Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Νευρωνικά Δίκτυα & Ευφυή Υ/Σ (Ροή Υ) ~ Χειμερινό Εξάμηνο 2019-20
**Τρίτη Εργαστηριακή Άσκηση ~ Ομάδα MB4**

Αβραμίδης Κλεάνθης ~ 03115117
Κρατημένος Άγγελος ~ 03115025
Πανίδης Κωνσταντίνος ~ 03113602

# 1 Εισαγωγή

Η τρίτη εργαστηριακή άσκηση αποσκοπεί στην εξοικείωση με τη βιβλιοθήκη ανάπτυξης μοντέλων Deep Learning TensorFlow 2.0 και στη βελτιστοποίηση μιας σειράς μοντέλων πάνω στο dataset CIFAR-100 [1]. Η ανάπτυξη, η εκπαίδευση και ο έλεγχος των μοντέλων έγινε μέσω του Jupyter Notebook της εκφώνησης σε Colab, έτσι ώστε να επωφεληθούμε από την επιτάχυνση που προσφέρει στους υπολογισμούς η ενσωματωμένη cloud GPU. Στα παραδοτέα περιλαμβάνεται ο κώδικας του Notebook σε μορφή .ipynb, .py και .html. Αφού μελετήσαμε το αντίστοιχο TensorFlow documentation και τις ιδιότητες του CIFAR, προχωράμε στην εξέταση 2 τεχνικών μάθησης: την υλοποίηση δικών μας μοντέλων "from scratch" και το Transfer Learning.

Τα μοντέλα που περιγράφονται παρακάτω εκπαιδεύτηκαν σύμφωνα με τον κώδικα της εκφώνησης και με τις ακόλουθες baseline παραμέτρους: Adam Optimizer with 0.0001 Learning Rate, Sparse Categorical Cross-Entropy Loss, 128 Batch Size. Επίσης εφαρμόστηκε Early Stopping μέθοδος τερματισμού της εκπαίδευσης. Συγκεκριμένα, κάθε μοντέλο εκπαιδεύεται μέχρις ότου το validation loss δε βελτιώνεται για 25 συνεχόμενες εποχές ή μέχρις ότου παρέλθουν 500 εποχές. Μετά το πέρας της εκπαίδευσης επιστρέφεται όχι το τελευταίο αλλά το καλύτερο μοντέλο μέχρι τότε με βάση το validation loss (checkpointing).

# 2 Data Pre-Processing

Το σύνολο δεδομένων CIFAR-100 αποτελείται από 60000 32X32 έγχρωμες εικόνες που ανήκουν σε 100 κλάσεις (600 εικόνες ανά κλάση). Το dataset χωρίζεται σε train set 50000 εικόνων (500 ανά κλάση) και test set 10000 εικόνων (100 ανά κλάση). Πρόκειται για μια εξέλιξη και σαφώς πιο δύσκολη εκδοχή του dataset CIFAR-10. Οι 100 κλάσεις ομαδοποιούνται σε 20 υπερ-κλάσεις. Κάθε εικόνα λοιπόν έρχεται με ένα fine label για την κλάση και ένα coarse label για την υπερ-κλάση στην οποία ανήκει.

Συνηθίζουμε σε Computer Vision tasks να εφαρμόζουμε augmentation στα δεδομένα μας έτσι ώστε τα διαθέσιμα μοντέλα όχι μόνο να έχουν περισσότερα data προς εκμάθηση, αλλά συγχρόνως να μάθουν αποτελεσματικά πιο γενικευμένα features από ένα πιο ευρύ και ποικιλόμορφο dataset. Στη δική μας περίπτωση εφαρμόσαμε τον ImageDataGenerator στην αρχή του κώδικα, μόλις γίνεται η εισαγωγή του dataset. Έτσι όλα τα μοντέλα εκπαιδεύονται εξαρχής με augmentation ενώ παρατηρείται αύξηση του χρόνου εκπαίδευσης κατά 2 με 3 φορές, πράγμα λογικό αφού τα data μας έχουν πολλαπλασιαστεί. Το augmentation που εφαρμόσαμε αφορά στον καθρέφτισμα της εικόνας ως προς τον οριζόντιο άξονα, ζουμάρισμα, περιστροφή και shift κατά μήκος και πλάτος. Τέλος να αναφέρουμε ότι ο Generator έχει την επιλογή για scaling (1/255) το οποίο ουσιαστικά κανονικοποιεί τις τιμές κάθε εικόνας στο [0,1]. Αυτό αποφεύχθηκε όμως γιατί έχει πραγματοποιηθεί νωρίτερα κατά τη φόρτωση του dataset.

Όσον αφορά το χρόνο εκτέλεσης, διερευνήσαμε τρόπους με τους οποίους θα επισπεύσουμε τους χρόνους εκπαίδευσης, λόγω και της επαύξησης που προαναφέρθηκε. Συγκεκριμένα, έχει εφαρμοστεί prefetching ούτως ώστε τα δεδομένα που αναμένεται να εκτελεστούν σύντομα να προετοιμάζονται και να γίνονται άμεσα διαθέσιμα. Μια άλλη από τις διαθέσιμες τεχνικές του TensorFlow που εφαρμόσαμε είναι το caching, δηλαδή η αποθήκευση των δεδομένων σε cache προκειμένου να μην απαιτείται πολύς χρόνος για την επαναφόρτωσή τους σε κάθε εποχή εκπαίδευσης.

Παρατηρούμε τέλος ότι στη περίπτωσή μας είναι εφικτή η απευθείας φόρτωση του συνόλου των images και κατά συνέπεια δεν απαιτείται κάποια περαιτέρω επεξεργασία σε επίπεδο μνήμης και αποθήκευσης. Αυτό φυσικά δε συμβαίνει πάντοτε, ιδίως όταν έχουμε tasks με video (Action Recognition, Sign Language Recognition, κλπ). Με τον καθιερωμένο τρόπο εκτέλεσης των προγραμμάτων μάθησης τα δεδομένα αυτά αποθηκεύονται απευθείας σε μεταβλητές στην RAM της CPU/GPU που χρησιμοποιούμε. Αυτό καθιστά δύσκολη έως αδύνατη τη διαχείρισή τους. Σε αυτή τη περίπτωση υπάρχουν διάφορες τεχνικές που μπορούν να εφαρμοστούν όπως αυτής της τμηματικής φόρτωσης των δεδομένων σπάζοντας το dataset σε επιμέρους αρχεία. Μια άλλη μέθοδος είναι το format TFRecord του TensorFlow που επιτρέπει την σειριακή αποθήκευση δυαδικών εγγραφών. Μια εξίσου αποτελεσματική τεχνική είναι η φόρτωση των διευθύνσεων των δεδομένων αντί των ίδιων. Αυτό σημαίνει ότι το train set θα αποτελείται από ένα x_train και ένα y_train, όπου το δεύτερο θα περιλαμβάνει τα labels, ενώ το πρώτο απλά τις διευθύνσεις όπου βρίσκονται αποθηκευμένα τα data (./path/image1, ./path/image2, etc). Στη συνέχεια δίνουμε στο μοντέλο μας τα δεδομένα μέσω ενός train and validation generator, με περιορισμένο batch size όπου αυτός φροντίζει να φορτώνει την πραγματική εικόνα για κάθε path που του παρέχεται.

# 3 Models from Scratch

Για την ταξινόμηση εικόνων του CIFAR-100 (ενός μοναδικού υποσυνόλου σύμφωνα με το seed μας) θα χρησιμοποιήσουμε συνελικτικά νευρωνικά δίκτυα – CNN, καθώς τα τελευταία χρόνια έχουν αναδειχθεί ως τα πλέον αποτελεσματικά μοντέλα στην ανάλυση και ταξινόμηση δισδιάστατων σημάτων όπως οι εικόνες. Στην ενότητα αυτή θα περιγράψουμε τις αρχιτεκτονικές των μοντέλων και τις παραμέτρους που χρησιμοποιούμε για την εξαγωγή ενός baseline result. Η βελτιστοποίηση και σύγκριση επίδοσης των διάφορων παραμέτρων γίνεται στην ενότητα 4. Εξετάζουμε για αρχή το μοντέλο **simple** από την εκφώνηση:

Το **simple** είναι ένα απλό Sequential δίκτυο με 3 συνελικτικά και 2 πυκνά επίπεδα. Η αρχιτεκτονική αναλυτικά:

1. 2D Standard Convolution Layer with 32 output filters, ReLU activation and 3X3 Convolution Kernel
2. 2D Max Pooling Layer over a 2X2 Kernel
3. 2D Standard Convolution Layer with 64 output filters, ReLU activation and 3X3 Convolution Kernel
4. 2D Max Pooling Layer over a 2X2 Kernel
5. 2D Standard Convolution Layer with 64 output filters, ReLU activation and 3X3 Convolution Kernel
6. Flattening Layer in order to adjust the next Dense Layers
7. Dense Layer with 64 output cells and ReLU activation
8. Dense Layer with 100 output cells (1 for each class) and Softmax activation for probabilistic output

Εξετάζουμε τώρα το μοντέλο **basic** που σχεδιάσαμε με βάση τη βιβλιογραφία σχετικά με baseline CNN αρχιτεκτονικές. Πρόκειται για ένα πολυεπίπεδο μοντέλο με 3 convolutional hyper-layers και 2 dense layers στο τελείωμα. Συγκεκριμένα:

1. 2D Standard Convolution Layer with 32 output filters, ReLU activation and 2X2 Convolution Kernel
2. 2D Standard Convolution Layer with 64 output filters, ReLU activation and 2X2 Convolution Kernel
3. Batch Normalization Layer
4. LeakyReLU activation Layer with $\alpha$=0.3
5. 2D Max Pooling Layer over a 2X2 Kernel
6. Dropout Layer with 0.2 rate

7. 2D Standard Convolution Layer with 128 output filters, ReLU activation and 2X2 Convolution Kernel
8. Batch Normalization Layer
9. LeakyReLU activation Layer with $\alpha$=0.3
10. 2D Max Pooling Layer over a 2X2 Kernel
11. Dropout Layer with 0.2 rate

12. 2D Standard Convolution Layer with 256 output filters, ReLU activation and 3X3 Convolution Kernel
13. Batch Normalization Layer
14. LeakyReLU activation Layer with $\alpha$=0.3
15. 2D Max Pooling Layer over a 2X2 Kernel
16. Dropout Layer with 0.3 rate

17. Flattening Layer in order to adjust the next Dense Layers
18. Dense Layer with 1024 output cells
19. LeakyReLU activation Layer with $\alpha$=0.3
20. Dropout Layer with 0.5 rate
9. Dense Layer with 100 output cells and Softmax activation for probabilistic output

Τα μοντέλα εκπαιδεύτηκαν σύμφωνα με τις προδιαγραφές που περιγράφησαν στην ενότητα 1 και δίνουν τα εξής αποτελέσματα:

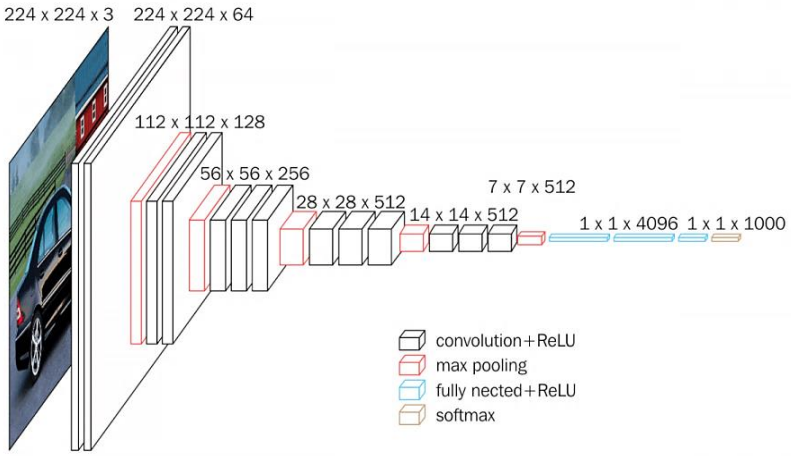
- Simple: 1.40 Test Loss, 0.59 Test Accuracy
- Basic: 1.11 Test Loss, 0.66 Test Accuracy

# 4 Transfer Learning Models

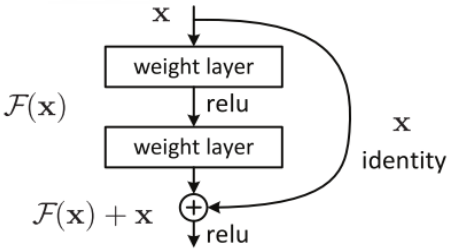
Η τεχνική της μεταφοράς μάθησης είναι μια σημαντική μέθοδος που έχει αναπτυχθεί τα τελευταία χρόνια για να επιλύσει ένα από τα κυριότερα προβλήματα στο χώρο του Machine Learning, αυτό της έλλειψης διαθέσιμων δεδομένων σε πολλά συνήθως πιο specialized tasks. Με αυτή τη τεχνική αξιοποιείται η «γνώση» μοντέλων που έχουν εκπαιδευθεί σε μεγάλα datasets ενός γενικότερου task ούτως ώστε να γίνει πιο εύκολη και πιο αντιπροσωπευτική η εκπαίδευση σε πιο εξειδικευμένα tasks που δέχονται ίδιας μορφής δεδομένα, αλλά τα διαθέσιμα datasets είναι ανεπαρκή. Για παράδειγμα ένα σύστημα αναγνώρισης πινακίδων οδικής κυκλοφορίας που μπορεί να αξιοποιηθεί στην αυτόνομη οδήγηση είναι πολύ αποδοτικότερο να εκκινήσει την εκπαίδευσή του

από ένα προεκπαιδευμένο μοντέλο αναγνώρισης εικόνων, όπως το ImageNet που θα χρησιμοποιήσουμε και εδώ. Συγκεκριμένα, επιλέξαμε 3 ευρέως χρησιμοποιούμενα μοντέλα για να εφαρμόσουμε transfer learning, προεκπαιδευμένα στο ImageNet:

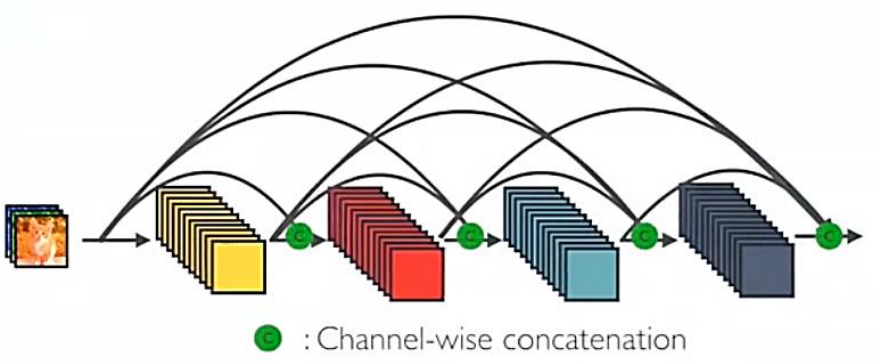
Το πρώτο μοντέλο που θα εξετάσουμε είναι το **VGG16** που δίνεται στην εκφώνηση. Πρόκειται για ένα CNN που προτάθηκε από τους K. Simonyan, A. Zisserman [2] και πετυχαίνει 92.7% accuracy στο ImageNet dataset (14 εκ. εικόνες 1000 κλάσεων). Το μοντέλο αυτό βελτιώνει το AlexNet χρησιμοποιώντας large-scale kernel φίλτρα στα πρώτα συνελικτικά επίπεδα. Δίνουμε μια σύνοψη της αρχιτεκτονικής του, όπου βλέπουμε πως ακολουθεί τη δομή των layers που χρησιμοποιήσαμε παραπάνω. Πρακτικά, μια αρχιτεκτονική όπως αυτή του **basic** μοντέλου είναι μια VGG-like αρχιτεκτονική:

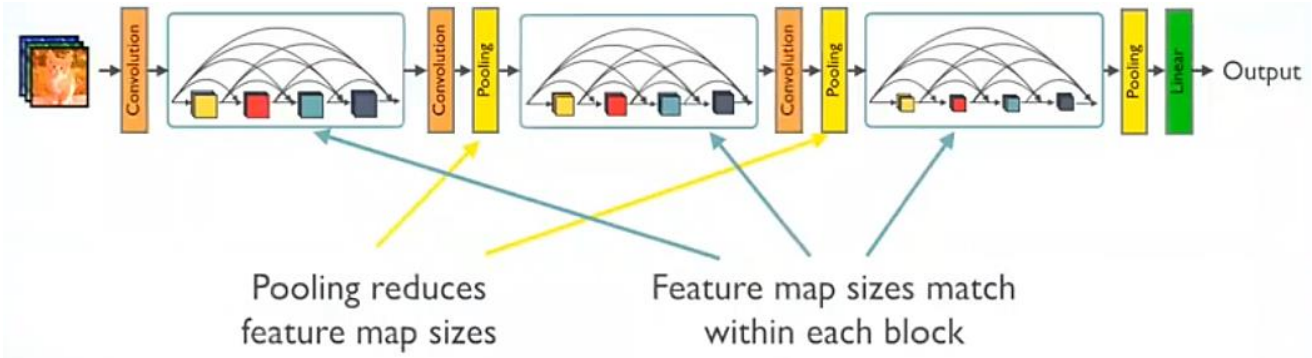



Εξετάζουμε τώρα το μοντέλο **ResNet** [3], ένα ισχυρό backbone μοντέλο που χρησιμοποιείται συχνά σε computer vision tasks, όπως αυτό της εργαστηριακής άσκησης. Η ιδιαιτερότητα αυτού του μοντέλου είναι το identity shortcut connection, η σύνδεση δηλαδή του αποτελέσματος ενός layer με ένα «μακρινό» layer, αγνοώντας τα αμέσως επόμενα στη σειρά. Αυτή η προδιαγραφή βοηθά στην αντιμετώπιση του προβλήματος του vanishing gradient:




Το τελευταίο μοντέλο που θα παρουσιάσουμε είναι το **DenseNet** [4]. Σε αντίθεση με την προηγούμενη αρχιτεκτονική όπου ένα επίπεδο μπορεί να δέχεται input από ένα οποιοδήποτε προηγούμενο, εδώ δέχεται πολλαπλά inputs από όλα τα προηγούμενα επίπεδα, προσομοιάζοντας ένα είδος επεξεργασίας συλλογικής γνώσης. Αυτό επιτρέπει στο δίκτυο να είναι απλούστερο σε δομή και απαιτήσεις μνήμης και υπολογισμών. Η αρχιτεκτονική του δικτύου περιλαμβάνει παρόμοια μοτίβα με τα προηγούμενα. Ένα αρχικό συνελικτικό επίπεδο ακολουθεί η παραπάνω δομή, επαναληπτικά παρεμβάλλοντας επίπεδα Batch Normalization, ReLU Activation και Convolution:



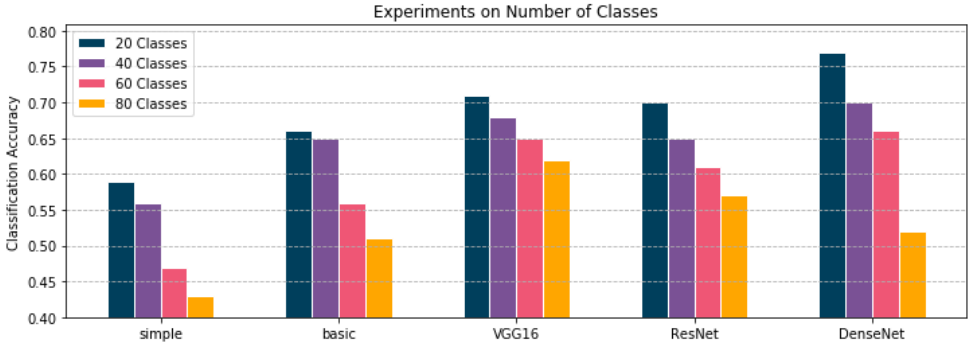




Για την εκπαίδευση των παραπάνω μοντέλων με μεταφορά μάθησης χρησιμοποιήθηκαν οι default τιμές των παραμέτρων που αναφέρθηκαν παραπάνω. Επιπροσθέτως, τα βάρη των εκπαιδευμένων δικτύων κρατούνται frozen, δηλαδή δεν μεταβάλλονται κατά την επανεκπαίδευση. Τα μοντέλα φορτώθηκαν από τις βιβλιοθήκες του TensorFlow και συμπληρώθηκαν το καθένα με ένα Dropout layer για να μειωθεί το overfitting στο ImageNet και ένα Dense layer για την ορθή επανεκπαίδευση.

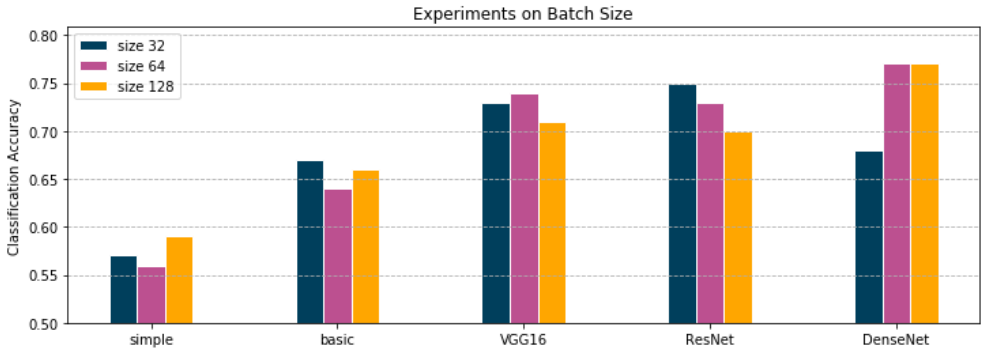
# 5 Βελτιστοποίηση & Παρατηρήσεις

Όσον αφορά τις μεθόδους βελτιστοποίησης των μοντέλων και της εκπαίδευσης, έχουμε ήδη προβεί σε ενέργειες όπως η αξιοποίηση Dropout [5] επιπέδων (random μηδενισμός ενός ποσοστού των βαρών) και Early Stopping/Checkpointing που μεταξύ άλλων μειώνουν το overfitting. Θα διερευνήσουμε τις τιμές ορισμένων κρίσιμων παραμέτρων για την εκπαίδευση και απόδοση των μοντέλων. Τα συγκριτικά πειράματα ήταν γενικά μικρής κλίμακας και έγιναν manually, χωρίς keras tuner:

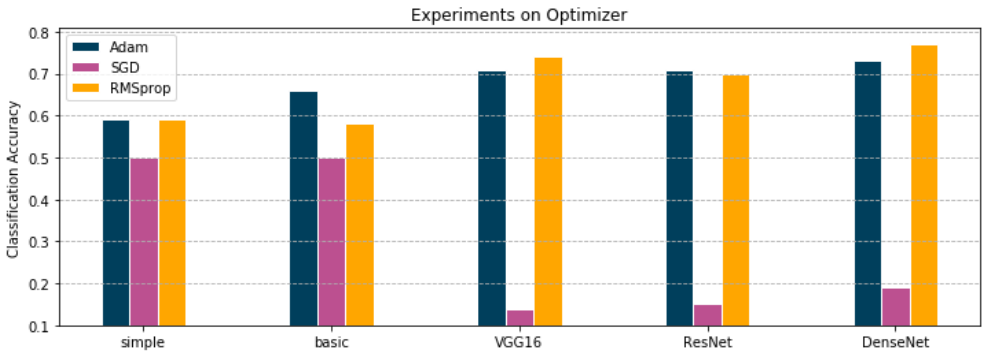
- Number of Classes



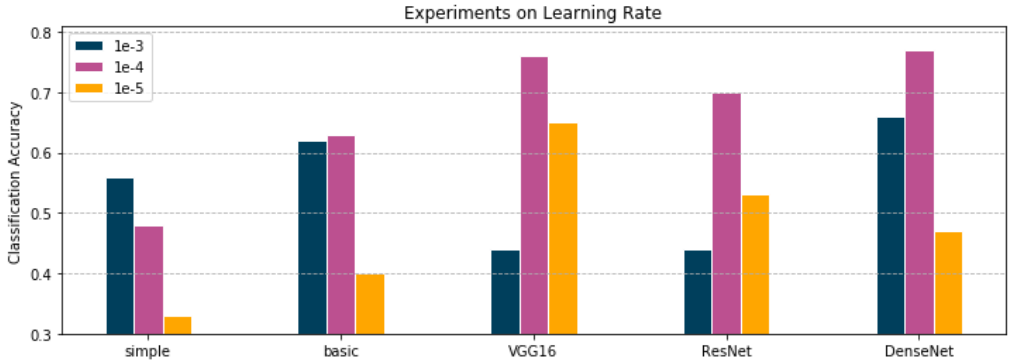

- Batch Size



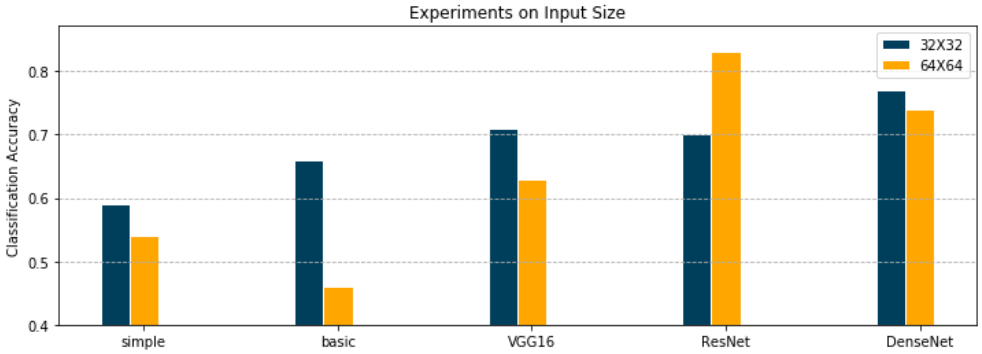

- Optimizer




- Learning Rate

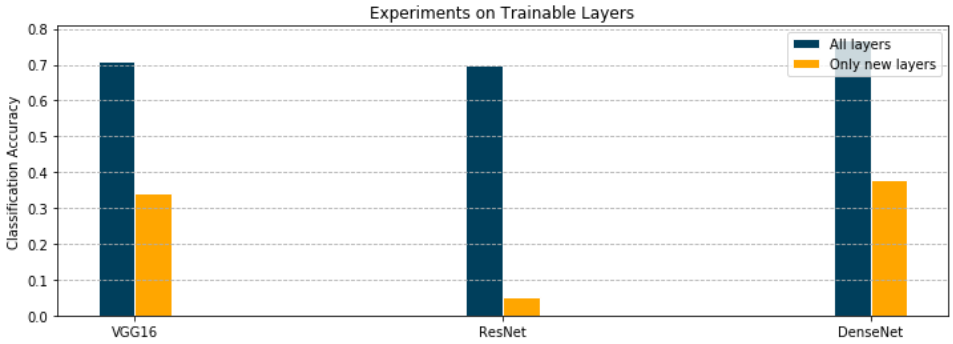


- Resized Input




Προχωράμε σε σειρά παρατηρήσεων με βάση τα παραπάνω αποτελέσματα: Είναι φανερό ότι όσο αυξάνουμε τον αριθμό των κλάσεων τόσο μειώνεται το accuracy, το οποίο ισχύει για όλα τα μοντέλα. Μικρές αλλαγές στο batch size φαίνεται να μην επηρεάζουν ιδιαίτερα την επίδοση των νευρωνικών. Παρόλα αυτά, ακραίες τιμές (1 ή π.χ. 512) είναι πιθανό να επιδράσουν αρνητικά στο αποτέλεσμα. Ύστερα, ο Stochastic Gradient Descent Optimizer φαίνεται να αποτυγχάνει εντελώς σε μοντέλα που χρησιμοποιούν transfer learning. Οι Adam και RMSprop έχουν πολύ παρόμοια επίδοση, ενώ ο SGD φαίνεται να υστερεί συνολικά, πιθανόν λόγω της απλότητάς του. Το learning rate επιδρά ως εξής: Πολύ μικρά learning rate βλέπουμε ότι δεν αφήνουν τα μοντέλα να κάνουν converge για fixed αριθμό εποχών (πράμα που παρατηρείται και από τις καμπύλες των validation losses). Είναι πιθανό με αύξηση των εποχών να είχαμε ισοδύναμα αποτελέσματα με τα άλλα learning rate. Από την άλλη μικρά learning rate της τάξης $10^{-3}$ δε βοηθούν επίσης στον εντοπισμό τοπικού ελαχίστου λόγω μεγάλων «αλμάτων». Έτσι το $10^{-4}$ κρίνεται το πλέον αποτελεσματικό. Τέλος συμπεραίνουμε πως η αλλαγή στο input size δε φαίνεται να προσδίδει καθαρότερες εικόνες, εξαιρώντας το μοντέλο ResNet το οποίο υπό αυτές τις συνθήκες πετυχαίνει το βέλτιστο της άσκησης 83% accuracy. Επικεντρώνουμε τώρα στην τεχνική μεταφοράς μάθησης και διερευνούμε επιπλέον τα εξής:

- Number of Trainable Layers




Όπως περιμέναμε, από τα αποτελέσματα των πειραμάτων φαίνεται ότι η τεχνική του Transfer Learning βοηθάει στην εκμάθηση χαρακτηριστικών και στην τελική επίδοση στο test set. Ο λόγος της επιτυχίας του είναι όπως προείπαμε η δυνατότητα αξιοποίησης γνώσης από πολύ βαθιά και πολύπλοκα δίκτυα σε πολύ μεγάλες βάσεις, όσο και στο γεγονός ότι τα νευρωνικά αυτά μαθαίνουν εξαιρετικά χρήσιμα features πάνω στις εικόνες, τα οποία εμείς δανειζόμαστε για το δικό μας υποπρόβλημα. Όσον αφορά το ποια επίπεδα εκπαιδεύονται παρατηρούμε ότι όταν δεν εκπαιδεύουμε τα νέα layers έχουμε κατακόρυφη πτώση της επίδοσης. Απαιτείται λοιπόν η εκπαίδευση του νευρωνικού στο σύνολο των επιπέδων του, κάτι που αποδεικνύει την σημασία που έχει ο τρόπος αρχικοποίησης των βαρών τέτοιων μοντέλων.

# 6 Πηγές ~ Βιβλιογραφία

[ 1 ] https://www.cs.toronto.edu/~kriz/cifar.html

[ 2 ] Simonyan, K. & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition

[ 3 ] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778.

[ 4 ] G. Huang, Z. Liu, L. v. d. Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 2261-2269.

[ 5 ] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1 (January 2014), 1929–1958.