Ψηφιακή Επεξεργασία Εικόνας
Εργασία 2

Εύρεση γωνίας περιστροφής: 2
function findRotationAngle.m 2
Περιστροφη εικόνας: 3
rotateImage.m 3
Εύρεση περιγράμματος: 5
getcontour.m 5
Περιγραφέας περιγράμματος: 6
describer.m 6
Εξαγωγή γραμμάτων από την εικόνα κειμένου : 7
getletters.m 7
Εκπαίδευση αλγορίθμου: 8
mapletters.m 8
Δοκιμή αλγορίθμου: 9
testsystem.m 9
Τελικό αποτέλεσμα : 9
readtext.m 9
Ακρίβεια: 9
accuracy.m 9
Εφαρμογή 10
demo.m 10
Εικόνα εισόδου: text1.png (b=1) 10
Εικόνα εισόδου: text2.png 11

Γκατζή Κωνσταντίνα
ΑΕΜ: 10037
mail: gikonstan@ece.auth.gr

Η παρούσα εργασία έχει σκοπό την οπτική αναγνώριση χαρακτήρων (ocr) με χρήση μηχανικής μάθησης για την εκπαίδευση του αλγορίθμου. Συγκεκριμένα, η αναγνώριση κειμένου από εικόνα .png. Παρουσιάζονται οι συναρτήσεις που δημιουργήθηκαν, με τη σειρά που χρησιμοποίηθηκαν σύμφωνα με τα βήματα των οδηγιών της εκφώνησης.

Εύρεση γωνίας περιστροφής:

function findRotationAngle.m

Η εικόνα εισόδου δεν είναι απαραίτητο ότι θα δωθεί χωρίς να γραμματα να έχουν κλίση. Οπότε πριν ξεκινήσει η διαδικασία ocr πρέπει αυτό το πρόβλημα να έχει επιλυθεί. Αυτή η συνάρτηση παίρνει ως είσοδο την εικόνα και επιστρέφει τη γωνία που χρειάζεται να περιστραφεί. Αν αυτό το βήμα είναι περιττό, η γωνία που θα επιστραφεί θα είναι 0 μοίρες. Αρχικά η εικόνα μετατρέπεται από rgb σε gray scale, αφού το χρώμα δεν χρειάζεται σαν πληροφορία. Η εικόνα θολώνεται με τη χρήση της `imgaussfilt` έτσι ώστε οι γραμμές του κειμένου να ενωθούν και να είναι μόνο αυτές ξεκάθαρες στην εικόνα.
Παρουσιάζεται η εικόνα που προκύπτει από την εφαρμογή του φίλτρου στο text1

Image 1: Blurred image , lines are connected

Στην εικόνα αυτή, υπολογίζεται το μέτρο του διακριτού μετασχηματισμού Fourier για να υπολογιστεί η συχνότητα μεταβολής φωτεινότητας στο κείμενο από την οποία θα εξάγουμε το συμπέρασμα για τη γωνία περιστροφής. Θέλουμε να αναγνωρίσουμε τις γραμμές οπότε στη συνέχεια βρίσκουμε την προβολή του μετρου του μετασχηματισμού Fourier στον οριζόντιο άξονα και τοποθετούμε τις μέγιστες τιμές της σε έναν 2D πίνακα ίδιου μεγέθους με την εικόνα. Με την χρήση της `atan2` υπολογίζουμε μια πρώτη εκτίμηση της γωνίας. Συνήθως αυτή η εκτίμηση είναι λανθασμένη διότι ο μετασχηματισμός Fourier δεν είναι το ιδανικό εργαλείο για τον υπολογισμό κλίσης. Για παράδειγμα, με τη χρήση του μετασχηματισμού Hough, η εκτίμηση της γωνίας θα ήταν απόλυτα ακριβής.

Οπότε πραγματοποιείται στη συνέχεια μια σειριακή αναζήτηση γύρω από την εκτίμηση, σε ένα μεγάλο σχετικά εύρος. Αναιρείται η περιστροφή σε κάθε γωνία που εμπεριέχεται στο εύρος με την συνάρτηση `imrotate` (Η συνάρτηση αυτή χρησιμοποιείται μόνο στο loop. Στο τέλος η εικόνα περιστρέφεται σύμφωνα με τη γωνία που χρειάζεται με διαφορετικό τρόπο) Σε αυτό το loop υπολογίζεται κάθε φορά η προβολή της φωτεινότητας στον κάθετο άξονα, και συγκρίνεται με την φωτεινότητα της πραγματικής εικόνας με τη μεταβλητη `best_goodness_of_fit`. Τέλος είναι ξεκάθαρη η τιμή της πραγματικής γωνίας με την οποία χρειάζεται να περιστρέψουμε την εικόνα.

Περιστροφη εικόνας:

rotateImage.m

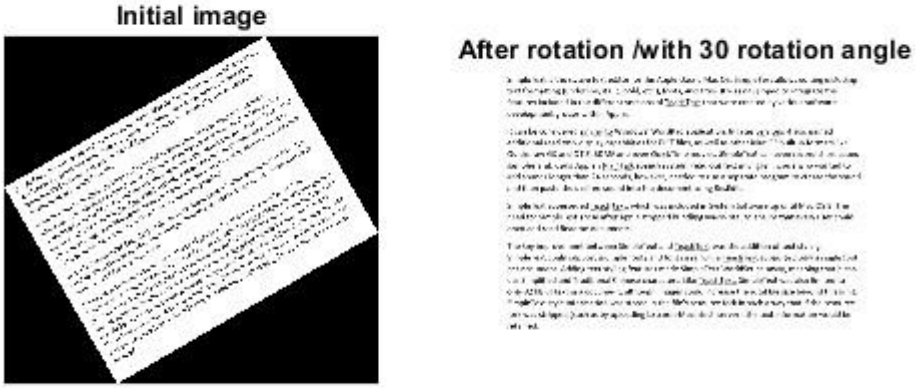
Σε συνεργασία με την προηγούμενη, αυτή η συνάρτηση περιστρέφει την εικόνα, έτσι ώστε τα γράμματα να εμφανίζονται οριζόντια, χωρίς κλίση. Έχει σαν είσοδο την εικόνα και τη γωνία, ενω επιστρέφει την εικόνα μετά την περιστροφή.
Αρχικά η εικόνα μετατρέπεται σε gray scale. Έπειτα υπολογίζεται ο πίνακας περιστροφής με τη συγκεκριμένη γωνία και εφαρμόζεται σαν γεωμετρικός μετασχηματισμός στην εικόνα με την συνάρτηση `imwarp`. Σε περίπτωση που η εικόνα είχε όντως κλίση, είναι λογικό το μέγεθος της διορθωμένης είναι μικρότερο από αυτό της αρχικής και εμφανίζεται με ένα μαύρο πλαίσιο γύρω της. Οπότε αφού βρω το σημείο που περιέχεται η εικόνα μέσα στο πλαίσιο, το περικόπτω και εφαρμόζω γραμμική παρεμβολή για να αυξήσω το μέγεθος της.

Παρουσιάζονται σε 2 plots η αρχική και η τελική εικόνα όταν η περιστροφή έγινε με 0 μοίρες

Image 2

Έπειτα σε περίπτωση που η εικόνα χρειαζόταν περιστροφη.

Image 3



Εύρεση περιγράμματος:

getcontour.m

Ο σκοπός αυτής της συνάρτησης είναι να δέχεται την εικόνα ενός γράμματος (ο τρόπος που εξάγονται τα γράμματα από την εικόνα κειμένου εξηγείται αργότερα) και να επιστρέφει ένα cell με τις συντεταγμένες των περιγραμμάτων του.
Αρχικά με την `bwconncomp` εντοπίζονται τα ενωμένα στοιχεία της εικόνας γράμματος με την τιμή 0, δηλαδή τα pixels που περιέχουν το γράμμα. Για την επεξεργασία που θα γίνει χρειάζεται να πάρουμε το συμπληρωματικό της εικόνας και έπειτα να ξανακάνουμε το ίδιο για να γυρίσουμε στο κανονικό.
Έπειτα υπολογίζεται η dilated εικόνα.

Image 4: Dilated letter image

Στη συνέχεια από την εικόνα αυτή αφαιρείται η αρχική και έτσι παίρνουμε το περίγραμμά της

Image 5: Outline dilated

Έπειτα το μόνο που χρειάζεται είναι να εφαρμόσουμε thinning στο περίγραμμα και να υπολογίσουμε το συμπληρωματικό. Παρουσιάζονται τα περιγράμματα για τα γράμματα “a”, “f”, “l” και “e”.

Image 6: Outline of letter a

Image 7: Outline of letter f

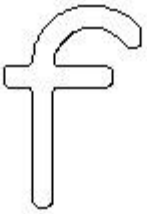


Image 8: Outline of letter l

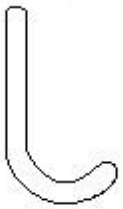
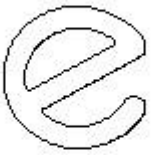


Image 9: Outline of letter e



Περιγραφέας περιγράμματος:

describer.m

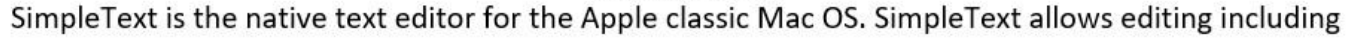
Η συνάρτηση αυτή προετοιμάζει τα περιγράμματα για να συγκριθούν. Δέχεται σαν είσοδο τις συντεταγμένες από ένα περίγραμμα μόνο. Το input max χρειάζεται έτσι ώστε σε κάθε τάξη περιγραμμάτων, οι περιγραφείς να έχουν το ίδιο μήκος για να μπορούν να συγκριθούν. Στις οδηγίες προτείνεται η εφαρμογή της συνάρτησης `interp1`, όμως στην συγκεκριμένη συνάρτηση, απλά πρόσθεσα μηδενικά στις ακολουθίες, στα στοιχεία που χρειαζόταν να συμπληρωθούν για να φτάσουν το μέγιστο μέγεθος που υπάρχει σε κάθε τάξη. Έπειτα, οι συντεταγμένες μετατρέπονται σε μιγαδικές ακολουθίες και υπολογίζεται ο διακριτός μετασχηματισμός Fourier τους, αφαιρώντας το πρώτο του στοιχείο.
Παρατήρησα ότι αυτή ήταν η καλύτερη επιλογή μεγέθους, έτσι ώστε να μη χάνεται καθόλου πληροφορία.

Εξαγωγή γραμμάτων από την εικόνα κειμένου :

getletters.m

Η συνάρτηση αυτή έχει σκοπό από την εικόνα .png που περιέχει το κείμενο να επιστρέφει ένα cell με όλες τις συντεταγμένες όλων των γραμμάτων και χαρακτήρων της εικόνας. Δέχεται σαν είσοδο την εικόνα και την μεταβλητή m, που δείχνει σε ποια γραμματοσειρά αναφερόμαστε. Αρχικά εφαρμόζεται η `findRotationAngle` και η `rotateImage` στην εικόνα έτσι ώστε να περιστραφεί αν χρειάζεται. Έπειτα αρχικοποιείται η μεταβλητή που αποδηκεύει τα δεδομένα της εξόδου και η μεταβλητη m που χρειάζεται για να μετρηθούν τα γράμματα της εικόνας.
Πρέπει να ξεχωρίσουμε τις περιοχές που περιέχουν γραμμές οπότε υπολογίζουμε την προβολή στον κάθετο άξονα και την κάνουμε πιο ομαλή με φίλτρο κινούμενου μέσου όρου. Χρειάζεται διαφορετικό για την δεύτερη γραμματοσειρά διότι οι γραμμές είναι πιο πυκνές. Θέτουμε όριο και ξεχωρίζουμε σε binary μεταβλητή τα σημεία που περνάνε το όριο από τα υπόλοιπα. Τα σημεία που το ξεπερνάνε θεωρούμε ότι είναι οι γραμμές. Εφόσον η πληροφορία αυτή είναι αποθηκευμένη σε binary μεταβλητή μπορώ να χρησιμοποιήσω την συνάρτηση `bwconncomp` για να βρω τα ενωμένα σημεία με τιμή 0 που περιέχουν τις συντεταγμένες των γραμμών. Εξάγω στην μεταβλητή props όλες αυτές τις περιοχές και συνεχίζω με loop στην καθεμία. Για κάθε γραμμή, με την πληροφορία των συντεταγμένων της κάνω περικοπή το σημείο της εικόνα που την περιέχει.

Image 10: Line image



Συνεχίζω εφαρμόζοντας μορφολογικό τελεστή open για να βελτιώσω την εικόνα γραμμής. Πρέπει τώρα να ξεχωρίσω τις λέξεις οπότε υπολογίζω την προβολή φωτεινότητας στον οριζόντιο άξονα, και κάνω την ίδια διαδικασία. Εφαρμόζω φίλτρο κινούμενου μέσου όρου, θέτω όριο, αποθηκεύω σε binary μεταβλητή τα σημεία που το ξεπερνάνε και βρίσκω τα ενωμένα σημεία που δηλώνουν τις συντεταγμένες των λέξεων μέσα σε κάθε γραμμή. Με επαναληπτική διαδικασία εξάγω κάθε εικόνα λέξης με περικοπή στα κατάλληλα σημεία.

Image 11: Word image

Τέλος πρέπει σε κάθε λέξη να ξεχωρίσω τα γράμματα. Εδώ παρουσιάστηκε πρόβλημα λόγω των υπογραμμισμένων λέξεων που δημιουργούσαν λανθασμένη προβολή φωτεινότητας της λέξης. Για να αντιμετωπιστεί αυτό το πρόβλημα, η εικόνα λέξης μετατράπηκε σε binary με το threshold που κρίθηκε κατάλληλο έτσι ώστε οι υπογραμμίσεις να έχουν την τιμή 1 και να μην επηρεάζουν τη λέξη.

Έτσι η διαδικασία έγινε με τον ίδιο τρόπο. Υπολογίστηκε η προβολή φωτεινότητας στον οριζόντιο άξονα και καταλήξαμε στο να έχουμε την εικόνα κάθε γράμματος ξεχωριστά. Παρατήρησα ότι στο 2ο κείμενο τα γράμματα ήταν υπερβολικά πυκνογραμμένα και χρειάστηκε ένα υπερβολικά μεγάλο όριο στη συνάρτηση προβολής φωτεινότητας για να επιλέξω τα σημεία που περιείχαν γράμματα.

Image 12: Letter image

Χρησιμοποιείται η συνάρτηση `padarray` έτσι ώστε να προστεθεί στην εικόνα γράμματος ένα λευκό pixel προς όλες τις διαστάσεις και να είναι πιο ξεκάθαρο το περίγραμμα χωρίς να το περιορίζουν οι διαστάσεις της εικόνας.
Καλείται η συνάρτηση `getcontour` και τα cells των συντεταγμένων του περιγράμματος κάθε γράμματος αποθηκεύονται στην μεταβλητή data.
Στο πρώτο κείμενο παρατήρησα ότι το σύμβολο του κέρσορα αποθηκεύτηκε σαν γράμμα οπότε το αφαίρεσα στο τέλος.

Εκπαίδευση αλγορίθμου:

mapletters.m

Αυτή η συνάρτηση έχει σκοπό να εκπαιδεύσει τον αλγόριθμο αναγνώρισης χαρακτήρων χρησιμοποιώντας το 70% του κειμένου (Train set) αφού ζητήθηκε να γίνει διαχωρισμός 70:30. Δέχεται σαν είσοδο την μεταβλητή d με τα cells που προέκυψαν από την προηγούμενη συνάρτηση και τη μεταβλητή που δηλώνει σε ποια από τις 2 γραμματοσειρές αναφέρομαι. Επιστρέφει 3 tables με τα δεδομένα που εκπαίδευσαν τον αλγόριθμο σε κάθε τάξη και τα 3 μέγιστα μεγέθη ακολουθιών για κάθε τάξη που θα χρησιμοποιηθούν στην συνάρτηση `describe` για να μετατρέψουν όλες τις ακολουθίες κάθε τάξης στο ίδιο μέγεθος και να μπορούν να συγκριθούν. Το maximum της 3ης τάξης αρχικοποιείται γιατί υπάρχει περίπτωση να μη βρεθεί κανένα γράμμα με 3 περιγράμματα. Έπειτα κρατάμε μόνο το 70% των γραμμάτων του κειμένου και αποθηκεύουμε το πλήθος τους.
Διαβάζουμε το κείμενο που αντιστοιχεί σε κάθε γραμματοσειρά και εφαρμόζονται κάποιες τροποποιήσεις για να αφαιρέσουμε τα κενά και να το μετατρέψουμε σε array of chars.
Στη μεταβλητή labels αποθηκεύουμε την τάξη κάθε γράμματος χρησιμοποιώντας το πλήθος των cells που χρειάζεται για το περίγραμμά του. Για κάθε τάξη αποθηκεύω όλα τα μήκη ακολουθιών και βρίσκω τα μέγιστα για τον σκοπό που εξηγήθηκε παραπάνω. Προσέχω ξανά την περίπτωση του να μην υπάρχει κανένα γράμμα με 3 περιγράμματα.
Για κάθε γράμμα αποθηκεύω σε διαφορετική μεταβλητή για κάθε τάξη τους περιγραφείς και τα αντίστοιχα γράμματα του .txt κειμένου και δημιουργώ τα 3 tables που αντιστοιχούν περιγραφείς με χαρακτήρες. Έτσι ο αλγόριθμος μηχανικής μάθησης έχει αρκετά δεδομένα για να αναγνωρίσει με τη μέθοδο κοντινότερων γειτόνων οποιοδήποτε γράμμα της

συγκεκριμένης γραμματοσειράς. Στο τέλος θα γίνει και προσπάθεια να αναγνωριστούν γράμματα άλλης γραμματοσειράς.

Δοκιμή αλγορίθμου:

testsystem.m

Αυτή η συνάρτηση έχει σκοπό να δοκιμάσει τον εκπαιδευμένο αλγόριθμο στο υπόλοιπο κείμενο και δέχεται σαν όρισμα τις συντεταγμένες κάθε γράμματος, τα 3 tables και τις maximum τιμές που υπολογίστηκαν για κάθε τάξη. Εφαρμόζεται σε ολόκληρο το κείμενο, όχι μόνο στο τελευταίο 30%, έτσι ώστε να παρουσιαστεί σαν αποτέλεσμα ολόκληρο. Ξεκινάμε αποθηκεύοντας το πλήθος των γραμμάτων και ταξινομώντας τα με labels. Για κάθε γράμμα αρχικοποιείται η μεταβλητή distance μεταξύ των περιγραφέων και σκανάρεται όλο το table της αντίστοιχης τάξης, για να βρεθεί ο χαρακτήρας με τον οποίο ο περιγραφέας του γράμματος έχει το μικρότερο τετραγωνικό σφάλμα. Όλοι οι χαρακτήρες αποθηκεύονται στην μεταβλητή εξόδου, όπως και τα labels.

Τελικό αποτέλεσμα :

readtext.m

Αυτή η συνάρτηση έχει σκοπό να εφαρμόζει τις προηγούμενες και να παρουσιάζει το τελικό αποτέλεσμα. Δέχεται σαν είσοδο την εικόνα .png που περιέχει το κείμενο, τη μεταβλητή a που δηλώνει τη γραμματοσειρά και την μεταβλητή b που δηλώνει τα δεδομένα στα οποία θα δοκιμάσω το σύστημα (η χρήση της θα εξηγηθεί πιο αναλυτικά αργότερα). Δίνει σαν έξοδο τα chars από κάθε εικόνα γράμματος και τα αντίστοιχα labels. Αρχικά καλεί τη συνάρτηση `getletters` και την `mapletters` . Αν η μεταβλητή b έχει την τιμή 3 χρησιμοποιείται η εικόνα text2 για αναγνώριση, πάνω σε αλγόριθμο εκπαιδευμένο στην εικόνα text1. Αλλιώς η εκπαίδευση και η αναγνώριση γίνονται στην ίδια γραμματοσειρά.

Ακρίβεια:

accuracy.m

Η συνάρτηση αυτή υπολογίζει την ακρίβεια της μεθόδου αναγνώρισης. Δέχεται σαν είσοδο το κείμενο .txt, τον πίνακα chars που υπολογίσαμε και τα labels. Αρχικά αφαιρεί τα κενά από το κείμενο και υπολογίζει τον πίνακα confusion_matrix που στην πρώτη του στήλη έχει τους αριθμούς από τους χαρακτήρες που ταξινομήθηκαν σωστά για κάθε τάξη και στην δεύτερη,

αυτούς που αναγνωρίστηκαν λάθος. Ο πίνακας weighted_accuracy έχει τους μέσους όρους των σωστών για κάθε τάξη.

Εφαρμογή

demo.m

Εικόνα εισόδου: text1.png (b=1)

Στην περίπτωση που θέτω ως εικόνα εισόδου την text1.png, εκπαιδεύω το σύστημα με αυτήν την εικόνα και έπειτα προσπαθώ να το αναγνωρίσω, το κείμενο που παράγεται είναι αυτό:

Text is:
SimpleTextisthenativetexteditorfortheAppleclassicMacOS.SimpleTextallowseditingincludingtextformatting(underline,italic,bold,etc.),fonts,andsizes.ItwasdevelopedtointegratethefeaturesincludedinthedifferentversionsofTeachTextthatwerecreatedbyvarioussoftwaredevelopmentgroupswithinApple.ItcanbeconsideredsimilartoWindows'WordPadapplication.InlaterversionsitalsogainedadditionalreadonlydisplaycapabilitiesforPICTfiles,aswellasotherMacOSbuilt-informatslikeQuickdrawGXandQTIF,3DMFandevenQuickTimemovies.SimpleTextcanevenrecordshortsoundsamplesand,usingApple'sPlainTalkspeechsystem,readouttextinEnglish.Userswhowantedtoaddsoundslongerthan24seconds,however,neededtouseaseparateprogramtocreatethesoundandthenpastethedesiredsoundintothedocumentusingResEdit.SimpleTextsupersededTeachText,whichwasincludedinSystemSoftwareuptilMacOS8.TheneedforSimpleTextaroseafterApplestoppedbundlingMacWrite,toensurethateveryusercouldopenandreadReadmedocuments.ThekeyimprovementbetweenSimpleTextandTeachTextwastheadditionoftextstyling.SimpleTextcouldsupportmultiplefontsandfontsizes,whileTeachTextsupportedonlyasinglefontperdocument.AddingtextstylingfeaturesmadeSimpleTextWorldScript-savvy,meaningthatitcanuseSimplifiedandTraditionalkhinesecharacters.sikeTeathTent,SimpleTextwasalsolimitedtoonlya2k8oftextinadocument,nalthughimagescouldincreasethetotalfilesizebeyondthisli

mit.SimpleTextstyleinformationwasstoredinthefile'sresourceforkinsuchawaythatiftheresourceforkwasstripped(suchasbyuploadingtoanon-Macintoshserver),thetextinformationwouldberetained.

Οι πίνακες είναι οι εξής:

```
Confusion matrix is:
   857     8
   625     1
    32     2

weighted_accuracy is:
    0.9908
    0.9984
    0.9412
```

Εικόνα εισόδου: text2.png

Σε αυτήν την περίπτωση που κάνω την εκπαίδευση και την αναγνώριση σε αυτό το κείμενο τα αποτελέσματα είναι τα εξής:

Text is:
thebattleofthecoralsea,foughtduring4-8may1942,wasamajornavalbattleinthepacifictheaterofworldwariibetweentheimperialjapanesenavy(ijn)andnavalandairforcesfromtheunitedstatesandaustralia.thebattlewasthefirstactioninwhichaircraftcarriersengagedeachother,aswellasthefirstinwhichneithersside'sshipssightedorfireddirectlyupontheother.inanattempttostrengthentheirdefensivepositioningfortheirempireinthesouthpacific,japaneseforcesdecidedtoinvadeandoccupyportmoresbyinnewguineaandtulagiinthesoutheasternsolomonislands.theplantoaccomplishthis,calledoperationmo,involvedseveralmajorunitsofjapan'scombinedfleet,includingtwofleetcarriersandalightcarriertoprovideaircoverfortheinvasionfleets,undertheoverallcommandofjapaneseadmiralshigeyoshiinoue.theuslearnedofthejapaneseplanthroughsignalsintelligenceandsentt

wounitedstatesnavycarriertaskforcesandajointaustralian-americancruiserforce,undertheoverallcommandofamericanadmiralfrankj.fletcher,toopposethejapaneseoffensive.on3-4may,japaneseforcessuccessfullyinvadedandoccupiedtulagi,althoughseveraloftheirsupportingwarshipsweresurprisedandsunkordamagedbyaircraftfromtheusfleetcarrieryorktown.nowawareofthepresenceofuscarriersinthearea,thejapanesefleetcarriersadvancedtowardsthecoralseawiththeintentionoffindinganddestroyingthealliednavalforces.beginningon7may,thecarrierforcesfromthetwosidesexchangedinairstrikesovertwoconsecutivedays.thefirstday,theussankthejapaneselightcarriershoho,whilethejapanesesankausdestroyerandheavilydamagedafleetoiler(whichwaslaterscuttled).theneotday,thejapanesefleetcarriershokakuwasheavilydamaged,theusfleetcarrierlexingtonwascriticallydamaged(andwasscuttledasaresult),andtheyorktownwasdamaged.withbothsideshavingsufferedheavylossesinaircraftandcarriersdamagedorsunk,thetwofleetsdisengagedandretiredfromthebattlearea.becauseofthelossofcarrieraircover,inouerecalledtheportmoresbyinvasionfleet,intendingtotryagainlater.

```
Confusion matrix is:
        1729           3
         147           0
           0           0

weighted_accuracy is:
    0.9983
    1.0000
       NaN
```

Παρατήρησα ότι επειδή τα γράμματα ήταν αρκετά πυκνά και χωρίς καλή ανάλυση τα περισσότερα περιγράμματα θεωρήθηκαν ένα αντί για 2. Η εικόνα των περιγραμμάτων ήταν υπερβολικά μικρή σε διαστάσεις έτσι τα pixels δεν μπορούσαν να διαχωριστούν. Όμως αν ο αλγόριθμος εκπαιδευτεί σε αυτό το κείμενο μπορεί να τα αναγνωρίσει, απλά δεν έχει πολλά σύμβολα στις τάξεις εκτός της 1ης.