Ταξινόμηση του Modelnet10 με την αρχιτεκτονική του δικτύου PointNet και επανεκπαίδευση του με την τεχνική transfer learning στο Modelnet40

Ανανιάδου Χριστίνα, Καγιόγλου Παναγιώτα

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
ΔΠΜΣ Προηγμένα Συστήματα Υπολογιστών και Επικοινωνιών, Φεβρουάριος 2021

Περιληψη

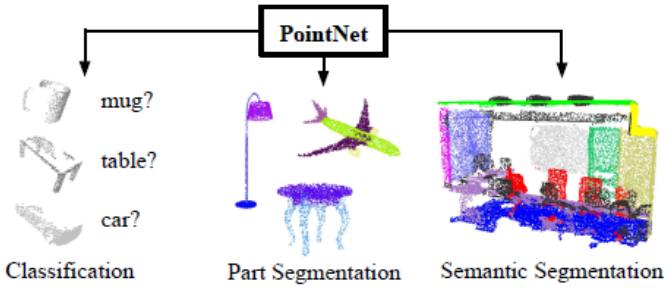
Στη συγκεκριμένη εργασία εξετάστηκαν τρισδιάστατα γεωμετρικά δεδομένα, νέφη σημείων, και ταξινομήθηκαν με *τη χρήση του δικτύου αρχιτεκτονικής PointNet. Το σύνολο δεδομένων που χρησιμοποιήθηκε για την εκπαίδευση του δικτύου είναι το Modelnet10. Έπειτα ακολούθησε αξιολόγηση στο σύνολο δοκιμής και εκπαίδευσης για διαφορετικές επιλύσεις του δικτύου. Στο δεύτερο μέρος της εργασίας έγινε χρήση του παραπάνω δικτύου και με βάση την τεχνική του transfer learning επαναεκπαιδεύτηκε στο σύνολο δεδομένων Modelnet40. Μετα την εκπαίδευση του αξιολογήθηκε στο σύνολο δοκιμής.*

Λέξεις κλειδιά—συνελικτικά νευρωνικά δίκτα, ταξινόμηση, νέφος σημείων, transfer learning

I. Εισαγωγη

Σε αυτή την εργασία διερευνούμε αρχιτεκτονικές deep learning ικανές να επιλύσουν προβλήματα που αφορούν τρισδιάστατα γεωμετρικά δεδομένα όπως νέφη σημείων. Τα τυπικά συννελικτικά δίκτυα απαιτούν συνηθισμένη μορφή δεδομένων όπως πλέγματα εικόνων ή 3D voxels προκειμένου να μπορούν να αποδώσουν σωστά τα βάρη. Στην περίπτωση όμως του νέφους σημείων, επειδή δεν είναι μια κοινή μορφή δεδομένων, οι περισσότεροι ερευνητές συνήθως καταφεύγουν στη μετατροπή τους σε κανονικά πλέγματα 3D voxel πριν τα τροφοδοτήσουν στο νευρωνικό δίκτυο. Αυτός ο μετασχηματισμός όμως έχει ως αποτέλεσμα την άσκοπη δημιουργία μεγάλου όγκου δεδομένων. Για το λόγο αυτό έχει δημιουργηθεί μια διαφορετική αναπαράσταση εισόδου για την περίπτωση των σημείων νέφους και ονομάζεται PointNet.

Το PointNet δέχεται νέφη σημείων ως είσοδο και έχει ως έξοδο είτε την ετικέτα της κλάσης για ολόκληρο το νέφος είτε ανά σημείο ή τμήμα της εισόδου. Η βασική αρχιτεκτονική του δικτύου είναι αρκετά απλή καθώς κάθε σημείο αναπαρίσταται με τις συντεταγμένες (x,y,z) και επεξεργάζεται ανεξάρτητα και με τον ίδιο τρόπο. Κλειδί για την αρχιτεκτονική του δικτύου αποτελεί η χρήση της μεθόδου max pooling. Το δίκτυο μαθαίνει αποτελεσματικά τα πιο αντιπροσωπευτικά χαρακτηριστικά και κωδικοποιεί τους λόγους επιλογής. Τέλος, τα fully connected



Εικόνα 1. ***Εφαρμογές PointNet***

layers συλλέγουν τη πληροφορία και τη μεταφέρουν στον τελικό ταξινομητή.

Στόχος αυτής της εργασίας είναι η χρήση της αρχιτεκτονικής PointNet στην επίλυση δυο διαφορετικών προβλημάτων. Το πρώτο αφορά την ταξινόμηση του σετ δεδομένων ModelNet10 και τη σύγκριση αποτελεσμάτων τροποποιώντας ορισμένες παραμέτρους όπως ο optimizer και ο ρυθμός μάθησης. Το δεύτερο πρόβλημα κάνει χρήση του προηγούμενου δικτύου και με την τεχνική transfer learning επαναεκπαιδεύεται στο σύνολο δεδομένων ModelNet40. Το σύνολο δεδομένων ModelNet10 αποτελείται από νέφη σημείων που ανήκουν τις εξής 10 κλάσεις: {bathtub, bed, chair, desk, dresser, monitor, night_stand, sofa, table, toilet}. Ενώ το σύνολο δεδομένων Modelnet40 αποτελείται από νέφη σημείων που ανήκουν στις προηγούμενες 10 και σε επιπλέον 30 κλάσεις.

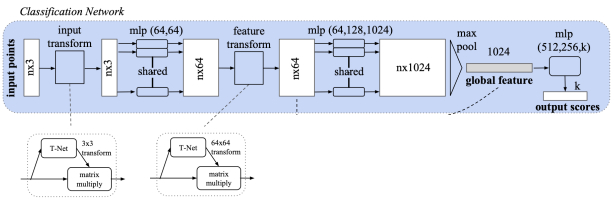
II. Περιγραφη Δικτυου

A. Προεπεξεργασία δεδομένων

Τα νέφη σημείων πριν εισαχθούν στο μοντέλο που χρησιμοποιεί την αρχιτεκτονική PointNet τροποποιήθηκαν κατάλληλα, ώστε ο αλγόριθμος εκπαίδευσης να μπορεί να γενικεύει καλύτερα. Πιο συγκεκριμένα, χρησιμοποιήθηκαν τεχνικές όπως transform, normalization και augmentation. Κατά την διαδικασία του transform ο αλγόριθμος επιλέγει 1024 σημεία από τις επιφάνειες του αντικειμένου που μελετάται. Το normalization γίνεται με σκοπό να εκφραστούν όλα τα

δεδομένα εντός συγκεκριμένων αριθμητικών ορίων κατά του άξονες x,y,z με σκοπό να μην προκύψουν προβλήματα σύγκλισης στον αλγόριθμο. Τέλος οι μελετητές του PointNet εφάρμοσαν την διαδικασία του augmentation on-the-fly στα δεδομένα του μοντέλου τους. Αναλυτικότερα, τα περιστρέψανε τυχαία κατά τον άξονα z ενώ ταυτόχρονα εισήγαν θόρυβο μέσω μιας gaussian κατανομής με μηδενική διάμεσο και 0.002 τυπική απόκλιση. Συνοψίζοντας, λόγω αυτών των τυχαίων μετασχηματισμών, δεν λαμβάνουμε τις ίδιες εικόνες κάθε φορά με αποτέλεσμα το μοντέλο αν εκπαιδεύεται σε διαφορετικού τύπου δεδομένα και να γενικεύει καλύτερα.

B. Αρχιτεκτονική δικτύου



*Εικόνα 2. **Αρχιτεκτονική δικτύου***

Το PointNet μοντέλο μπορεί να χωριστεί σχηματικά σε τρία μέρη. Το πρώτο μέρος του μοντέλου σχετίζεται με το Input και Feature transform και έχει την ακόλουθη μορφή :

Input transform > k=3 > TNet > Feature transform > k=64 > TNet

όπου TNet είναι ένα layer με την ακόλουθη μορφή:

- Convolutional
    1. (k,64,1)
    2. (64,128,1)
    3. (128,1024,1)
- Fully Connected
    1. (1024,512)
    2. (512,256)
    3. (256,k*k)
- Batch Normalization
    1. (64)
    2. (128)
    3. (124)
    4. (512)
    5. (256)

Το δεύτερο μέρος σχετίζεται με την αναγνώριση των χαρακτηριστικών των δεδομένων εκπαίδευσης και έχει την ακόλουθη μορφή:

Conv1 (3,64,1) > Bn1(64) > Conv2(64,128,1) > Bn2(128) > Conv3(128,1024,1) > Bn3(1024)

Το τρίτο και τελευταίο μέρος του δικτύου σχετίζεται με την δημιουργία των fully connected layers και κάνει την πρόβλεψη σε ποια κλάση ανήκει το αντικείμενο που εισάγεται σαν είσοδο.

FC1 (1024,512) > BN1(512) > BN2(256) > DropOut (30%) > LongSoftMax (dim=1) > FC3(256, Αριθμός κλάσεων)

Transfer Learning

Transfer learning είναι μια διαδικασία κατά την οποία γίνεται χρήση της «γνώσης», που αποκτήθηκε κατά την επίλυσή ενός μοντέλου Α, σε ένα μοντέλο Β το οποίο καλείται να λύσει ένα παρόμοιο πρόβλημα με το μοντέλο Α. Στην παρούσα εργασία χρησιμοποιήθηκε η γνώση του μοντέλου PointNet10 που εκπαιδεύτηκε στο σύνολο δεδομένων Modelnet10, ώστε να εκπαιδευτεί το σύνολο δεδομένων Modelnet40 με την τεχνική transfer learning.
Πιο συγκεκριμένα όπως φαίνεται και στην Εικόνα 3 κατά την διαδικασία της εκπαίδευσης ο αλγόριθμος μπορούσε να τροποποιήσει μόνο τις τιμές βαρών και bias των τελευταίων fully connected layers για δεδομένο αριθμό εποχών (τα layers που είναι εκτός των άγκιστρων στην Εικόνα 3). Πιο συγκεκριμένα δημιουργήθηκαν δύο μοντέλα. Στο πρώτο μοντέλο, ο αλγόριθμος «ξεπάγωνε» όλες τις τιμές των παραμέτρων μετά το πέρας της 7ης εποχής ενώ στο δεύτερο μετά την 2η. Επιπρόσθετα ξανά ορίστηκε ο αριθμός των εξόδων του τελευταίου fully connected layer, με σκοπό ο αλγόριθμος να προβλέπει 40 αντί για 10 κλάσεις. Τέλος το μοντέλο στο οποίο έγινε το unfreeze στο τέλος της δεύτερης εποχής, επιλύθηκε για learning rate 0,001 και 0,01.

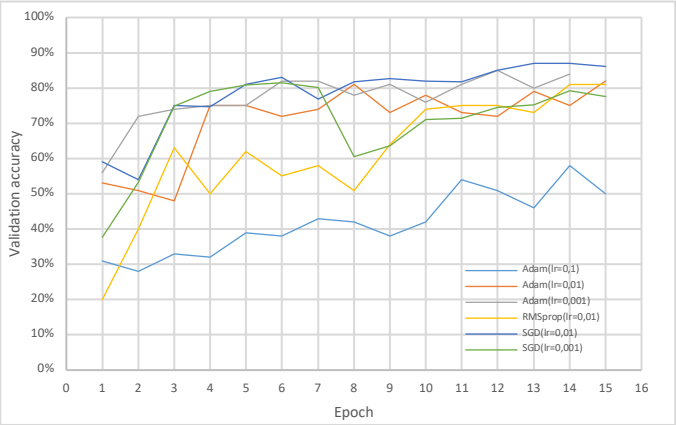
```
PointNet(
  (transform): Transform(
    (input_transform): Tnet(
      (conv1): Conv1d(3, 64, kernel_size=(1,), stride=(1,))
      (conv2): Conv1d(64, 128, kernel_size=(1,), stride=(1,))
      (conv3): Conv1d(128, 1024, kernel_size=(1,), stride=(1,))
      (fc1): Linear(in_features=1024, out_features=512, bias=True)
      (fc2): Linear(in_features=512, out_features=256, bias=True)
      (fc3): Linear(in_features=256, out_features=9, bias=True)
      (bn1): BatchNorm1d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn2): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn3): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn4): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn5): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (feature_transform): Tnet(
      (conv1): Conv1d(64, 64, kernel_size=(1,), stride=(1,))
      (conv2): Conv1d(64, 128, kernel_size=(1,), stride=(1,))
      (conv3): Conv1d(128, 1024, kernel_size=(1,), stride=(1,))
      (fc1): Linear(in_features=1024, out_features=512, bias=True)
      (fc2): Linear(in_features=512, out_features=256, bias=True)
      (fc3): Linear(in_features=256, out_features=4096, bias=True)
      (bn1): BatchNorm1d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn2): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn3): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn4): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      (bn5): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    )
    (conv1): Conv1d(3, 64, kernel_size=(1,), stride=(1,))
    (conv2): Conv1d(64, 128, kernel_size=(1,), stride=(1,))
    (conv3): Conv1d(128, 1024, kernel_size=(1,), stride=(1,))
    (bn1): BatchNorm1d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (bn2): BatchNorm1d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (bn3): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (fc1): Linear(in_features=1024, out_features=512, bias=True)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc3): Linear(in_features=256, out_features=10, bias=True)
  (bn1): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (bn2): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (dropout): Dropout(p=0.3, inplace=False)
  (logsoftmax): LogSoftmax(dim=1)
)
```

*Εικόνα 3. **Τεχνική transfer learning (αποτύπωση των layer του μοντέλου PointNet10)***

III. ΑΠΟΤΕΛΕΣΜΑΤΑ

Με τη χρήση της αρχιτεκτονικής του δικτύου PointNet το μοντέλο μας εκπαιδεύτηκε στο σύνολο δεδομένων Modelnet10. Ακολούθησαν έξι διαφορετικές επιλύσεις τροποποιώντας τον optimizer και τον ρυθμό μάθησης. Πιο συγκεκριμένα, δοκιμάστικαν ο optimizer Adam με ρυθμό μάθησης 0,001, 0,01 και 0,1 , ο optimizer SGD με ρυθμό μάθησης 0,001 και 0,01 και τέλος ο optimizer RMSprop με

ρυθμό μάθησης 0,01. Τα συγκεντρωτικά αποτελέσματα παρουσιάζονται παρακάτω.



Εικόνα 4. ***Validation accuracy ανά εποχή για ModelNet10***

Στο Εικόνα 4 αποτυπώνεται το accuracy στο σύνολο επικύρωσης σε σχέση με τις εποχές για τις διαφορετικές επιλύσεις που προαναφέρθηκαν. Όπως φαίνεται η καλύτερη επίδοση επιτυγχάνεται με τον optimizer SGD και ρυθμό μάθησης (lr = 0,01) αφού το accuracy φτάνει έως και 87% . Ενώ η χειρότερη επίδοση γίνεται με τον optimizer Adam και ρυθμό μάθησης (lr = 0,1) με τελικό validation accuracy 58%. Επίσης παρατηρούμε πως οι optimizers RMSprop (lr = 0,01) και SGD (lr = 0,001) έχουν τη μεγαλύτερη "αστάθεια" αφού η διακύμανση τους είναι η μεγαλύτερη. Πιο συγκεκριμένα για τον πρώτο το χαμηλότερο accuracy ξεκινάει από 20% και φτάνει 81% ενώ για τον δεύτερο ξεκινάει από 37% και φτάνει 82%. Ωστόσο, επειδή το accuracy από μόνο του δεν αποτελεί επαρκή μετρική για διεξαγωγή συμπερασμάτων, θα εξετάσουμε και άλλες μετρικές όπως precision, recall και f1-score μέσω του classification report.

	precision	recall	f1-score	support
bathtub	0.93	0.80	0.86	50
bed	0.86	0.96	0.91	100
chair	0.85	1.00	0.92	100
desk	0.84	0.37	0.52	86
dresser	0.51	0.47	0.49	86
monitor	1.00	0.74	0.85	100
night_stand	0.53	0.90	0.66	86
sofa	0.94	0.96	0.95	100
table	0.81	0.93	0.87	100
toilet	0.99	0.82	0.90	100
accuracy			0.80	908
macro avg	0.83	0.79	0.79	908
weighted avg	0.83	0.80	0.80	908

Adam lr = 0,001

	precision	recall	f1-score	support
bathtub	1.00	0.70	0.82	50
bed	0.83	0.85	0.84	100
chair	0.96	0.93	0.94	100
desk	0.75	0.60	0.67	86
dresser	0.56	0.65	0.60	86
monitor	0.98	0.89	0.93	100
night_stand	0.57	0.67	0.62	86
sofa	0.91	0.92	0.92	100
table	0.74	0.87	0.80	100
toilet	0.97	0.90	0.93	100
accuracy			0.81	908
macro avg	0.83	0.80	0.81	908
weighted avg	0.83	0.81	0.82	908

Adam lr = 0,01

	precision	recall	f1-score	support
bathtub	0.94	0.30	0.45	50
bed	0.90	0.89	0.89	100
chair	0.92	0.91	0.91	100
desk	0.77	0.59	0.67	86
dresser	0.61	0.81	0.70	86
monitor	0.83	0.93	0.88	100
night_stand	0.89	0.36	0.51	86
sofa	0.67	0.97	0.80	100
table	0.76	0.90	0.82	100
toilet	0.88	0.92	0.90	100
accuracy			0.79	908
macro avg	0.82	0.76	0.75	908
weighted avg	0.81	0.79	0.78	908

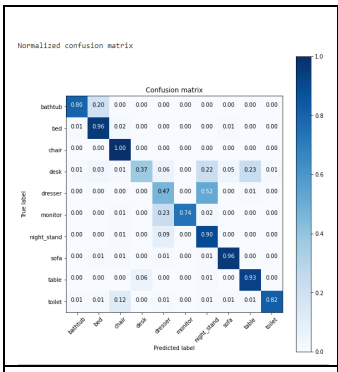
SGD lr = 0,001

	precision	recall	f1-score	support
bathtub	0.94	0.88	0.91	50
bed	0.96	0.86	0.91	100
chair	0.92	0.99	0.95	100
desk	0.64	0.81	0.72	86
dresser	0.69	0.92	0.79	86
monitor	0.93	0.95	0.94	100
night_stand	0.86	0.58	0.69	86
sofa	0.94	0.97	0.96	100
table	0.85	0.69	0.76	100
toilet	0.99	0.95	0.97	100
accuracy			0.86	908
macro avg	0.87	0.86	0.86	908
weighted avg	0.87	0.86	0.86	908

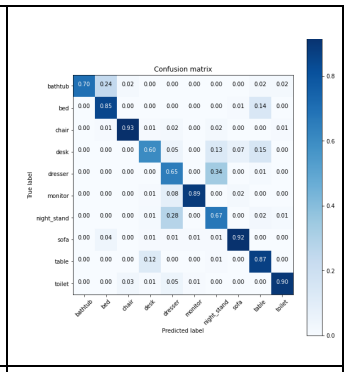
SGD lr = 0,01

Εικόνα 5. ***Classification report για Adam και SGD optimizers***

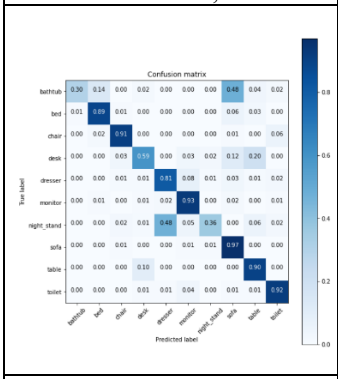
Στην Εικόνα 5 παρατηρούμε ότι για optimizer Adam και ποσοστό μάθησης (lr = 0,001) οι κλάσεις desk και dresser έχουν πολύ χαμηλό recall 0,37 και 0,47 αντίστοιχα. Αυτό σημαίνει ότι λιγότερα από τα μισά δείγματα που ανήκουν στις κλάσεις desk και dresser θα αναγνωριστούν ότι πράγματι ανήκουν σ' αυτές. Πιο αναλυτικά, όπως φαίνεται και στην Εικόνα 6 για την περίπτωση του desk μεγάλο ποσοστό αναγνωρίζεται ως night_stand και table, ενώ για την περίπτωση του dresser η πλειοψηφία αναγνωρίζεται ως night_stand. Από την άλλη, για ποσοστό μάθησης (lr = 0,01) για τις κλάσεις desk και dresser παρατηρούνται πιο ικανοποιητικά ποσοστά recall 0,60 και 0,65 αντίστοιχα. Έτσι παρόλο που στην Εικόνα 4 το validation accuracy φτάνει σε υψηλότερο ποσοστό, 85% ,για lr = 0,001 από ότι για lr = 0,01 που είναι 82%, καταλήγουμε να θεωρούμε το μοντέλο με lr = 0,01 καλύτερο καθώς έχει πιο ικανοποιητικές μετρικές για όλες τις κλάσεις, σε αντίθεση με το μοντέλο με lr = 0,001 που αδυνατεί να ταξινομήσει σωστά τις κλάσεις desk και dresser σε ποσοστό μεγαλύτερο του 50%.



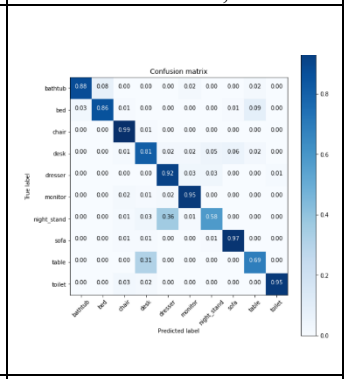
Adam lr = 0,001



Adam lr = 0,01



SGD lr = 0,001



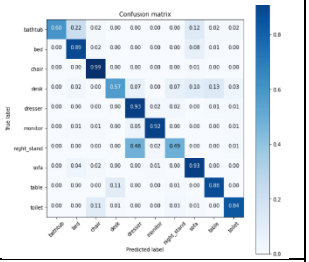
SGD lr = 0,01

Εικόνα 6. ***Confusion matrix για Adam και SGD optimizers***

Για την περίπτωση του optimizer SGD τα αποτελέσματα που αποτυπώνονται στην Εικόνα 5 δείχνουν ότι για lr = 0,001 οι κλάσεις bathtub και night_stand δεν έχουν καλό ποσοστό recall, 0,3 και 0,36 αντίστοιχα. Όπως φαίνεται και στο confusion matrix η κλάση bathtub αποτυπώνεται κατά 48% ως sofa ενώ η κλάση night_stand κατά 48% επίσης ως dresser. Αντίθετα για lr = 0,01 θα λέγαμε πως οι μετρικές είναι καλύτερες για όλες τις κλάσεις εκτός από αυτή του table που το recall πέφτει από 0,90 σε 0,69. Συγκεντρωτικά όμως το f1-score

αυξάνεται από 79% σε 86%. Για αυτό το λόγο θεωρούμε θεωρούμε καλύτερη την απόδοση του SGD με lr = 0,01.

	precision	recall	f1-score	support
bathtub	1.00	0.60	0.75	50
bed	0.83	0.89	0.86	100
chair	0.85	0.99	0.92	100
desk	0.80	0.57	0.67	86
dresser	0.61	0.93	0.73	86
monitor	0.95	0.92	0.93	100
night_stand	0.78	0.49	0.60	86
sofa	0.79	0.93	0.85	100
table	0.86	0.88	0.87	100
toilet	0.92	0.84	0.88	100
accuracy			0.82	908
macro avg	0.84	0.80	0.81	908
weighted avg	0.84	0.82	0.82	908



RMSprop lr = 0,01

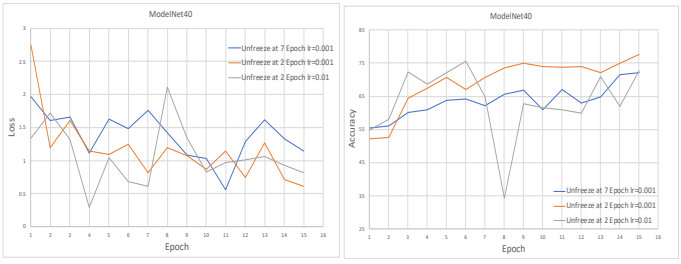
Εικόνα 7 ***Classification report και confusion matrix για RMSprop optimizer***

Όσον αφορά την περίπτωση του optimizer RMSprop δοκιμάστηκε μόνο για ποσοστό μάθησης lr = 0,01. Βλέπουμε στην Εικόνα 7 πως έχει γενικά καλές μετρικές με εξαίρεση την κλάση night_stand με recall 0,49 και f1-score 0,60 και την κλάση desk που είναι οριακά καλή με recall 0,57 και f1-score 0,67. Πιο αναλυτικά, για την κλάση night_stand παρατηρούμε και στο confusion matrix πως τις μισές φορές θα αποδωθεί σωστά η ετικέτα και τις υπόλοιπες μισές θα κατηγοριοποιηθεί ως dresser. Διακρίνουμε επομένως πως το συγκεκριμένο μοντέλο έχει μεγάλη αστοχία να εντοπίσει την κλάση night_stand.

Τέλος, για την περίπτωση του optimizer Adam με lr = 0,1 τα αποτελέσματα των confusion matrix και classification report συμφωνούν με την καμπύλη του validation accuracy ανα εποχή καθώς δείχνουν πως το μοντέλο αδυνατεί να κατηγοριοποιήσει σωστά παραπάνω από τις μισές κλάσεις.

Συμπερασματικά, με βάση τα classification report, confusion matrix και την καμπύλη validation accuracy ανά εποχή στην Εικόνα 4 καταλήγουμε πως το μοντέλο με optimizer SGD και ποσοστό μάθησης (lr = 0,01) είναι το καλύτερο για την επίλυση του προβλήματος ταξινόμησης του συνόλου δεδομένων Modelnet10 κάνοντας χρήση την αρχιτεκτονική του δικτύου PointNet.

Transfer Learning



Εικόνα 8 ***Validation accuracy και loss ανά εποχή για Modelnet40***

Στην παραπάνω εικόνα**Error! Reference source not found.** απεικονίζονται τα γραφήματα που συσχετίζουν το loss με τις εποχές και το accuracy του συνόλου επικύρωσης σε σχέση με τις εποχές.

Στην Εικόνα 8 φαίνεται πως το μοντέλο στο οποίο το unfreeze έχει γίνει στο τέλος της έβδομης εποχή καταφέρνει να φτάσει μέχρι 72% accuracy ενώ αντίθετα στο άλλο η μέγιστη τιμή του μέτρου accuracy είναι 78% (unfreeze στο τέλος της δεύτερης εποχής). Αυτή η διαφορά οφείλεται στο γεγονός ότι δίνουμε ένα προ εκπαιδευμένο μοντέλο, το οποίο αναφέρεται σε ένα dataset με 10 εξόδους ενώ αντίθετα το μοντέλο που αναπτύσσουμε έχει 40. Επομένως, η συνεισφορά του προ εκπαιδευμένου μοντέλου είναι σημαντική στην αρχή, ώστε το νέο δίκτυο να μην αρχίσει την εκπαίδευση από τυχαία βάρη, όμως με το πέρας των εποχών η συνεισφορά του μειώνεται. Το νέο δίκτυο πρέπει να εκπαιδευτεί στην αναγνώριση χαρακτηριστικών που δεν υπάρχουν στο προ εκπαιδευμένο. Η ίδια συμπεριφορά παρατηρείται και στην πτώση του loss με την πάροδο των εποχών. Το πρώτο μοντέλο φτάνει σε τελικό loss 1,14 ενώ στο δεύτερο μοντέλο το loss σχεδόν υποδιπλασιάζεται και φτάνει το 0,61.
Ακολουθεί, η σύγκριση των μοντέλων στα οποία το unfreeze έγινε μετά την δεύτερη εποχή αλλά έχουν διαφορετικό learning rate. Τόσο από την τελική τιμή του δείκτη accuracy όσο και από το τελικό loss των μοντέλων, συμπεραίνεται πως ο αλγόριθμος συμπεριφέρεται καλύτερα για lr=0,001. Πιο συγκεκριμένα, ο αλγόριθμος με learning rate 0,01 φτάνει σε τελικό validation accuracy ίσο με 72% και τελικό loss 0,81 ενώ παράλληλα παρουσιάζει έντονες διακυμάνσεις με την πάροδο των εποχών.

Στην συνέχεια παρατίθεται το classification report για το μοντέλο με την εφαρμογή του unfreeze των παραμέτρων στο τέλος της 7ης epoch (αριστερά) και για το μοντέλο με την εφαρμογή του unfreeze των παραμέτρων στο τέλος της 2ης epoch (το learning rate των μοντέλων είναι 0.001).

Στον παρακάτω πίνακα φαίνεται πως το μοντέλο με στο οποίο γίνεται unfreeze μετά τις 7 εποχές αδυνατεί αν προβλέψει την κλάση flower pot, καθώς μόνο το 5% των παραδειγμάτων που ανήκουν σε αυτή την κλάση τα αναγνωρίζει σωστά. Επιπρόσθετα, τις κλάσεις radio, sink, cup, dresser, piano και stool τις αναγνωρίζει σωστά με ποσοστό 15%, 20%, 30%, 37%, 37% και 45% αντίστοιχα. Στις υπόλοιπες κατηγορίες η απόδοσή του είναι ικανοποιητική και γενικά μπορεί να θεωρηθεί ως ένας αρκετά έγκυρος αταξινόμητής.

Επιπλέον, παρατηρούμε ότι το μοντέλο με unfreeze μετά τις 2 εποχές δεν μπορεί καθόλου να προβλέψει την κλάση flower pot (0% σωστά παραδείγματα) ενώ σφάλει και στις κλάσεις door, radio και wardrove με ποσοστό σωστών απαντήσεων 30%, 20% και 35%.

Γενικά και τα δύο μοντέλα αδυνατούν να προβλέψουν σωστά κάποιες κλάσεις ενώ παράλληλα το γενικό ποσοστό του accuracy στο validation set είναι παραπλήσιο. Επιπλέον, όπως είναι λογικό ο χρόνος εκπαίδευσης του δικτύου με unfreeze μετά τις 7 εποχές είναι μικρότερος από το άλλο μοντέλο, καθώς

ο αλγόριθμος δεν μπαίνει στην διαδικασία τροποποίησης των βαρών σε κάποια layers.

Επομένως λαμβάνοντας όλα τα παραπάνω υπόψη, ενδεχομένως το δεύτερο μοντέλο (unfreeze μετά τις 2 εποχές) να θεωρηθεί καλύτερο, διότι φτάνει σε ένα ικανοποιητικό validation accuracy και σφάλει πάνω από 50% μόλις σε τέσσερεις κλάσεις. Αντιθέτως, ενώ το πρώτο (unfreeze μετά τις 7 εποχές) σφάλει σε επτά και έχει ελαφρώς μειωμένο accuracy αλλά σχεδόν διπλάσιο loss.

	precision	recall	f1-score	support
airplane	0.98	1.00	0.99	100
bathtub	0.90	0.76	0.83	50
bed	0.86	0.91	0.88	100
bench	0.86	0.60	0.71	20
bookshelf	0.65	0.73	0.69	100
bottle	0.88	0.91	0.89	100
bowl	0.53	1.00	0.69	20
car	0.85	0.88	0.87	100
chair	0.90	0.95	0.92	100
cone	0.67	0.90	0.77	20
cup	0.43	0.30	0.35	20
curtain	0.48	0.50	0.49	20
desk	0.82	0.59	0.69	86
door	0.62	0.90	0.73	20
dresser	0.71	0.37	0.49	86
flower_pot	0.14	0.05	0.07	20
glass_box	0.62	0.84	0.71	100
guitar	0.99	0.98	0.98	100
keyboard	0.87	1.00	0.93	20
lamp	0.45	0.75	0.57	20
laptop	0.95	1.00	0.98	20
mantel	0.70	0.62	0.66	100
monitor	0.87	0.76	0.81	100
night_stand	0.51	0.51	0.51	86
person	0.71	0.60	0.65	20
piano	0.80	0.37	0.51	100
plant	0.67	0.62	0.64	100
radio	0.33	0.15	0.21	20
range_hood	0.91	0.51	0.65	100
sink	0.57	0.20	0.30	20
sofa	0.94	0.80	0.86	100
stairs	0.71	0.60	0.65	20
stool	0.50	0.45	0.47	20
table	0.52	0.89	0.65	100
tent	0.43	0.75	0.55	20
toilet	0.97	0.78	0.87	100
tv_stand	0.39	0.61	0.47	100
vase	0.61	0.74	0.67	100
wardrobe	0.50	0.55	0.52	20
xbox	0.52	0.70	0.60	20
accuracy			0.72	2468
macro avg	0.68	0.68	0.66	2468
weighted avg	0.74	0.72	0.72	2468

	precision	recall	f1-score	support
airplane	0.99	0.99	0.99	100
bathtub	0.95	0.74	0.83	50
bed	0.96	0.68	0.80	100
bench	0.39	0.70	0.50	20
bookshelf	0.75	0.81	0.78	100
bottle	0.91	0.92	0.92	100
bowl	0.86	0.90	0.88	20
car	0.82	0.97	0.89	100
chair	0.96	0.96	0.96	100
cone	0.95	0.95	0.95	20
cup	0.43	0.50	0.47	20
curtain	0.47	0.80	0.59	20
desk	0.71	0.64	0.67	86
door	1.00	0.30	0.46	20
dresser	0.53	0.55	0.54	86
flower_pot	0.00	0.00	0.00	20
glass_box	0.81	0.60	0.69	100
guitar	0.98	0.99	0.99	100
keyboard	0.70	0.95	0.81	20
lamp	0.76	0.80	0.78	20
laptop	0.95	1.00	0.98	20
mantel	0.85	0.80	0.82	100
monitor	0.89	0.87	0.88	100
night_stand	0.64	0.58	0.61	86
person	0.75	0.75	0.75	20
piano	0.84	0.77	0.80	100
plant	0.69	0.84	0.76	100
radio	0.36	0.20	0.26	20
range_hood	0.99	0.66	0.79	100
sink	0.65	0.65	0.65	20
sofa	0.77	0.86	0.81	100
stairs	0.67	0.60	0.63	20
stool	0.82	0.70	0.76	20
table	0.78	0.88	0.83	100
tent	0.62	0.75	0.68	20
toilet	0.79	0.95	0.86	100
tv_stand	0.56	0.80	0.66	100
vase	0.84	0.71	0.77	100
wardrobe	0.47	0.35	0.40	20
xbox	0.44	0.55	0.49	20
accuracy			0.78	2468
macro avg	0.73	0.73	0.72	2468
weighted avg	0.79	0.78	0.78	2468

Εικόνα 9 ***Classification report για Modelnet40 (a)unfreeze 7 epoch (b)unfreeze 2 epoch***

Επιπρόσθετα, το classification report του μοντέλου με unfreeze στην δεύτερη εποχή και learning rate ίσο με 0,01 εδειξε πως αυτό το μοντέλο δεν ανταποκρίθηκε επαρκώς. Πιο συγκεκριμένα, σε 13 κλάσεις κατάφερε να ταξινομήσει σωστά λιγότερο από το 50% των αποτελεσμάτων τους. Αυτές οι κλάσεις είναι cup, curtain, desk, dress, flower_pot, glass_box, night_stand, piano, mantel, sink, stool, wardrobe και xbox. Για αυτό τον λόγο δεν γίνεται μεγαλύτερη ανάλυση και αναφορά.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Συνοψίζοντας όλα τα παραπάνω μπορούμε να καταλήξουμε στα εξής συμπεράσματα:

- Για την ταξινόμηση του Modelnet10 με την αρχιτεκτονική του δικτύου PointNet καταλήγουμε πως το καλύτερο μοντέλο επιτυγχάνεται με τη χρήση του SGD optimizer και ποσοστό μάθησης (lr = 0,01). Όπως είδαμε πετυχαίνεται validation accuracy 87% και οι μετρικές precision, recall, f1-score είναι αρκετά ικανοποιητικές για όλες τις κλάσεις.

- Για το πρόβλημα της ταξινόμησης του Modelnet40 με την αρχιτεκτονική του PointNet καταλήγουμε πως τα μοντέλα με unfreeze μετά τις 2 και 7 εποχές επέδειξαν παραπλήσια συμπεριφορά. Ωστόσο, το μοντέλο με unfreeze μετά τις 2 εποχές και lr=0.001, λόγω της καλύτερης απόδοσης του (f1-score), μπορεί να θεωρηθεί το πιο αντιπροσωπευτικό για το classification του Modelnet40.

ΠΗΓΕΣ

[1] Charles R. Qi, Hao Su, Kaichun Mo, Leonidas J Guibas, “Pointnet: Deep Learning on Points Sets for 3D Classification and Segmentation”, Cornell University, December 2016.

[2] https://github.com/nikitakaraevv/pointnet