Σκοπός της εργασίας

Σκοπός της εργασίας είναι η εφαρμογή μεθόδων κατανεμημένης εκπαίδευσης βαθιών νευρωνικών δικτύων (DNN) με χρήση της βιβλιοθήκης TensorFlow , ώστε να μελετηθεί πως ο αριθμός των κατανεμημένων κόμβων εκπαίδευσης επηρεάζει τον χρόνο εκπαίδευσης των μοντέλων.

Μέθοδος κατανεμημένης εκπαίδευσης

Η τεχνική που χρησιμοποιούμε για την κατανεμημένη εκπαίδευση είναι η MultiWorkerMirroredStrategυ() από το distribute πακέτο του tensorflow.

Συνοπτικά ορίζουμε την TF\_CONFIG μεταβλητή περιβάλλοντος με την λίστα με τα ip adresses των workers , με τον ρόλο και τον αύξων αριθμό του κάθε worker . Στην συγκεκριμένη περίπτωση έχουμε μέχρι 3 workers (τα 3 vms που μας παραχωρήθηκαν στον ωκεανό)

Οι ips τους είναι :

* VM1 : 192.168.1.1:2222
* VM2 : 83.212.80.22:2222 (master vm)
* VM3 : 192.168.1.3:2222

Specs των Vms :

* RAM:
* CPU core number :
* Disk Space :

Άμα δεν έχουμε κατανεμημένη εκπαίδευση το VM που χρησιμοποιείται είναι το VM2.

Άμα έχουμε κατανεμημένη εκπαίδευση σε 2 workers τα VMs που χρησιμοποιούνται είναι το VM1 και τοVM2.

Άμα έχουμε κατανεμημένη εκπαίδευση σε 3 workers, χρησιμοποιούνται όλα τα VMs (δηλαδή και το VM1 και τοVM2 και το VM3)

Η διαδικασία λοιπόν είναι :

Ταυτόχρονα ορίζονται οι TF\_CONFIG στα vms τα οποία θα χρησιμοποιηθούν για κατανεμημένη εκπαίδευση με την χρήση του run\_distributed\_training.sh script το οποίο δέχεται σαν όρισμα τον αριθμό των workers. (πρέπει να δοθεί και σαν όρισμα τό όνομα του μοντέλου για να περαστεί στο worker.py)

Στην συνέχεια εκτελείται αυτόματα ( γίνεται trigger από το run\_distributed\_training.sh)o σε κάθε επιλεγμένο VM το python script worker.py , το οποίο δέχεται σαν όρισμα τον αριθμό των workers και το όνομα του dataset με βάση το οποίο θα εκπαιδεύσουμε το αντίστοιχο μοντέλο.

Η MultiWorkerMirroredStrategy() στρατηγική του tensorflow την οποία χρησιμοποιούμε για κατανεμημένη εκπαίδευση είναι μια στρατηγική που χρησιμοποιείται για σύγχρονη εκπαίδευση , με βάση την οποία δημιουργούνται αντίγραφα όλων των μεταβλητών του μοντέλου σε όλους τους workers , εκπαιδεύονται αυτές οι μεταβλητές σε ένα μέρος του dataset σε κάθε worker και στο τέλος της κάθε εποχής οι μεταβλητές (που πλέον είναι διαφορετικές σε κάθε VM) , “συγχωνεύονται” και ξαναποκτούν τις ίδιες τιμές σε όλους τους workers. Μετά ξεκινά μια νέα εποχή εκπαίδευση και αυτό συνεχίζεται για τον αριθμό των εποχών που έχουμε ορίσει εμείς.

Έτσι μπορούν ξεχωριστά τμήματα του ίδιου dataset να εκπαιδεύονται ταυτόχρονα σε διαφορετικά μηχανήματα (στην δική μας περίπτωση σε εώς 3 διαφορετικά VMS)

Μοντελα και DATASET

Τα μοντέλα που χρησιμοποιήθηκαν είναι τα πλέον ενδεδειγμένα και ευρέως διαδεδομένα για την πιο επιτυχημένη πρόβλεψη του κάθε dataset με το οποίο επιλέξαμε να εργαστούμε.

Τα dataset που επιλέξαμε είναι 5 από τα πιο διαδεδομένα opensource στο χώρο των DNN και παρακάτω ακολουθεί λίστα με το όνομα του κάθε dataset και το είδος του μοντέλου που επιλέχθηκε , το batch size και ο αριθμός των εποχών που θεωρήθηκαν τα πιο κατάλληλα για την εκπαίδευση :

**Σημείωση** : το input name είναι το όνομα που πρέπει να δοθεί στο run\_distributed\_training.sh για να επιλέξουμε το συγκεκριμένο dataset- μοντέλο.

1)

* Input name : cifar\_10
* Dataset : CIFAR-10

ένα dataset του πανεπιστημίου του Toronto , που αποτελείται από 6000 32Χ32 pixels , έγχρωμες εικόνες που χωρίζονται σε 10 κλάσεις με 6000 εικόνες ανά κλάση .

* Model : ένα συνελλικτικό νευρωνικό δίκτυο 20 επιπέδων ( 6 Conv2d , 6 BatchNormalization ,

3 MaxPooling2D , 1 Flatten , 2 Dropout , 2 Dense ). Η αρχιτεκτονική του μοντέλου που μόλις περιγράψαμε όπως χρησιμοποιήθηκε από εμάς φαίνεται παρακάτω.



Batch size : 256 / (αριθμό worker nodes)

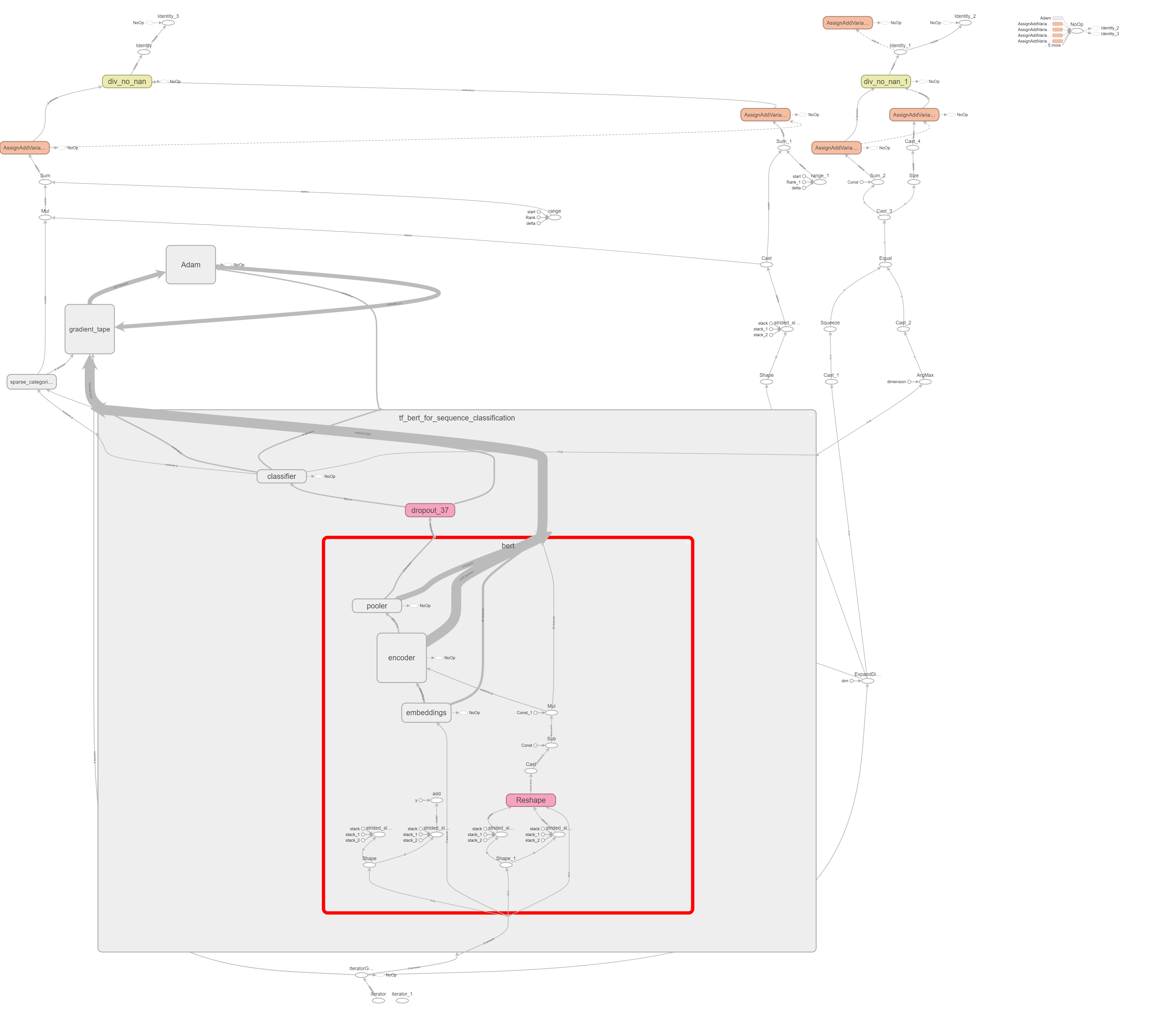
* Epochs : 5

2)

* Input name : bert\_imdb
* Dataset :IMDb sentiment analysis dataset

ένα dataset που αποτελείται από 50000 reviews ταινιών από το site imdb , τα reviews κατηγοριοποιούνται είτε ως θετικά είτε ως αρνητικά.

* Model : ΒERT , ένα μοντέλο που χρησιμοποιείται για ανάλυση κειμένου και βασίζεται πάνω σε transformers . Η αρχιτεκτονική του bert όπως χρησιμοποιήθηκε από εμάς φαίνεται παρακάτω.

Σημειώνουμε πως λόγω περιορισμών της ram μόνο το τελευταίο επίπεδο του bert είναι εκπαιδεύσιμο και τα υπόλοιπα τα πήραμε ως pretrained.

* Batch size :512 / (αριθμό worker nodes)
* Epochs : 5

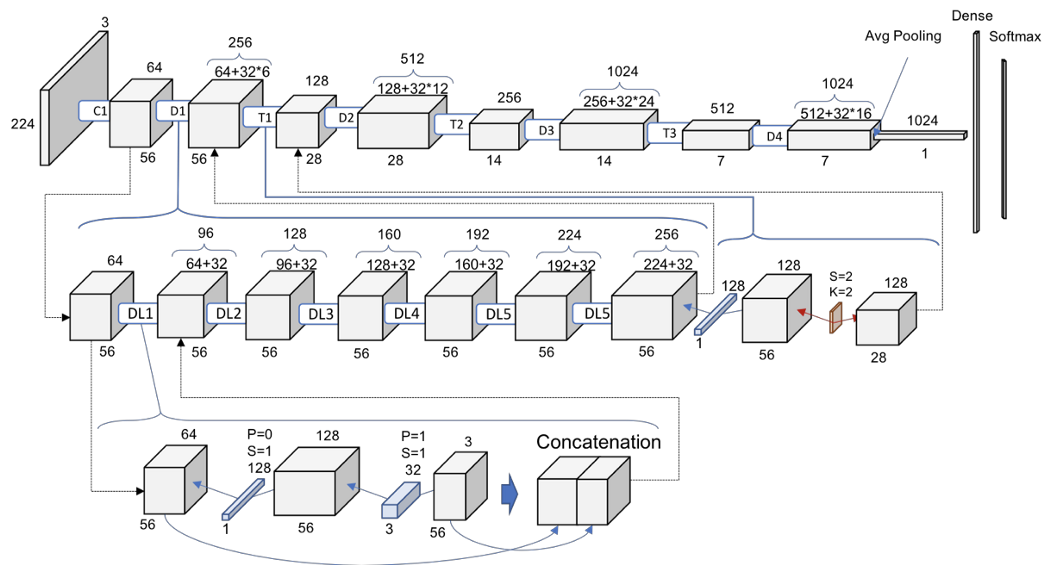
3)

* Input name :natural\_images\_densenet
* Dataset : Natural Images

ένα dataset 6889 έγχρωμων εικόνων σχετικά υψηλής ανάλυσης οι οποίες χωρίζονται σε 8 κλάσεις.

* Model : Densenet121 , ένα CNN που το κάθε επίπεδο συνδέεται με όλα τα άλλα επίπεδα . Αποτελείται από 1 7Χ7 Conv2d , 58 3Χ3 Conv2d , 61 1x1 Conv2D , 4 AvgPool , 1 Fully connected layer.

Η αρχιτεκτονική του densenet121 όπως χρησιμοποιήθηκε από εμάς φαίνεται παρακάτω.

  
 Επίσης για την δική μας υλοποίηση προστέθηκαν στο τέλος του densenet121 , ένα επίπεδο GlobalAveragePooling2D , 2 BatchNormalization , 2 Dropout , 2 Dense

* Batch size :512 / (αριθμό worker nodes)
* Epochs :5

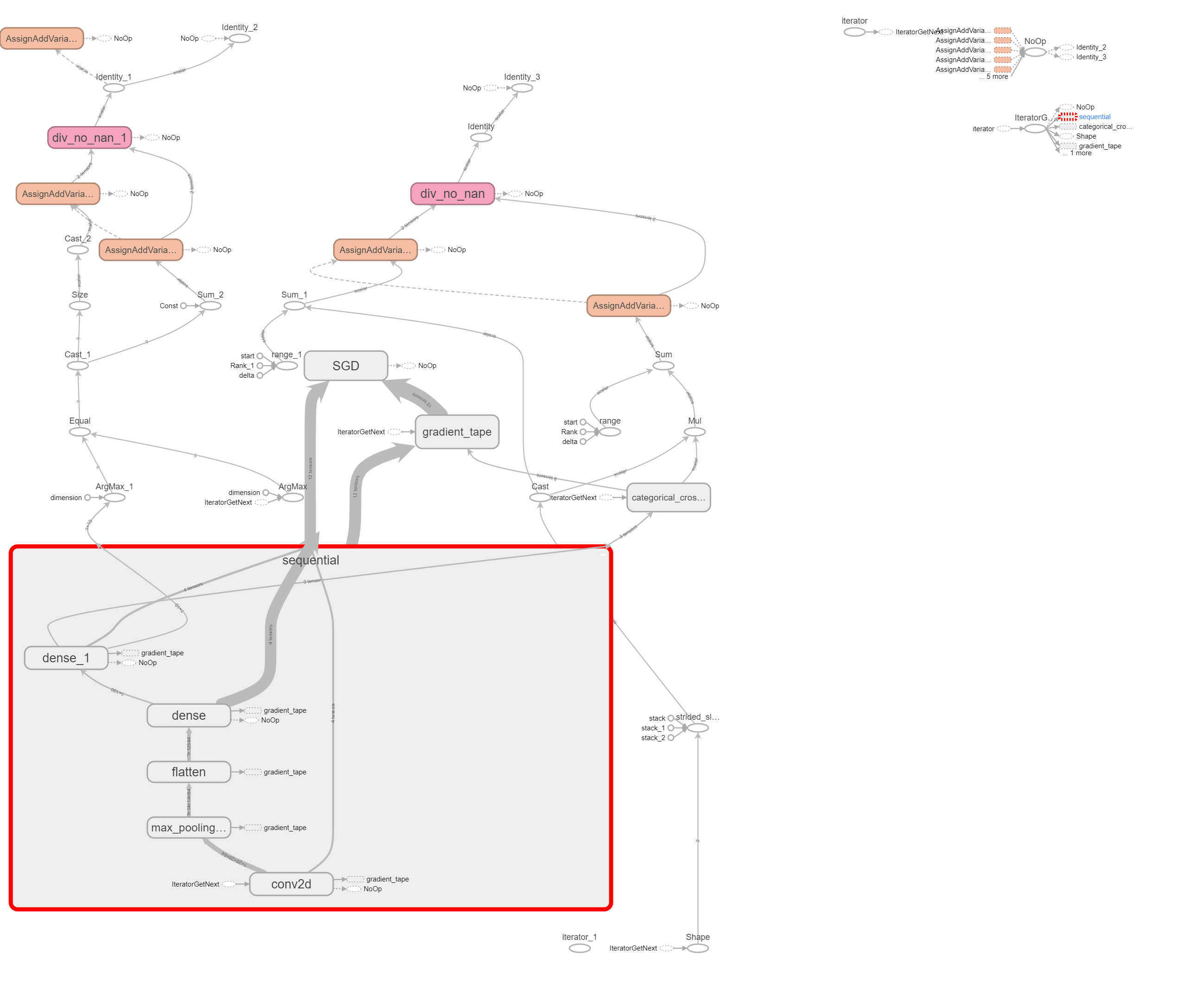
4)

* Input name : fashion\_mnist
* Dataset : Fashion- MNIST ,

ένα dataset 60000 μαυρόασπρων εικόνων μεγέθους 28Χ28 pixels ειδών μόδας , τα οποία χωρίζονται σε 10 κλάσεις.

* Model : χρησιμοποήθηκε ένα απλό CNN μόλις 5 επιπέδων ( 1 Conv2d ,

1 MaxPooling2D , 1 Flatten , 2 Dense) για να δούμε την επίδραση της κατανεμημένης εκπαίδευσης και σε σχετικά ρηχά νευρωνικά. Η αρχιτεκτονική του απλού αυτού CNN φαίνεται και παρακάτω.



* Batch size : 512 / (αριθμό worker nodes)
* Epochs : 5

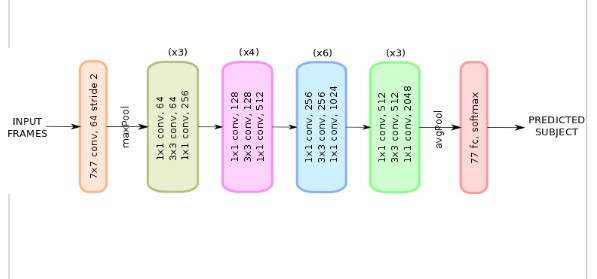
5)

* Input name : mnist
* Dataset : MNIST database of handwritten digits

60000 μαυρόασπρες εικόνες 28Χ28 pixels , χειρόγραφων ψηφίων από το 0 εώς το 9.

* Model :ResNet50 , ένα CNN 50 επιπέδων ( 48 Conv2d , 1 MaxPool , 1 AvgPool) .

Η αρχιτεκτονική του densenet121 όπως χρησιμοποιήθηκε από εμάς φαίνεται παρακάτω.



* Batch size :512 / (αριθμό worker nodes)
* Epochs :5

Αποτελέσματα

Για κάθε dataset θα παρασταθούν 2 διαγράμματα για την κατανεμημένη εκπαίδευση του αντίστοιχου μοντέλου .

Ένα του συνολικού χρόνου εκπαίδευσης συναρτήσει του αριθμού των worker -nodes. (ίσως και ανα εποχή)

Ένα της επι τοις εκατό μείωσης του χρόνου εκπαίδευσης σε σχέση με την μη κατανεμημένη εκπαίδευση συναρτήσει του αριθμού των worker -nodes.

Σχολιασμός αποτελεσμάτων :

Παρατηρούμε πως η εκπαίδευση σε Χ αριθμό κόμβων δεν οδηγεί σε μείωση του χρόνου εκπαίδευσης στον αρχικό δια Χ. Αυτό συμβαίνει γιατί υπάρχει και ένα overhead για ενημέρωση και “συμπτηξη” των ξεχωριστών βαρών σε κάθε κόμβο για την δημιουργία των κοινών βαρών Αυτό το overhead φίνεται να οφείλεται για περίπου Υ % του συνολικού χρόνου στην κατανεμημένη εκπαίδευση σε 2 κόμβους και για Ζ % στους 3.

References :

[1] https://lambdalabs.com/blog/tensorflow-2-0-tutorial-05-distributed-training-multi-node/?fbclid=IwAR33hA3V4M7-eMqikBvlH7OIwWTOa-F5eXHmdbL2908hM1Z\_uE9oveRbcH0

[2] [https://colab.research.google.com/github/tensorflow/tpu/blob/master/tools/colab/fashion\_mnist.ipynb#scrollTo=dgAHfQtuhddd](https://colab.research.google.com/github/tensorflow/tpu/blob/master/tools/colab/fashion_mnist.ipynb" \l "scrollTo=dgAHfQtuhddd)

[3] https://www.geeksforgeeks.org/cifar-10-image-classification-in-tensorflow/

[4] https://www.tensorflow.org/tutorials/distribute/multi\_worker\_with\_keras

[5] https://www.appservgrid.com/paw92/index.php/2020/06/06/how-to-setup-ssh-without-passwords-linux-hint/

[6]https://github.com/indiantechwarrior/Resnet\_Model\_TF2.0

[7]https://keras.io/api/applications/densenet/

[8]https://www.kaggle.com/datasets/prasunroy/natural-images

[9]https://www.ionos.com/help/server-cloud-infrastructure/getting-started/important-security-information-for-your-server/changing-the-default-ssh-port/

[10]http://yann.lecun.com/exdb/mnist/