [54]:
```
new_model = keras.Model(inputs = base_input, outputs = final_output)
```

In [55]:
```
new_model.summary()
```

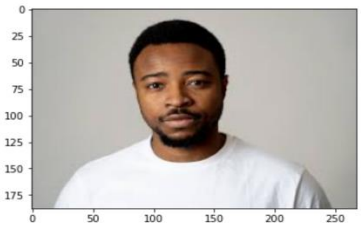
```
block_6_expand_relu (ReLU)      (None, 28, 28, 192)  0           block_6_expand_BN[0][0]
__________________________________________________________________________________________________
block_6_pad (ZeroPadding2D)     (None, 29, 29, 192)  0           block_6_expand_relu[0][0]
__________________________________________________________________________________________________
block_6_depthwise (DepthwiseCon (None, 14, 14, 192)  1728        block_6_pad[0][0]
__________________________________________________________________________________________________
block_6_depthwise_BN (BatchNorm (None, 14, 14, 192)  768         block_6_depthwise[0][0]
__________________________________________________________________________________________________
block_6_depthwise_relu (ReLU)   (None, 14, 14, 192)  0           block_6_depthwise_BN[0][0]
__________________________________________________________________________________________________
block_6_project (Conv2D)        (None, 14, 14, 64)   12288       block_6_depthwise_relu[0][0]
__________________________________________________________________________________________________
block_6_project_BN (BatchNormal (None, 14, 14, 64)   256         block_6_project[0][0]
__________________________________________________________________________________________________
block_7_expand (Conv2D)         (None, 14, 14, 384)  24576       block_6_project_BN[0][0]
__________________________________________________________________________________________________
block_7_expand_BN (BatchNormali (None, 14, 14, 384)  1536        block_7_expand[0][0]
__________________________________________________________________________________________________
block_7_expand_relu (ReLU)      (None, 14, 14, 384)  0           block_7_expand_BN[0][0]
```

In [56]:
```
new_model.compile(loss="sparse_categorical_crossentropy", optimizer = "adam", metrics = ["accuracy"])
```

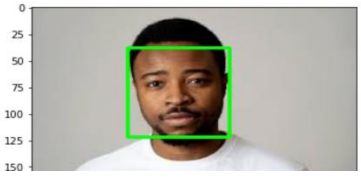
ΔΟΚΙΜΗ ΣΕ ΤΥΧΑΙΑ ΦΩΤΟΓΡΑΦΙΑ

Πριν την εφαρμογή του μοντέλου στις φωτογραφίες, ρυθμίσαμε τον κώδικα έτσι ώστε να σχηματίζει ένα ορθογώνιο γύρω από το πρόσωπο και να εστιάζει μονάχα εκεί για να έχουμε την καλύτερη δυνατή πρόβλεψη

```
In [119]: frame = cv2.imread("/Users/user/Desktop/neutral mofo.jpg") ## εισαγωγή διεύθυνσης εικόνας

In [120]: frame.shape
Out[120]: (188, 268, 3)

In [121]: plt.imshow(cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
Out[121]: <matplotlib.image.AxesImage at 0x1dee8c820>
```




```
In [122]: FaceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')

In [123]: gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

In [124]: gray.shape
Out[124]: (188, 268)

In [125]: faces = FaceCascade.detectMultiScale(gray,1.1, 4)   ## δημιουργία ορθογώνιου πλαισίου
          for x,y,w,h in faces:
              roi_gray = gray[y:y+h, x:x+w]
              roi_color = frame[y:y+h, x:x+w]
              cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),2)
              facess = FaceCascade.detectMultiScale(roi_gray)
              if len(faces) == 0:
                  print("Face not detected")
              else:
                  for (ex,ey,ew,eh) in faces:
                      face_roi = roi_color[ey: ey+eh]

In [126]: plt.imshow(cv2.cvtColor(frame, cv2.COLOR_BGR2RGB))
Out[126]: <matplotlib.image.AxesImage at 0x1e0980f70>
```



Ακολουθεί η πρόβλεψη της έκφρασης που φαίνεται με την μορφή 9.97 αντιστοιχεί στην τάξη 6 που είναι το συναίσθημα της ουδετερότητας! (Σωστή πρόβλεψη)

```
In [128]: final_image = cv2.resize(face_roi, (224,224))
          final_image = np.expand_dims(final_image,axis=0)
          final_image = final_image/255.0

In [129]: Predictions = new_model.predict(final_image)

In [130]: Predictions[0]

Out[130]: array([3.8067068e-05, 8.4541529e-09, 1.9496704e-03, 5.2011768e-05,
                 8.1251725e-05, 1.7598256e-06, 9.9787724e-01], dtype=float32)

In [131]: np.argmax(Predictions)

Out[131]: 6
```

Με αυτό το project έγινε :

- ✓ **κατανόηση της μεθόδου transfer learning**
- ✓ **χρήση pre-trained μοντέλου**
- ✓ **καθώς και δοκιμή του μοντέλου σε φωτογραφία διαφορετική του dataset εκπαίδευσης!**

III) DEEPFACE FACE EMOTION RECOGNITION

Στο συγκεκριμένο "απλό" project έγινε δοκιμή του deepface σε τυχαίες εικόνες του διαδικτύου καθώς και σε δικές μου! Παρακάτω θα δούμε τον κώδικα που το καθιστά αυτό δυνατό αλλά και τα αποτελέσματα του!

- Αρχικά εισάγουμε τις βιβλιοθήκες OpenCv / matplotlib και εν συνεχεία το deepface! Ακολουθούμε την ίδια διαδικασία όσον αφορά τα όρια του προσώπου και την σχεδίαση του ορθογωνίου γύρω απ' αυτό για καλύτερη ανάλυση του!

```
In [1]: import cv2

In [2]: img = cv2.imread('/Users/user/Desktop/Happy Boy.jpg')

In [3]: import matplotlib.pyplot as plt

In [4]: plt.imshow(img)
Out[4]: <matplotlib.image.AxesImage at 0x11c2abee0>
```


```
In [5]: plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
Out[5]: <matplotlib.image.AxesImage at 0x11c36b400>
```


```
In [6]: pip install deepface
from deepface) (1.1.2)
Requirement already satisfied: mtcnn>=0.1.0 in /opt/anaconda3/lib/python3.8/site-packages (
from deepface) (0.1.1)
Requirement already satisfied: six in /opt/anaconda3/lib/python3.8/site-packages (from gdow
n>=3.10.1->deepface) (1.15.0)
Requirement already satisfied: filelock in /opt/anaconda3/lib/python3.8/site-packages (from
gdown>=3.10.1->deepface) (3.0.12)
Requirement already satisfied: requests[socks]>=2.12.0 in /opt/anaconda3/lib/python3.8/site

In [7]: from deepface import DeepFace
```

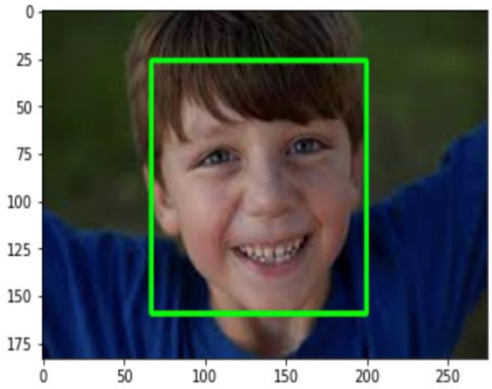
- Έπειτα "τρέχουμε" την πρόβλεψη του μοντέλου και εμφανίζεται γύρω από το πρόσωπο το πιο έντονο συναίσθημα!

```
In [8]: predictions = DeepFace.analyze(img)
        Action: race: 100%|██████████| 4/4 [00:03<00:00,  1.02it/s]

In [9]: predictions
Out[9]: {'emotion': {'angry': 0.009213439982318954,
           'disgust': 3.2649995478621274e-08,
           'fear': 4.126360485451804e-05,
           'happy': 99.90762473253909,
           'sad': 0.0007399025007721315,
           'surprise': 0.0041230798702414865,
           'neutral': 0.07825395090830911},
          'dominant_emotion': 'happy',
          'region': {'x': 67, 'y': 26, 'w': 133, 'h': 133},
          'age': 27,
          'gender': 'Man',
          'race': {'asian': 2.532595954835415,
           'indian': 0.2715089591220021,
           'black': 0.02583955938462168,
           'white': 67.43057370185852,
           'middle eastern': 7.23651647567749,
           'latino hispanic': 22.502967715263367},
          'dominant_race': 'white'}

In [10]: type(predictions)
Out[10]: dict

In [11]: predictions['dominant_emotion']
Out[11]: 'happy'

In [12]: faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xm

In [13]: gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

         faces = faceCascade.detectMultiScale(gray, 1.1, 4)


         for(x,y,w,h) in faces:
             cv2.rectangle(img, (x,y), (x+w,y+h), (0, 255, 0), 2)

In [14]: plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
Out[14]: <matplotlib.image.AxesImage at 0x151e484f0>
```

XAPA

```
In [14]: plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

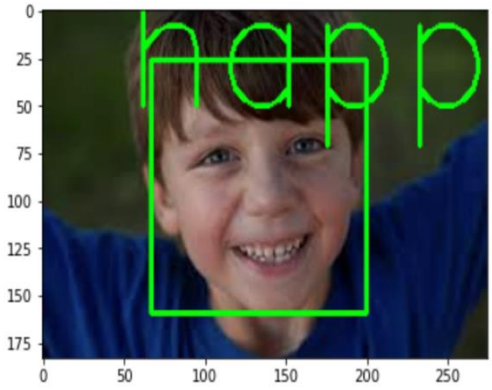
```
Out[14]: <matplotlib.image.AxesImage at 0x151e484f0>
```




```
In [15]: font = cv2.FONT_HERSHEY_SIMPLEX

cv2.putText(img, predictions['dominant_emotion'], (50,50), font, 3, (0, 255, 0), 2, cv2.LINE_4)
```

```
In [16]: plt.imshow(cv2.cvtColor(img, 4))
```

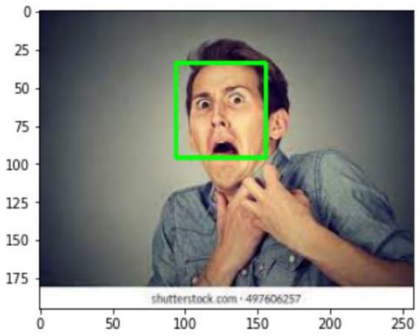
```
Out[16]: <matplotlib.image.AxesImage at 0x176671f70>
```



ΕΚΠΛΗΞΗ

```
In [17]: plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
```

```
Out[17]: <matplotlib.image.AxesImage at 0x14ad93bb0>
```




```
In [18]: font = cv2.FONT_HERSHEY_SIMPLEX

         cv2.putText(img, predictions['dominant_emotion'], (50,50), font, 3, (0, 255, 0), 2, cv2.LINE_4)
```

```
In [20]: predictions['dominant_emotion']
```

```
Out[20]: 'surprise'
```

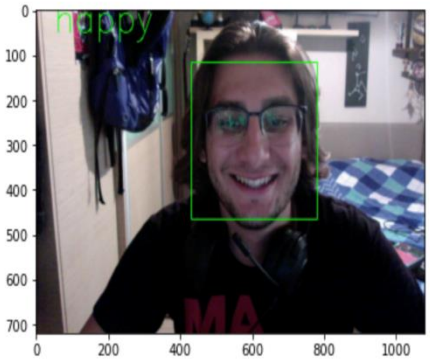
```
In [19]: plt.imshow(cv2.cvtColor(img, 4))
```

```
Out[19]: <matplotlib.image.AxesImage at 0x1494bb460>
```

ΕΦΑΡΜΟΓΗ DEEPFACE ΣΕ ΠΡΟΣΩΠΙΚΕΣ ΦΩΤΟΓΡΑΦΙΕΣ

Παρακάτω παρατίθενται ενδεικτικές εικόνες(των 3 συναισθημάτων) από πειραματική δοκιμή της εργασίας μαζί με τα αποτελέσματά τους.

ΧΑΡΑ

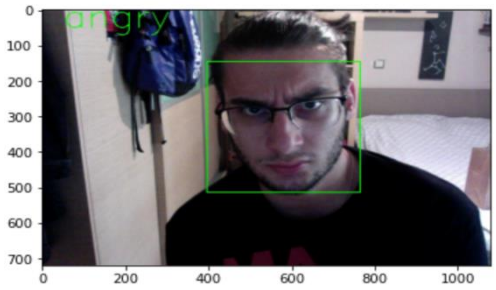



```
In [18]: predictions

Out[18]: {'emotion': {'angry': 1.0300662722784182e-05,
                      'disgust': 1.6751659484227577e-18,
                      'fear': 3.7072284803617295e-11,
                      'happy': 99.99998807907104,
                      'sad': 3.070007026373389e-08,
                      'surprise': 1.6351624102049073e-11,
                      'neutral': 1.5754002546941592e-06},
                     'dominant_emotion': 'happy',
                     'region': {'x': 431, 'y': 117, 'w': 349, 'h': 349},
                     'age': 32,
                     'gender': 'Man',
                     'race': {'asian': 0.005194191908231005,
                              'indian': 0.27916315011680126,
                              'black': 0.002109621709678322,
                              'white': 70.11755704879761,
                              'middle eastern': 24.888212978839874,
                              'latino hispanic': 4.707762598991394},
                     'dominant_race': 'white'}
```

Αποτέλεσμα "ΧΑΡΑ" από το DeepFace. Το DeepFace αξιολογεί την εικόνα με βάση τα έξι βασικά συναισθήματα (χαρά, λύπη, θυμός, φόβος, αηδία, έκπληξη) και την ουδετερότητα και επιστρέφει το κυριότερο συναίσθημα.

ΝΕΥΡΑ

```
In [34]: plt.imshow(cv2.cvtColor(img, 4))
Out[34]: <matplotlib.image.AxesImage at 0x14446b5e0>
```

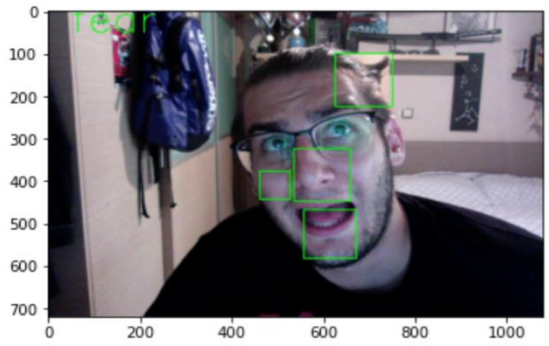



```
In [35]: predictions
Out[35]: {'emotion': {'angry': 78.33001613616943,
   'disgust': 4.90414109322046e-07,
   'fear': 0.09692359017208219,
   'happy': 0.0029844233722542413,
   'sad': 0.7374648470431566,
   'surprise': 0.0007920826647023205,
   'neutral': 20.831818878650665},
  'dominant_emotion': 'angry',
  'region': {'x': 396, 'y': 146, 'w': 368, 'h': 368},
  'age': 33,
  'gender': 'Man',
  'race': {'asian': 1.01453373208642,
   'indian': 9.244690835475922,
   'black': 1.2867345474660397,
   'white': 29.57858443260193,
   'middle eastern': 38.94897699356079,
   'latino hispanic': 19.92647796869278},
  'dominant_race': 'middle eastern'}
```

Αποτέλεσμα "ΝΕΥΡΑ" από το DeepFace. Το DeepFace αξιολογεί την εικόνα με βάση τα έξι βασικά συναισθήματα (χαρά, λύπη, θυμός, φόβος, αηδία, έκπληξη) και την ουδετερότητα και επιστρέφει το κυριότερο συναίσθημα.

ΦΟΒΟΣ

```
In [69]: plt.imshow(cv2.cvtColor(img, 4))
Out[69]: <matplotlib.image.AxesImage at 0x173ecb910>
```




```
In [70]: predictions
Out[70]: {'emotion': {'angry': 0.08828869433984823,
   'disgust': 0.00016924176663191535,
   'fear': 87.4174907901894,
   'happy': 12.050394767475751,
   'sad': 0.36407595351178007,
   'surprise': 0.00033012443666842033,
   'neutral': 0.0792447906980913},
  'dominant_emotion': 'fear',
  'region': {'x': 625, 'y': 100, 'w': 125, 'h': 125},
  'age': 24,
  'gender': 'Man',
  'race': {'asian': 0.8809936232864857,
   'indian': 1.1423341929912567,
   'black': 0.2333993325009942,
   'white': 72.96301126480103,
   'middle eastern': 16.88445806503296,
   'latino hispanic': 7.895802706480026},
  'dominant_race': 'white'}
```

Αποτέλεσμα "ΦΟΒΟΣ" από το DeepFace. Το DeepFace αξιολογεί την εικόνα με βάση τα έξι βασικά συναισθήματα (χαρά, λύπη, θυμός, φόβος, αηδία, έκπληξη) και την ουδετερότητα και επιστρέφει το κυριότερο συναίσθημα.

Συμπερασματικά, παρατηρήθηκε πως το deepface αποδίδει πολύ καλά στο πρόβλημα αναγνώρισης συναισθημάτων!!!

6)ΣΥΜΠΕΡΑΣΜΑΤΑ

- Στην παρούσα εργασία, η κατανόηση γνώσεων που αφορούν δομές και αρχές σχεδίασης νευρωνικών δικτύων καθώς και συνελικτικά νευρωνικά δίκτυα είναι ένα βασικό θέμα ακόμα και αν δεν απαιτείται η σχεδίαση ενός τέτοιου δικτύου, καθώς βοηθάει στο να γίνουν κατανοητές έννοιες και παράγοντες που μπορούν να επηρεάσουν την ακρίβεια των δεδομένων ή ακόμα και να δικαιολογήσουν τα αποτελέσματα. Μερικές τέτοιες έννοιες είναι η συνάρτηση ενεργοποίησης, τα φίλτρα, η σύνδεση μεταξύ των νευρώνων και άλλα. Στην διπλωματική εργασία, παρουσιάζονται όλες οι χρησιμοποιούμενες τεχνολογίες και περιβάλλοντα ανάπτυξης, όπως η μηχανική μάθηση και η βαθιά μάθηση καθώς και όλη η λειτουργικότητα. Επίσης, παρατίθενται και όλα τα απαραίτητα τμήματα κώδικα που αναπτύχθηκαν κατά την διάρκεια εκπόνησης της εργασίας.

- Γενικά όμως, μέσω της χρήσης διαφορετικών κορυφαίων μοντέλων και της παρατήρησης των αποτελεσμάτων, καταλαβαίνουμε πως πάντα θα υπάρχει χώρος για βελτίωση. Και είναι σημαντικό, καθώς αυτό σημαίνει πως μπορούμε συνεχώς να προχωράμε μπροστά και να πετυχαίνουμε ολοένα και καλύτερα αποτελέσματα!

Περιορισμοί

- Λόγω έλλειψης κατάλληλου εξοπλισμού(σταθερός υπολογιστής) δεν είχα την δυνατότητα να δημιουργήσω και να εκπαιδεύσω με ικανοποιητικά αποτελέσματα(ακρίβεια 85% και πάνω) εξ' ολοκλήρου ένα μοντέλο...!

7)ΠΑΡΑΠΟΜΠΕΣ

1. Andreas Muller : Introduction to Machine Learning with Python. Διαθέσιμο στο : https://files.isec.pt/DOCUMENTOS/SERVICOS/BIBLIO/INFORMAÇÕES%20ADICIONAIS/Introction-to-machine_Muller.pdf

2. François Chollet : Deep Learning with Python. Διαθέσιμο στο : https://www.manning.com/books/deep-learning-with-python

3. Machine Learning and Deep Learning for Emotion Recognition . Διαθέσιμο στο : https://upcommons.upc.edu/bitstream/handle/2117/184076/tfm-joan-sisquella.pdf?sequence=1&isAllowed=y

4. Facial Emotion Recognition Challenge Kaggle . Διαθέσιμο στο : https://www.kaggle.com/ashishpatel26/facial-expression-recognitionferchallenge

5. “DeepFace: Closing the Gap to Human-Level Performance in Face Verification”, Yaniv Taigman, Ming Yang, Marc’ Aurelio Ranzato, Lior Wolf. Διαθέσιμο στο: https://paperswithcode.com/paper/deepface-closing-the-gap-to-human-level-1

6. T. Dettmers, "Understanding Convolution in Deep Learning," 26 03 2015. Διαθέσιμο στο: http://timdettmers.com/2015/03/26/convolution-deep-learning/

7. Transfer Learning - Machine Learning's Next Frontier . Διαθέσιμο στο: https://ruder.io/transfer-learning/

8. The what's what of Keras and TensorFlow", Reetiesh Chandra Διαθέσιμο στο¨https://www.upgrad.com/blog/the-whats-what-of-keras-and-tensorflow/

9. Deepface,κατασκευήνευρωνικούδικτύου.Διαθέσιμο στο:https://github.com/serengil/deepface/blob/master/deepface/extendedmodels/Emotion.py

10. Keras Documentation. March, 2019. Διαθέσιμο στο: https://keras.io/examples/

11. Ronny Kohavi, Ronny Kohavi. Data mining tasks and methods: Classi- fication: decision-tree discovery Handbook of data mining and knowledge discovery Pages 267-276. Oxford University Press, Inc. New York, NY, USⒶc 2002

12. S.-Y. D. Bo-Kyeong Kim, Jihyeon Roh and S.-Y. Lee. Hierarchical committee of deep convolutional neural networks for robust facial expression recognition Journal on Multimodal User Interfaces, pages 1–17, 2015

13. N Srivastava, GE Hinton, A Krizhevsky, I Sutskever, R Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. Jour- nal of Machine Learning Research 15 (1), 1929-1958.

14. Jupyter Notebook. Διαθέσιμό στο :https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html

15. OpenCv. Διαθέσιμο στο : https://opencv.org

16. Τέχνητη Νοημοσύνη[Μάθημα] - ΠΑΔΑ

17. Μηχανική Μάθηση και Ανάλυση Μεγάλων Δεδομένων[Μάθημα] - ΠΑΔΑ

18. Facial Emotion Recognition. Διαθέσιμο στο : https://edps.europa.eu/system/files/2021-05/21-05-26_techdispatch-facial-emotion-recognition_ref_en.pdf

19. Emotion recognition: introduction to emotion recognition reading technology. Διαθέσιμο στο : https://recfaces.com/articles/emotion-recognition

20. Face Recognition-FaceNet. Διαθέσιμο στο: https://www.pluralsight.com/guides/face-recognition-walkthrough-facenet

21. Evaluation Metrics . Διαθέσιμο στο: https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226

22. Hyperparameter tuning. Διαθέσιμο στο: https://towardsdatascience.com/a-walkthrough-of-convolutional-neural-network-7f474f91d7bd